Morphing STARs vs drones and weather in TMA

Henrik Hardell
Communications and Transport Systems, ITN,
Linköping University, Norrköping, Sweden
Procedure Design Unit, Luftfartsverket (LFV),
Norrköping, Sweden

Anastasia Lemetti, Tatiana Polishchuk and Valentin Polishchuk
Communications and Transport Systems, ITN,
Linköping University, Norrköping, Sweden

Vishwanath Bulusu
Crown Consulting Inc.,
Moffett Field, California, USA
Cal Unmanned Lab, University of California,
Berkeley, California, USA

Enric Royo
The School of Industrial, Aerospace and
Audiovisual Engineering of Terrassa (ESEIAAT),
TU Barcelona, Spain

***Abstract*—We present an optimization framework for computing STARs that slowly change over time, while always avoiding a set of moving obstacles in TMA. The framework is applied to two types of obstacles: a drone intruder and hazardous weather. We demonstrate the output of our algorithms on synthesized drone intrusion incidents and real storm cells in Stockholm Arlanda terminal area.**

Keywords*—TMA route design; ATM and UTM; Weather avoidance

I. Introduction

Drones interfering with aircraft on approach are reported with alarming frequency. Such interference poses serious risk to development of the emerging drone industry, as it negatively impacts public acceptance of UAV (Unmanned Aerial Vehicle) operations. Economic damage and passenger discomfort is especially aggravated when drone sighting leads to cancellation of flights through the *whole* terminal airspace and prolonged freeze of airport operations.

We are thus motivated to consider closing only that *part* of the airspace where the drone is physically present and *recompute* the arrival tree so that the aircraft safely land, avoiding the drone-affected part. The challenge is that the drone is moving, implying that the drone-impacted area is a *moving* obstacle. In addition, since the drone may be spotted only once, its uncertainty region grows, meaning that the drone obstacle not only moves but also may expand with time. This makes the drone deconfliction similar to weather avoidance i.e. escaping both kinds of obstacles fits into the same abstract model.

In this paper we extend the optimization framework from [1] for constructing operationally feasible obstacle-avoiding *static* standard arrival routes (STAR), to handle obstacles *changing with time* and to output *morphing* STARs that never change abruptly (so controllers and pilots may easily see how the STARs evolve over time). Our mathematical model for the morphing STARs works with any type of abstract obstacles. To illustrate it, we present experimental results of application of the morphing STARs to drone deconfliction and to weather avoidance in the terminal maneuvering area (TMA) of Stockholm Arlanda airport. While the drone intrusion scenario is synthesized, for the weather data we took a real storm.

The paper is organized as follows. The remainder of this section reviews related work. Section II formalizes the model and Section III casts our problem as an integer linear program (IP). In Section IV we apply the IP to drone and weather avoidance in Stockholm Arlanda TMA. Section V concludes the paper.

A. Related work

Technologies proposed to detect, track and shoot down drones include radars, cameras, eagles [2], counter-drone UAVs [3] and jamming [4] – a comprehensive survey can be found, e.g., in the thesis [5]. Still, a commonly accepted and implementable solution has yet to be developed (e.g., the Blue Ribbon Task Force report [6] on UAS mitigation at airports unfortunately offers ”disconcerting” [7] conclusions). The governments are not ready to deploy the offered solutions primarily because technologies for multi-aviation airspace must first go through the cycle of validation and certification. Furthermore, the actual handling of the drone intruder is up to the police, who would appreciate information sharing both with ATM and UTM (or the future combined system). It is also necessary to correlate the detected drone with what UTM knows, in order to distinguish between a lost drone of a good citizen and a malicious intent to fly unnoticed.

Impact of deep convection and thunderstorms is also subject to ongoing research, e.g. Steiner et al. [8], [9] and Song et al. [10] investigated its implication both on the en-route

This research is supported by the SESAR Joint Undertaking under the European Union's Horizon 2020 research and innovation programme under grant agreement No 783287. It is also partially supported by Transportstyrelsen, Trafikverket and in-kind participation of LFV.

flow management and for terminal area applications. Klein et al. [11] used a high-level airport model to quantify the impact of weather forecast uncertainty on delay costs. Steiner et al. [12] discuss the crucial effect of accurate forecasts of high-impact winter weather for efficient management of airport and airline capacity and highlight the need of data sharing and integrated decision making between stakeholders. Recent works [13], [14] confirmed the relevance and emphasized the importance of quantification and analysis of the weather impact on airport operation. In addition, we refer the reader to publications by TBO-MET project [15] for a comprehensive account of weather impact on ATM.

Unlike the drone threats, avoiding hazardous weather has a long history in ATM. In particular, the idea of our morphing STARs is inspired by the enroute flexible flow corridors [16] that change shape as the weather obstacles move. However, the TMA setting and the solution techniques in this paper are completely different from [16]. Our solution is also completely different from the ones in [17], [18], [19], [20], [21], [22] in which the traffic is merged to a point, representing the runway, or to a circle around the runway. In this paper, the runway is a directed segment, and part of the challenge is to align the arrival flow with the runway.

II. Modeling and problem formulation

Following the setup in [1], the TMA is modelled by a rectangle (Fig. 1). A square grid V of points is laid down inside the rectangle, and every grid point is connected to its 8 immediate neighbors North, East, South, West, NE, SE, SW, and NW (these connections may be used by the STAR to link the TMA entry points to the runway). The grid points V, together with the connections (denoted by E) form the graph $G = (V, E)$ which we will search for the paths. The graph is *bidirected*: for any two vertices $i, j \in V$, both directed edges ij and ji belong to E. (We use the notation ij and i, j for the edge interchangeably.)

The runway is in the middle of the square, with the landing direction pointing South (RWY 19L). The airspace has four entry points: one on each side of the TMA. Such modeling is done only for simplicity, as our techniques work with arbitrarily shaped airspace (e.g., one way of handling an arbitrary-shape airspace may be to "carve it out" from the square by declaring as obstacles the parts of the square which do not belong to the airspace shape), any number of entry points (without restriction on their locations) and runway located anywhere. Basically, the model represents a generic TMA with entries from different directions and a runway somewhere inside.

In addition to the TMA with its entry points and the runway, the input to our problem consists of a set of moving obstacles: for any time step $t \in [1, \dots, T]$ within a given planning horizon T, we are given a set of polygons that affect the routes. In a pre-processing step, for every t we determine the subset $O_t \subseteq E$ of edges of G that overlap with the obstacles.

Weights of the edges are the last input. We use ℓ_{ij} to denote the weight assigned to edge $ij \in E$. In the weather avoidance application the weights indicate the severity of the convective weather. For the drone deconfliction the weights of edges outside the obstacles are just the lengths of the edges, while the edges inside the obstacles get infinite weight: travel through the obstacle is forbidden. Note the important difference—both algorithmic and conceptual—between weather avoidance and drone deconfliction: while in the former application a feasible solution always exists, in the latter, the obstacles are hard constraints that may not be intersected by the paths, which means that a problem instance may be infeasible.

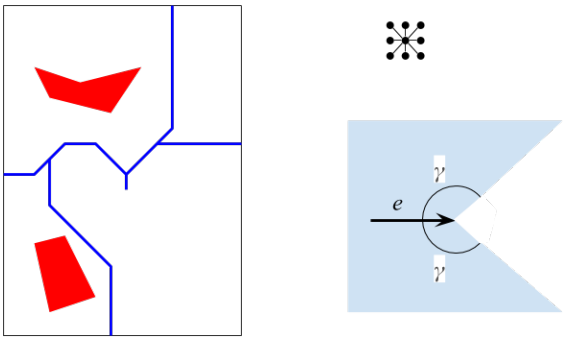


Fig. 1: Left: The STAR (blue) avoiding the obstacles (red). Right: A grid vertex is connected to the 8 neighbors. The turn constraint means that if edge e is used by a STAR, then no other edge in the blue region Γ_e (i.e,, an edge forming an angle less then γ with e) can be used in the STAR.

As the output of our problem, we seek a sequence $(\mathcal{T}_1, \dots, \mathcal{T}_\mathcal{T})$ of arrival trees (STARs) in G, that merge the entries to the runway. That is, for every $t \in [1, \dots, T]$, the tree $\mathcal{T}_t$ should have the entries as leaves and the runway as the root (contrary to the common convention, we assume that the edges of the arborescence are directed from leaves to root). The STARs should be disjoint from the obstacles – i.e., identifying a tree with its edges, we require $\mathcal{T}_t \cap O_t = \emptyset$. Operational feasibility of the STARs is ensured by requiring that they satisfy the requirements outlined in [1]: at most two routes merging at a point (translating to the requirement that every vertex of the tree must have in-degree less than or equal to 2), and the turn constraints (no sharp change of direction at any node). Last but not least, we add the new requirement that the STARs should change "slowly"; we model it by requiring that $\mathcal{T}_{t+1}$ and $\mathcal{T}_t$ should differ by at most U edges, where U is a parameter given in the input: $|\mathcal{T}_{t+1} \Delta \mathcal{T}_t| \leq U$ (where Δ denotes the symmetric difference).

Similarly to [1], we consider two objectives for any tree: minimizing the total weight of the entry-to-runway paths in the tree and the total weight of all edges in the tree (the two objectives are different because some edges are used by paths from more than one entry; see [1] for an in-depth study); we call the two objective functions the *path length* and the *tree weight* respectively. The objective function in our IP is the sum, over all timesteps, of a linear combination of the path

length and tree weight (see Equation (11) below).

III. The integer program

We formulate our problem as an integer program (IP). Our IP is a modification and extension of the IP from [1] for finding one, static STAR: we add the time dimension to allow the STAR to morph, and add constraints that assure consistency between the consecutive trees.

Variables: Our main binary decision variables $x_{e,t}$ indicate whether the edge $e \in E$ participates in the arrival tree at time $t \in [1,\dots,T]$ (i.e., whether $e \in \mathcal{T}_t$). We also have integer flow variables $f_{e,t}$ that give the *flow* on edge $e=(i,j)$ (i.e., the flow from i to j) at time t; the flow is the number of entry–runway paths that go through the edge, or in other words, the number of entry points that connect to the runway using the edge.

Tree constraints: The following constraints ensure that at any time t, the edges with positive decision variables ($x_{e,t}=1$) form a tree that connects the entry points to the runway. We use $\mathcal{P}$ to denote the set of the entry points and r to denote the runway. Both the entry points and the runway are assumed to be vertices of G, i.e., $(\mathcal{P} \cup r) \subset V$.

$$\sum_{k:(k,i)\in E} f_{ki,t} - \sum_{j:(i,j)\in E} f_{ij,t} = \begin{cases} |\mathcal{P}| & i = r \\ -1 & i \in \mathcal{P} \\ 0 & i \in V \setminus \{\mathcal{P} \cup r\} \end{cases} \quad \forall t \in [1,\dots,T] \tag{1}$$

$$x_{e,t} \geq \frac{f_{e,t}}{Q} \quad \forall e \in E, \forall t \in [1,\dots,T] \tag{2}$$

Here Q is a large number (e.g., $Q=|\mathcal{P}|$). Equation (1) ensures that a flow of $|\mathcal{P}|$ reaches the runway r, a flow of 1 leaves every entry point, and in all other vertices of the graph the flow is conserved. Equation (2) enforces edges with a positive flow to participate in the STAR, and since $x_{e,t}$ are part of the objective function (see Equation (11) below), edges with 0 flow will have $x_{e,t}=0$.

Degree constraints: Equations (1)-(2) are standard IP constraints for a MinCostFlow Steiner tree formulation. The following constraints ensure that the requirements put on the degrees of the tree nodes are satisfied:

$$\sum_{k:(k,i)\in E} x_{ki,t} \leq 2 \quad \forall i \in V \setminus \{\mathcal{P} \cup r\}, \forall t \in [1,\dots,T] \tag{3}$$

$$\sum_{j:(i,j)\in E} x_{ij,t} \leq 1 \quad \forall i \in V \setminus \{\mathcal{P} \cup r\}, \forall t \in [1,\dots,T] \tag{4}$$

$$\sum_{k:(k,r)\in E} x_{kr,t} = 1 \quad \forall t \in [1,\dots,T] \tag{5}$$

$$\sum_{j:(i,j)\in E} x_{ij,t} = 1 \quad \forall i \in \mathcal{P}, \forall t \in [1,\dots,T] \tag{6}$$

Equation (5) ensures that the runway r has one in-going edge, Equation (6) makes sure that each entry point has one outgoing edge. The maximum indegree of 2 for all other vertices is given by Equation (3), and the maximum outdegree of 1 – by Equation (4).

Turn angle constraints: Next, we ensure that for each edge $e=(i,j)$ used in any arrival tree, all outgoing edges at j form an angle of at least γ with e (see Fig. 1, right). Let Γ_e be the set of all outgoing edges from j that form an angle $\leq \gamma$ with e, i.e., $\Gamma_e = \{(j,k) : \angle ijk \leq \gamma, (j,k) \in E\}$, and let $c_e = |\Gamma_e|$. We add the following constraint:

$$c_e x_{e,t} + \sum_{e' \in \Gamma_e} x_{e',t} \leq c_e \quad \forall e \in E, \forall t \in [1,\dots,T] \tag{7}$$

By Equation (7) we can either use edge $x_{e,t}$ (which sets the left-hand side to c_e, the upper bound provided by the right-hand side, and prohibits the use of any edge in Γ_e), or we may use any subset of the edges in Γ_e.

Consistency between trees of consecutive timesteps: Recall that we measure consistency in terms of the number of different edges used in the trees, and we want to assure that the trees do not change too much from a timestep to the next timestep. To implement this we define a binary variable $y_{e,t}$ that equals $|x_{e,t} - x_{e,t+1}|$ (Equations (8)-(9)), and then limit the number of differing edges in Equation (10) by parameter U:

$$y_{e,t} \leq x_{e,t} - x_{e,t+1} \quad \forall e \in E, \forall t \in [1,\dots,T-1] \tag{8}$$

$$y_{e,t} \leq x_{e,t+1} - x_{e,t} \quad \forall e \in E, \forall t \in [1,\dots,T-1] \tag{9}$$

$$\sum_{e\in E} y_{e,t} \leq U \quad \forall t \in [1,\dots,T-1] \tag{10}$$

Objective function: Since the flow variable $f_{e,t}$ represents the total number of entry–runway paths that go through edge e at time t, the path length, i.e., the total length of all entry-to-runway paths in the tree $\mathcal{T}_t$ is $\sum_{e\in E} \ell_{e,t} f_{e,t}$. The tree weight is $\sum_{e\in E} \ell_{e,t} x_{e,t}$. Our objective function is a linear combination of the path length and tree weight, summed over the planning horizon:

$$(1-\beta) \sum_{t\in[1,\dots,T]} \sum_{e\in E} \ell_{e,t} f_{e,t} \quad + \quad \beta \sum_{t\in[1,\dots,T]} \sum_{e\in E} \ell_{e,t} x_{e,t} \tag{11}$$

We used $\beta = .1$, prioritizing path length (to make sure $x_{e,t}=0$ for edges with 0 flow, β must be strictly greater than 0).

Note that our IP is run only once, altogether for all $t \in [1,\dots,T]$ (i.e., we do not run a separate IP for each timestep t).

IV. Experimental results

This section presents application of the IP from Section III to two types of events in Stockholm Arlanda TMA: drone intrusion and inclement weather. We covered the TMA with a 66nmi x 90nmi rectangle and laid out an 11x15 grid G with distance 6nmi between the gridpoints (i.e., a grid pixel has side length of 6nmi). The grid edges outside the TMA were declared as permanent obstacles, to make sure the STARs stay within the TMA.

For the drone scenario, we generate the initial point p where the drone was detected at time $t=1$ so that the projected

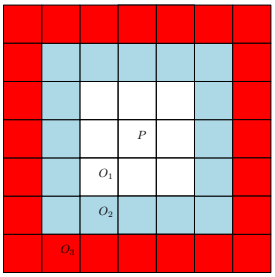


Fig. 2: If a drone is detected in the pixel P, during one timestep it will stay in P or a neighboring pixel. The growth of the drone obstacle at timesteps 2 and 3 is shown with the lightblue and red.

drone-impacted area does not reach the runway during the time horizon (otherwise no feasible solution can be found). Since drones in airport vicinity are often simply spotted by naked eye, without the ability to give an accurate estimate of its position, we add an uncertainty region around the intruder sighting. Specifically, let P be the pixel containing p. We assume that the error in determining the drone position is 3nmi (half the pixel size); then at time $t = 1$ when the drone is detected, it can be anywhere within P. We declare the 3x3 square of pixels around P as the obstacles O_1 (Fig. 2). We set the time unit (the interval between t and $t+1$, or equivalently, the time between STAR updates) to 6nmi/v where v is (an upper bound on) the speed of the drone. Put differently, the time unit is chosen so that the drone may cover maximum 6nmi (the side of a pixel) in one timestep. This way, by the end of the first time unit (i.e., by the time $t = 2$), the drone is guaranteed to stay within O_1.

The closed airspace (i.e., the uncertainty region within which the drone may potentially be found) expands with time: the obstacles O_{t+1} are obtained by adding to O_t the belt of pixels that share boundary with O_t (refer to Fig. 2). This is a safe growth: by induction on t, the drone cannot get outside O_t by the end of t-th time interval. We are thus always one step ahead of the drone, conservatively closing parts of the airspace before the drone can possibly reach them.

Overall, during an intrusion incident, the authorities would have to supply our framework with estimations or best guesses of two drone characteristics – the maximum speed v and the maximum lifetime ℓ. The latter may be based on the battery life, time to detect the drone and mitigate the intrusion, etc. (see [5, Section 3.2] for a discussion). We translate ℓ into the time horizon $T = \lceil \ell/(6\text{nmi}/v) \rceil$ expressed as the number of timesteps. Our IP then outputs the sequence $(\mathcal{T}_1, \ldots, \mathcal{T}_\mathcal{T})$ of the necessary number T of STARs, updated with the corresponding frequency – the duration of the time unit. If an update of the drone location arrives (e.g., if the drone is tracked), the obstacles can be recomputed accordingly. We model the situation when the drone is seen only once.

For our experiments we chose 2 scenarios: a very good drone (representing the most impactful case) and an average drone. If a detection system or another source gives an indication of what kind of drone the airport is dealing with, there is no need to hedge against the worst case. We chose the values for v and ℓ based on Google answer boxes for "drone speed" and "how long a drone flies", which inform that "Drones from leaders in the drone industry have a maximum speed between 50-70mph" [23] and that "A drone can fly for about 15-30 minutes on average for most of the popular drones available" [24]. We therefore took $v = 72$nmi/h and $\ell = 30$min for the advanced drone scenario. Note that speed and endurance are conflicting objectives, so the high speed and long flight time may not necessarily co-exist in a single drone model, meaning that ours is a conservative estimate of the intruder performance (exploring the full Pareto frontier of drone speed/fligthtime is outside our scope). The chosen v implies that the drone crosses the 6nmi pixel in 5min, which becomes the time unit for our STAR update; the 30min horizon thus consists of $T = 6$ timesteps. For the "average" drone we halved the speed (leading to the timestep of 10min) and assumed lifespan $\ell = 20$min, translating to the need to commute just two trees ($T = 2$).

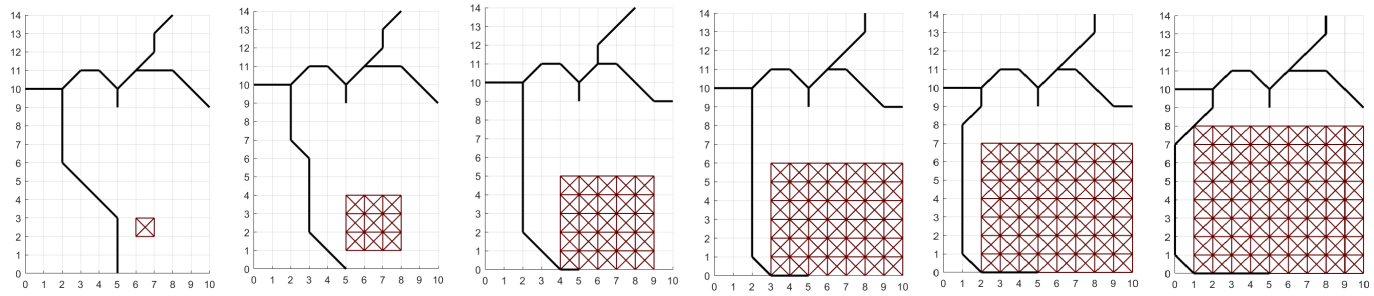
Finally, the STAR "smoothness" parameter U was set to 10. Because a STAR consists of approximately 50 edges, this means that at most 20% of the tree is allowed to change from timestep to timestep. While in principle it allows half of the edges to change after 5 steps, such drastic overall changes were never observed in our experiments. This is likely due to our objective of minimizing the total length of all STARs. In other words, trying to stay short, the STARs "hug" the obstacles, locally avoiding them, but not changing much globally.

We solve the IP using AMPL and Gurobi optimization software installed on a very powerful Tetralith server [25], utilizing the Intel HNS2600BPB computer nodes with 32 CPU cores, 384 GiB, provided by the Swedish National Infrastructure for Computing (SNIC). One problem instance took $\sim$10 minutes to solve.

Figure 3 covers both scenarios: the first two subfigures (for $T = 1$ and $T = 2$) correspond to the slow drone scenario, and all the six trees illustrate the worst-case drone expansion scenario. The STARs computed by our algorithms avoid the obstacles, staying close to them – this is due to the objective of minimizing the length of the trees. Note that when the obstacle covers the entry point, the tree follows the TMA boundary until the path can enter the TMA – in practice, the entry may be moved there.

We now describe our experiments with the weather. Since a weather event is evolving more slowly than a drone intrusion incident, we use hourly weather updates to define the obstacles O_t at every hour t and update the STARs every hour during a time horizon of $T = 5$ hours.

We used a Convective Available Potential Energy (CAPE) metric, which is an indicator of the instability of the atmosphere and can be used to assess the potential for the development of convection, which can lead to heavy rainfall, thunderstorms and other severe weather. CAPE is the energy a parcel of air has for upward motion, measured in joules per kilogram of air (J/kg). Observed values in thunderstorm



(a) T = 1 (b) T = 2 (c) T = 3 (d) T = 4 (e) T = 5 (f) T = 6

Fig. 3: The drone-affected area is dark red. The STARs for every 5min within a 30-min time horizon are black.

environments often may exceed 1000 J/kg. Note, that our method is universal, and CAPE can be replaced with any other indicator of the convective activity, or other weather phenomena evolving in time horizon.

We took the historical weather data from the ECMWF ERA5 reanalysis dataset provided via the C3S Data Store [26]. ERA5 provides estimates for each hour of the day and the data cover the Earth on a $0.25°$ grid. July 29, 2018 was chosen as the highest values of CAPE were recorded on this day (above 1000 J/kg). We took CAPE values for the $59° - 61°N, 17° - 19°E$ grid as it covers the Stockholm Arlanda TMA area.

The weather cells are given weight based on the severity, and paths with low weight (weather exposure) were sought. Weights to all possible links are assigned based on the location of the weather cells, and we assign higher weights to the links impacted by the higher activity weather. Since the weather cells granularity is not the same as the grid granularity, we assign weights of the nearest weather point to all links in the grid cell. Also, when one link is close to more than one weather cell, the highest weight of all close-by cells is assigned. In Figure 4 the weather-impacted links are colored as follows: A CAPE value below 800 is given a weight of 1 and is not shown in the figures (we assume low impact of the connective weather associated with these values). Light yellow corresponds to a CAPE value of 800-899, yellow corresponds to a CAPE value of 900-999, orange corresponds to a CAPE value of 1000-1099, red corresponds to a CAPE value of 1100-1199 and dark red corresponds to a CAPE value above 1200. Figure 4 shows how the high activity storm cells are being avoided in most cases.

First, we set $U = 5$ in this experiment. Given such a low flexibility in the number of links the trees can be different in, and having all the trees optimized together in one program, the resulting routes are not always passing through the cells with the lowest weather activity (e.g. the route from the south entry in Figure 4 (b)).

With $U = 20$ (Figure 5), the difference between the trees is more noticeable. The resulting paths are better adjusted to the current weather conditions (e.g. for $T = 2$ in Figure 5 (b), the route from the south entry is now passing through the cells with the lowest weather activity). Another example is the rightmost STAR for $T = 6$, where in the first case ($U = 5$, Figure 4 (f)) the path from the south entry is passing through some weather-impacted regions, while in the second case ($U = 20$, Figure 5 (f)) safely avoids them.

This demonstrates the trade-off between the total path length and the consistency between the trees objectives. We present the total path lengths for the resulting trees for different values of the STAR "smoothness" parameter U in Table I and visualize the Pareto-optimal solutions in Figure 6. High values of the total path lengths are resulting from the high weights of the tree edges belonging to the weather-impaired regions.

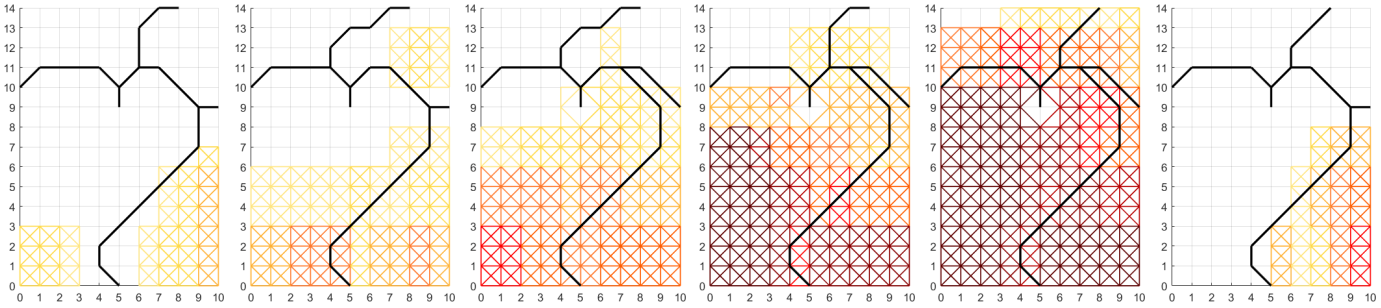
TABLE I: Total weighted path lengths for the trees with different STAR "smoothness" parameter U

U	$T = 1$	$T = 2$	$T = 3$	$T = 4$	$T = 5$	$T = 6$	Total
5	87	1026	2019	4106	9300	532	17070
10	41	878	1971	3862	9350	135	16237
20	37	582	1834	3434	9400	37	15324
30	36	537	1729	3433	9300	36	15071

V. Conclusion and Future Work

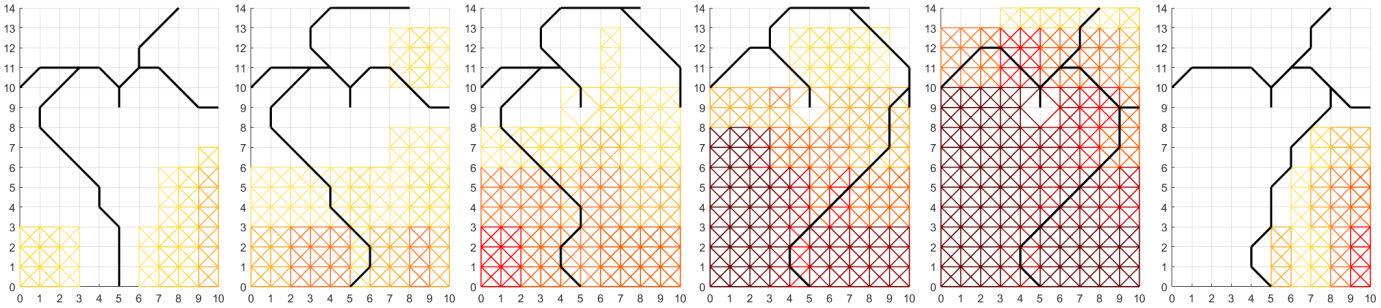
Drone interference on approach is a well recognized issue and brainstorming on its solution is ongoing (see e.g., 4 ways for drone threat reduction from The Hill [27]). This paper adds a thought on a possible mitigation strategy by pure ATM means, without involving any interception equipment. Indeed, shooting into a congested TMA, even with high-precision drone-aimed cannons would likely be an unpopular solution. On the contrary, applying soft measures, like the one proposed in this paper, may be hardly noticeable by the passengers (other than the potential delays due to rerouting) and will allow the authorities to avoid bad publicity. Future work may look more closely at the drone impact at different heights along the STAR, as the small drones would not be able to reach the altitudes at which aircraft usually arrive at the entry fix.

On the technical side, our solution entails extending an IP-based optimization framework for obstacle-avoiding STAR generation [1] to the case of moving obstacles. The extended framework was applied to two types of TMA obstacles – drones and convective weather. Our algorithms output slowly



(a) $T = 1$ (b) $T = 2$ (c) $T = 3$ (d) $T = 4$ (e) $T = 5$ (f) $T = 6$

Fig. 4: Optimized STARs with weighted links based on severity of the weather, for six different time steps T for $U = 5$. CAPE intensity is illustrated by the color of the links. The darker colors, the higher CAPE values.



(a) $T = 1$ (b) $T = 2$ (c) $T = 3$ (d) $T = 4$ (e) $T = 5$ (f) $T = 6$

Fig. 5: Optimized STARs with weighted links based on severity of the weather, for six different time steps T for $U = 20$. CAPE intensity is illustrated by the color of the links. The darker colors, the higher CAPE values.

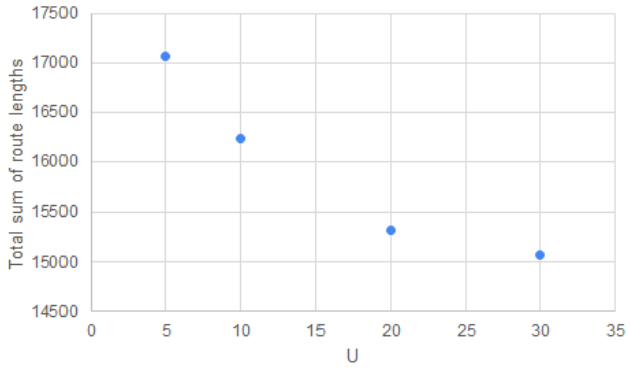


Fig. 6: Pareto optimal solutions for two objectives: STAR "smoothness" and the total route lengths over all STARs.

morphing STARs that avoid the obstacles and do not change too fast over time.

While for the proof-of-concept we modeled the drone obstacle as a growing *square*, the drone-affected area may be more accurately modeled by a growing *disk*. Our approach is directly extendable to working with disks or, in fact, arbitrarily shaped closed off areas. For instance, in presence of winds, an elongated shape of possible drone locations may be more appropriate. The main idea is that the drone obstacle is simply a set of parameters, which may be adapted as new information becomes available. Handling the multitude of such parameters may quickly become infeasible for a human controller, implying that our methods may rather be applicable at a higher level of ATS automation (e.g., in a to-be-designed more dynamic UAM airspace) where the information about the morphing paths will be easily uplinked to the FMS. At this point even the terminology "STAR" may become obsolete, as the used routes are not the standard published ones.

One important aspect ignored by our model is keeping the aircraft separated while the STAR is morphing. In [28], the algorithm from [1] was modified to ensure the separation along a *static* STAR. The modification may be potentially combined with the extension in this paper to enforce separation on the *morphing* STAR. Of course in practice, during a drone interference incident, the situation will be given extra attention from the TMA controllers who will vector the aircraft if necessary for the separation. Keeping the aircraft separated during a weather event is, on the contrary, a nominal situation.

In addition to the constraints on the STAR considered in this paper, [1] also took into account STAR–SID deconfliction by vertical separation. It is straightforward to add such separation to our constraints. We did not do it in order to focus on a minimal non-trivial problem in this paper, highlighting the novelty (avoiding moving obstacles); a detailed study of the STAR–SID interaction was done in [1]. Yet again, during the drone intrusion departing traffic might be held on the

ground, while in the weather avoidance scenario, taking SID into account may be included into the optimization framework.

A standard assumption in all research on STAR design is that the runway configuration is *specified*, i.e., the landing runway is *given* in the input. It may be interesting to lift this assumption and extend our framework by allowing to change the configuration and land (possibly, part of) the traffic on another runway. This may be particularly useful if landing all aircraft to the single, given runway is infeasible.

Given the huge impact of weather on ATM, it may be worth exploring how weather data granularity influences the arrival routes. For that, one could either take several weather data sources, or simply coarsen the given weather data by blurring it (i.e., considering it on a coarser grid). Then, for the STARs computed using the different-granularity weather, several performance indicators (PIs) may be evaluated. The PIs may be compared both among the STARs and versus the real operational flights.

The first tree $\mathcal{T}_1$ in our STAR sequence $(\mathcal{T}_1, \ldots, \mathcal{T}_\mathcal{T})$ is output by our IP. Alternatively, one could take some existing tree $\mathcal{T}_0$ as the input and require that also the first tree $\mathcal{T}_1$ does not differ too much from $\mathcal{T}_1$: $|\mathcal{T}_1 \Delta \mathcal{T}_0| \leq U$. This would be a straightforward addition to our framework.

ACKNOWLEDGMENT

We thank Helena Samsioe (Glonhe), Attila Takacs (Ericsson), Dr. Parimal Kopardekar (NASA) and Billy Josefsson (LFV) for helpful discussions. We acknowledge the anonymous reviewers for their valuable comments imoroving the presentation of the paper.

REFERENCES

[1] T. Andersson Granberg, T. Polishchuk, V. Polishchuk, and C. Schmidt, "Automatic design of aircraft arrival routes with limited turning angle," in *ATMOS 2016, August 25, 2016, Aarhus, Denmark*, vol. 54, 2016, pp. 9–1.

[2] A. Toor, "Holland's drone-hunting eagles are ready to fly," 2016, https://theverge.com/2016/9/13/12900596/netherlands-drone-hunting-eagles, last accessed 14.02.2020.

[3] Fortem Technologies, "Ultimate c-uas interceptor," 2020, https://fortemtech.com/products/dronehunter/, last accessed 14.02.2020.

[4] Rafael, "Drone dome – c-uas," https://www.rafael.co.il/worlds/air-missile-defense/c-uas-counter-unmanned-aircraft-systems/, last accessed 14.02.2020.

[5] E. Royo, "Mitigate drone intrusions in TMA," B.S. thesis, Linköping University and Technical University of Barcelona, 2019, supervisors: T. Polishchuk and V. Polishchuk.

[6] Blue Ribbon Task Force, "On uas mitigation at airports," 2019, https://dronelife.com/2019/10/07/the-report-on-drone-mitigation-at-airports-is-out-and-its-disconcerting/, last accessed 14.02.2020.

[7] M. McNabb, "The report on drone mitigation at airports is out: and it's disconcerting," 2019, https://dronelife.com/2019/10/07/the-report-on-drone-mitigation-at-airports-is-out-and-its-disconcerting/, last accessed 14.02.2020.

[8] M. Steiner, R. Bateman, D. Megenhardt, Y. Liu, M. Xu, M. Pocernich, and J. Krozel, "Translation of ensemble weather forecasts into probabilistic air traffic capacity impact," *Air Traffic Control Quarterly*, vol. 18, no. 3, pp. 229–254, 2010.

[9] M. Steiner, W. Deierling, K. Ikeda, E. Nelson, and R. Bass, "Airline and airport operations under lightning threats-safety risks, impacts, uncertainties, and how to deal with them all," in *6th AIAA Atmospheric and Space Environments Conference*, 2014, p. 2900.

[10] L. Song, D. Greenbaum, and C. Wanke, "The impact of severe weather on sector capacity," in *ATM Seminar*, 2009.

[11] A. Klein, S. Kavoussi, and R. S. Lee, "Weather forecast accuracy: Study of impact on airport capacity and estimation of avoidable costs," in *ATM Seminar*, 2009.

[12] M. Steiner, "Coping with adverse winter weather: emerging capabilities in support of airport and airline operations," *JOURNAL OF AIR TRAFFIC CONTROL*, 2015.

[13] S. Reitmann, S. Alam, and M. Schultz, "Advanced quantification of weather impact on air traffic management," in *ATM Seminar*, 2019.

[14] M. Steinheimer, C. Kern, and M. Kerschbaum, "Quantification of weather impact on air arrival management," in *ATM Seminar*, 2019.

[15] TBO-MET, "Meteorological uncertainty management for trajectory based operations," 2018, https://tbomet-h2020.com/, last accessed 14.02.2020.

[16] J. Krozel, J. S. Mitchell, A. Pääkkö, and V. Polishchuk, "Throughput/complexity tradeoffs for routing traffic in the presence of dynamic weather," in *International Conference on Research in Air Transportation*, 2010.

[17] J. Krozel, C. Lee, and J. S. Mitchell, "Turn-constrained route planning for avoiding hazardous weather," *Air Traffic Control Quarterly*, vol. 14, no. 2, pp. 159–182, 2006.

[18] D. M. Pfeil and H. Balakrishnan, "Identification of robust terminal-area routes in convective weather," *Transportation Science*, vol. 46, no. 1, pp. 56–73, 2012.

[19] D. Michalek and H. Balakrishnan, "Dynamic reconfiguration of terminal airspace during convective weather," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 4875–4881.

[20] J. Krozel, J. Prete, J. Mitchell, J. Kim, and J. Zou, "Capacity estimation for super-dense operations," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 7226.

[21] J. Prete, J. Krozel, J. Mitchell, J. Kim, and J. Zou, "Flexible, performance-based route planning for super-dense operations," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 6825.

[22] S. Choi, D. G. Mulfinger, J. E. Robinson III, and B. J. Capozzi, "Design of an optimal route structure using heuristics-based stochastic schedulers," *Journal of Aircraft*, vol. 52, no. 3, pp. 764–777, 2015.

[23] Drone Omega, "How fast do drones fly," https://www.droneomega.com/how-fast-do-drones-fly/, last accessed 14.02.2020.

[24] ——, "How long can a drone fly," https://www.droneomega.com/how-long-can-a-drone-fly/, last accessed 14.02.2020.

[25] "Tetralith server, NSC, Linköping University," https://www.nsc.liu.se/systems/tetralith/, accessed on 01/05/2020.

[26] "Copernicus Climate Change Service (C3S) Data Store, European Centre for Medium-Range Weather Forecasts (ECMWF)," https://cds.climate.copernicus.eu, last accessed 20.01.2020.

[27] Y. A. Barsoum and A. Lacher, "Four ways to reduce the threat of small drones," 2019, https://thehill.com/opinion/technology/447679-four-ways-to-reduce-the-threat-of-small-drones, last accessed 14.02.2020.

[28] J. Dahlberg, T. A. Granberg, T. Polishchuk, C. Schmidt, and L. Sedov, "Capacity-driven automatic design of dynamic aircraft arrival routes," in *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. IEEE, 2018, pp. 1–9.