Control of Aircraft in the Terminal Manoeuvring Area using Parallelised Sequential Monte Carlo

Alison Eele, Jan Maciejowski[*], Thomas Chau and Wayne Luk[†] [‡]

This paper reports on the use of a parallelised Model Predictive Control, Sequential Monte Carlo algorithm for solving the problem of conflict resolution and aircraft trajectory control in air traffic management specifically around the terminal manoeuvring area of an airport. The target problem is nonlinear, highly constrained, non-convex and uses a single decision-maker with multiple aircraft. The implementation includes a spatio-temporal wind model and rolling window simulations for realistic ongoing scenarios. The method is capable of handling arriving and departing aircraft simultaneously including some with very low fuel remaining. A novel flow field is proposed to smooth the approach trajectories for arriving aircraft and all trajectories are planned in three dimensions. Massive parallelisation of the algorithm allows solution speeds to approach those required for real-time use.

I. Introduction

Air Traffic Management (ATM) is concerned with the routing, safety and scheduling of aircraft in regions of airspace. Currently this role is performed by Air Traffic Control (ATC) and is a very human orientated process, although steps towards autonomy have been made in recent years.[1] ATC has complete control of the aircraft through all stages of flight, from pre-flight flight plans, updates and additional instructions based on traffic flow, to landing scheduling. The majority of ATC concerns lie with the avoidance of dangerous encounters through maintenance of safe separation between aircraft. If a loss of the minimum safe separation between aircraft does not have safety repercussions (for example in mid-long term detection) the priorities of the aircraft are considered by the ATC when resolving the conflict. These priorities can include avoidance of turbulence, minimising fuel usage or reaching a desired time of arrival.

The task of ATC is further complicated by the introduction of uncertainty. This uncertainty is introduced by the effect of the wind, incomplete knowledge of the physical coefficients of the aircraft and imprecision in the execution of ATC commands. With the addition of uncertainty tasks such as conflict detection become a matter of assessing the probability of a conflict given the current state of the airspace and the uncertainty of the future positions of the aircraft. Probabilistic models can be used to estimate the distribution of the future state of the aircraft given the current state.

Analysis has been conducted on where the greatest savings can be made throughout an aircraft's flight trajectory with a network-wide drive to reduce emissions and save energy. The cruise behaviour of aircraft has long been an active area of research, but there are many savings which can be made in the climb and descent phases of flight.[2] Efficiency of aircraft arrivals and departures can be improved by increasing the number of continuous climb departures (CCDs) and continuous descent approaches (CDAs). In continuous climbs or descents the aim is to minimise the amount of time aircraft spend holding at fixed altitudes. In a congested airspace especially around the terminal manoeuvring area (TMA) accommodating for such manoeuvres places additional stress on the ATC system.

Many existing multi-aircraft trajectory optimisation methods focus purely on cruise-like conditions where trajectories can be approximated in two dimensions or movement in the third dimension is highly penalised for passenger comfort.[3–5] Some work has been done using a novel conic approximation of the TMA to extend a 2-dimensional approach.[6] Methods such as mixed integer linear programming (MILP) require a

[*] Department of Engineering University of Cambridge, Cambridge, UK (e-mail: {aje46,jmm1}@cam.ac.uk)

[†] Department of Computing, Imperial College, London, UK (email: {c.chau10,w.luk}@imperial.ac.uk)

[‡]This work was supported by EPSRC (Engineering and Physical Sciences Research Council - UK) Grant No. EP/G066477/1

linearisation of the problem and are too slow on centralised problems with large numbers of vehicles due to the number of binary variables.[7]

Works more specific to the TMA often focus primarily on runway ordering optimisation through dynamic programming techniques,[8] which can include stochastic elements to account for uncertainty within the problem.[9] The work of 10 compares direct and indirect optimisation methods for a single aircraft's landing trajectory with the eventual use of a dynamic programming technique to allow for real time operation. The work generated paths of the type Shortest and Fastest Continuous Descent Approach (SF-CDA) able to reduce commercial aircraft annoyances from noise and fuel consumption.

This paper tackles the TMA problem through the use of model predictive control and sequential Monte Carlo optimisation. In model predictive control (MPC) an (open loop) optimal control problem is solved at each time step.[11] This optimisation problem can be non-convex because of local minima and lack of smoothness primarily from nonlinear dynamics, non-convex constraints or a non-convex objective function. Such optimisation problems are challenging to solve and can require sophisticated optimisation techniques. Stochastic optimisation allows for modelling the uncertainty present in ATC whilst also being able to cope with non-convex problems.

Sequential Monte Carlo (SMC) optimisation is better known for its use in 'particle filtering' (for model estimation[12]). It was observed that MPC optimisation shares similar features to that of model estimation and through this connection SMC was applied to MPC optimisation by 13. The work of 13–15 include examples of SMC applied to MPC of multiple vehicles flying in two and three dimensions under the predicted effects of wind and additional uncertainty. These works also observed that the time required to solve such problems were currently prohibitive despite the positive features of the solutions and indicated that potential time savings could be achieved through parallelisation.

Parallelisation has become a popular topic of research due to improvements in the accessibility and availability of both software and hardware solutions. Graphics Processing Units (GPUs) are one of the key technologies to emerge into the mainstream. Originally developed as dedicated devices for real-time graphics rendering they have entered into scientific computing circles by offering and facilitating exploitation of their many multi-core processors. Monte Carlo methods, when implemented on GPUs, have a proven record of significant speed-up for statistical, chemical and economic applications.[16–19]

Previous work by the authors has also demonstrated a 98% computational speed up for implementing SMC on a GPUs with cruising aircraft avoidance based applications.[20] The speed-up was of a great enough magnitude to allow the problems to be solved in real time for up to 20 vehicles in dense scenarios with a nonlinear model and non-convex constraints. This previous work considered only Gaussian white noise as disturbance on the controls of the aircraft, did not include a wind model and focused on scenarios of aircraft avoidance in cruise-like conditions.

This paper considers the more complex task of optimising the controls of aircraft approaching and departing from an airport in the terminal manoeuvring area (TMA). These aircraft are now subject to additional constraints; multi priority objective functions; a spatio-temporal wind model; and rolling window simulations.

The paper is organised as follows: Section II and Section III review the SMC optimisation method whilst discussing the customisation for the considered application; Section IV explains the parallelisation of the algorithm; Section V presents the results; Finally, Section VI presents conclusions.

II. Model Predictive Control

Model Predictive Control (MPC), also known as Receding Horizon Control, is an optimisation-based control strategy. A constrained, finite-horizon, optimal control problem is solved online at each iteration based on the available state/measurements. This solution yields the controls for the duration of a finite horizon. The first control move of the optimal sequence is then applied to the aircraft. The new states are measured and the optimisation loop begins again. For the air traffic control problem this forms a closed loop control system, as depicted in Figure 1. Many alternative forms of optimisation have been used in the first block of the diagram and often, in simpler systems, linear or quadratic programming can be applied. In this paper Sequential Monte Carlo optimisation is used as the method for solving for the applied control sequence. The remainder of this paper will deal with operations occurring in the SMC optimisation section of the update cycle.

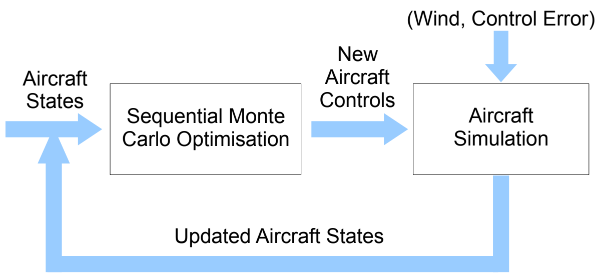

Figure 1. Graphical Representation of MPC Update Step

III. Sequential Monte Carlo

The underlying concept of SMC is to approximate a sequence of distributions of interest as a collection of L discrete masses of the variables (more commonly referred to as particles). These particles are weighted by a collection of weights to reflect the shape of the distribution. As the distribution to be approximated can vary with time the weights and particles are propagated iteratively using a sequential importance sampling and resampling mechanism. This sampling and resampling mechanism uses the particles of iteration $J-1$ to obtain new particles at iteration J. In this way a population of particles is iterated upon until a final sample is drawn from the population to act as an estimator of the maximisers.

Algorithm 1 Sequential Monte Carlo

```
1: $J \leftarrow 0$
2: Define a monotonically increasing SampleSchedule of length $J_{\max}$
3: Clone all aircrafts' current states to give each particle its own copy.
4: Set the weights of all aircraft to $1/L$ where $L$ is the number of particles.
5: For each particle and aircraft in the particle randomly generate controls over the entire horizon $H$
6: while $J \leq J_{\max}$ do
7:     $j \leftarrow 0$
8:     for each particle $l$ do
9:         while $j \leq$ SampleSchedule(J) do
10:            Sample disturbance realisations for all agents and all time steps to horizon $H$.
11:            Simulate trajectories for all agents till horizon $H$.
12:            if an aircraft $i$ fails constraints then
13:                aircraft's $i$'s weight set to 0.
14:            end if
15:            Calculate each aircraft's cost.
16:            Scale each aircraft's weightings by their cost.
17:            $j \leftarrow j+1$
18:        end while
19:    end for
20:    $J \leftarrow J+1$
21:    if $J \leq J_{\max}$ then
22:        Resample all particles
23:        Peturb all aircrafts" controls with Gaussian white noise.
24:        Set the weights of all agents to $1/L$ where $L$ is the number of particles.
25:    end if
26: end while
27: Draw final sample from particle population.
```

Algorithm 1 summarises the implementation of SMC. Each step of the algorithm shall now be dealt with in further depth with specific reference to the application of multiple aircraft trajectory planning with conflict avoidance.

III.A. Particles

A particle represents a single instance of the problem with the data of all associated agents. The number of particles for optimisation is a design variable dependent on the complexity of the problem to be optimised. The more complex the problem, the larger the number of particles may need to be to adequately characterise the search space. Inside each particle there is:

- An initial state for each of the N individual aircraft. These initial states for aircraft are the same across all particles and are provided to the optimisation at the beginning of the algorithm.
- The control inputs for every step of the MPC horizon H for every aircraft. In this paper's formulation there are 3 controls for each aircraft (thrust, bank angle and pitch angle) totalling $N * H * 3$ control inputs. These control inputs are different for every particle.
- A separate weighting for each of the N individual aircraft in the scenario, used for resampling.

The controls inside particles are initialised as random samples from a uniform distribution for each control between the maximum and minimum value. After this initialisation at line 5 the controls are then updated following resampling and perturbation in lines 22- 23. The perturbation moves particles locally in the search space, whilst resampling causes particles to cluster around areas of the search space with positive attributes as determined by the objective function. The weights of all aircraft are initialised as $1/L$ where L is the number of particles and these are updated in line 16.

III.B. Trajectory Planning and Disturbance Realisations

In lines 10-11 the future trajectory of each aircraft in each particle is simulated given the initial state and controls (from the particle's data) and the disturbance samples. The discretised aircraft dynamics model is a standard 6 DOF model with 6 states and 3 inputs

$$x_i(k+1) = x_i(k) + \delta t(v_{s,i}(k)\cos(\chi_i(k))\cos(\gamma_i(k))) + w_x(k)\delta t \tag{1a}$$
$$y_i(k+1) = y_i(k) + \delta t(v_{s,i}(k)\sin(\chi_i(k))\cos(\gamma_i(k))) + w_y(k)\delta t \tag{1b}$$
$$z_i(k+1) = z_i(k) + \delta t(v_{s,i}(k)\sin(\gamma_i(k))) \tag{1c}$$
$$v_{s,i}(k+1) = v_{s,i}(k) + \delta t(\frac{T_i(k) - D_i(k)}{m_i(k)} - g\sin(\gamma_i(k))) \tag{1d}$$
$$\chi_i(k+1) = \chi_i(k) + \delta t(\frac{L_i(k)\sin(\phi_i(k))}{m_i(k)v_{s,i}(k)}) \tag{1e}$$
$$m_i(k+1) = m_i(k) - \delta t(\eta T_i(k)) \tag{1f}$$

where: δt is the step length; x, y and z are the Cartesian coordinates of the aircraft (z acting as the altitude of the aircraft); m is the total mass of the aircraft; v_s is the true airspeed; χ is the heading angle; γ the climb angle and ϕ the bank angle. The subscript i denotes the i^{th} aircraft. The control variables are ϕ (bank), T (thrust) and γ (climb). Lift L and drag D are calculated using the standard aerodynamic relations. The fuel usage is controlled by the constant η.

Each aircraft is given an initial state $X_{0,i} = (x_{0,i}, y_{0,i}, z_{0,i}, \chi_{0,i}, v_{s,0,i}, m_{0,i})$. Departing aircraft are modelled from an initial altitude of 400m, where the aircraft is assumed to have taken off from the runway and completed its initial climb. A departing aircraft is considered as having left the TMA once it has reached a set distance D_{TMA} from the airport at which point it is assumed to enter a neighbouring control zone and is handled by alternative ATC.

Arrival aircraft are allowed the flexibility to enter the problem at any point along the TMA's boundary. Arrival aircraft are deemed to have 'landed' or passed from our control once they have satisfied the following constraints:

$$\sqrt{x_i(k)^2 + y_i(k)^2} \leq P_{\text{runway}} \tag{2a}$$
$$\beta_i(x_i(k), y_i(k), z_i(k)) \leq P_\beta \tag{2b}$$
$$|\arctan(y_i(k)/x_i(k))| \leq P_\chi \tag{2c}$$
$$|\chi_i(k) - 180| \leq P_\chi \tag{2d}$$
$$v_{s,i}(k) \leq P_{v_s} \tag{2e}$$

The first 3 of these constraints define the landing sector, an arc with radius P_{runway} horizontal angle P_χ and vertical angle P_β where β_i is described later in Equation 12. Constraint 2d ensures that the aircraft is pointing towards the runway assuming an East to West landing without significant crosswind. Finally constraint 2e limits the airspeed (P_{v_s})at which the aircraft can enter the landing sector for a successful landing. These are shown pictorially in Figure 2.

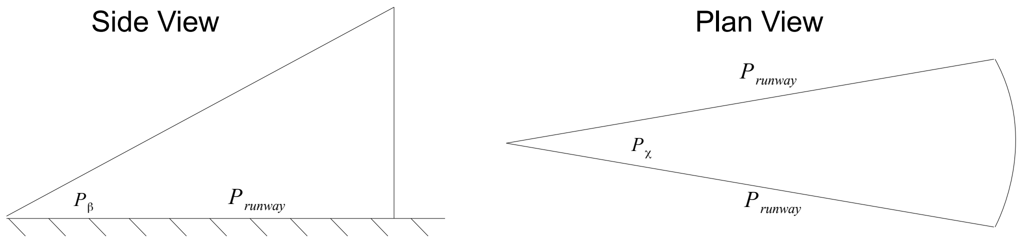

Figure 2. Landing Envelope Shape and Size

The disturbance realisations added to the system are the primary method of simulating uncertainty within the system. This uncertainty can come from wind, sensor noise, controller noise and human factors. Within the inner loop of optimisation (lines 9-18) each aircraft within a particle will have drawn SampleSchedule(J) disturbance realisations and planned the same number of trajectories. This inner loop serves to simulate different disturbance scenarios on an aircraft to determine if the controls are valid with respect to constraints, and their fitness with respect to the objective function. This information is stored in the aircraft's weight within the particle as described in the next two subsections. In applications with no uncertainty there would be no need for repeated planning and simulation of the aircraft and the inner loop would only be executed once. In this paper we have restricted the disturbances to being wind and this is discussed in greater depth in Section III.I.

III.C. Constraint Handling

There are two types of constraint handled by line 12. The first are unary constraints which deal only with one aircraft at a time. Examples of these include flight envelope constraints and the minimum aircraft mass constraint. The flight envelope constraints provide upper and lower bounds for both the state and control variables:

$$z_{\min} \leq z(t) \leq z_{\max} \tag{3a}$$
$$-\gamma_{\max} \leq \gamma(t) \leq \gamma_{\max} \tag{3b}$$
$$-\phi_{\max} < \phi(t) < \phi_{\max} \tag{3c}$$
$$T_{\min} \leq T(t) \leq T_{\max} \tag{3d}$$
$$v_{\min} \leq v_s(t) \leq v_{\max}. \tag{3e}$$

The minimum aircraft mass constraint enforces that the total aircraft mass must always remain above the mass of the aircraft alone without fuel:

$$m(t) \geq m_{\text{Aircraft}}. \tag{4}$$

This acts on a constraint on the total fuel use across the entire modelled flight. Further unary constraints can be added. For example to create exclusion zones where aircraft are not allowed to fly, or must fly above a certain altitude. These are not included in the existing model but can be implemented in future work.

The second type of constraint are the binary constraints. These hold between two aircraft in the same particle, such as in conflict avoidance. In this paper the protection zone around each aircraft is modelled as a cylinder with horizontal radius P_r and altitude separation of P_h. Two aircraft i and j avoid each other if they satisfy the following constraint for every time step in the MPC horizon:

$$\begin{aligned}(x_i(k) - x_j(k))^2 + (y_i(k) - y_j(k))^2 \geq (2P_r)^2 \\ \vee |z_i(k) - z_j(k)| \geq 2P_h \\ \forall k, \forall i, j \in \{1, ..., N\} : i \neq j.\end{aligned} \tag{5}$$

If an aircraft fails any of its unary constraints then its weight stored in the particle is set to 0. If a pair of aircraft fail a binary constraint then both aircraft have their weights set to 0. Any aircraft with a weighting of 0 will not be resampled in the resampling phase of the algorithm described in Subsection III.F.

The constraint handling method described in this section is a departure from the the generic formulation for a SMC optimisation. In the generic formulation a weight is linked to a single particle which would contain the entire simulation of all N aircraft. In that case if a single aircraft fails any constraint the entire simulation would be weighted as 0 and all controls from that particle discarded when the particles are resampled in the next iteration. Conversely, by using our implementation, aircraft are weighted individually inside a particle. If a single aircraft fails a constraint the non-failing aircraft weights would still be non-zero. This would allow their controls to continue to the next iteration. The application under consideration is a highly constrained situation and this alteration gives greater flexibility in keeping valid control solutions for aircraft which would otherwise have been discarded. This in turn allows a smaller number of particles, as the distribution they are required to model is less complex than in the case of a multi-aircraft weighted particle.

III.D. Particle Costing

The SMC method requires a non-negative function as the objective function J_T which is to be maximised. This objective function is made up of multiple elements and is different for arrival and departure aircraft due to their distinct priorities. However each element of the objective functions for both types of aircraft takes a basic form where we have negated the original minimisation function then normalised by adding the supremum of the function and dividing by the difference between the supremum and the infimum of the function. The supremum and infimum are readily available via geometric arguments as will be explained in greater depth case by case.

III.D.1. Departure:

Departing aircraft are given a desired target altitude Z_i, a desired airspeed $v_{s,D}$ and bearing $\theta_{i,F}$. These are used to evaluate the aircraft's progress in the objective function. However it is not necessary for the aircraft to have reached these before leaving the TMA as it can take a while to reach cruise altitude and traffic in the TMA could make reaching the exact target bearing unrealistic.

The departing aircrafts' objective function $J^D_{T,i}$ is made up of four elements: reaching the target bearing they need to achieve for onwards flight to their destination; a cost on fuel used; rewarding the aircraft for climbing towards their required cruise height and rewarding the aircraft for maintaining its desired speed.

$$J^D_{1,i}(k : k + H) = \frac{-A_i(k{:}k+H) + \sup A_i(k{:}k+H)}{\sup A_i(k{:}k+H) - \inf A_i(k{:}k+H)} \tag{6a}$$

$$J^D_{2,i}(k : k + H) = \frac{(m_i(k) - m_i(k+H)) + \delta t H T_{\max}\eta}{\delta t H T_{\max}\eta} \tag{6b}$$

$$J^D_{3,i}(k : k + H) = \frac{-B_i(k{:}k+H) + \sup B_i(k{:}k+H)}{\sup B_i(k{:}k+H) - \inf B_i(k{:}k+H)} \tag{6c}$$

$$J^D_{4,i}(k : k + H) = \frac{-C_i(k{:}k+H) + \sup C_i(k{:}k+H)}{\sup C_i(k{:}k+H) - \inf C_i(k{:}k+H)} \tag{6d}$$

where:

$$A_i(k:k+H) = \sum_{j=k}^{k+H} \frac{|\theta_i(j) - \theta_{i,F}|}{H} \tag{7a}$$

$$B_i(k:k+H) = \sum_{j=k}^{k+H} \frac{\sqrt{(z_{t_f,i} - z_i(j))^2}}{H} \tag{7b}$$

$$C_i(k:k+H) = \sum_{j=k}^{k+H} \frac{|v_{s,i}(j) - v_{s,D}|}{H} \tag{7c}$$

A is the absolute angle between the desired bearing of the aircraft leaving the airport and the actual bearing of the aircraft from the airport. B is the distance between the desired altitude for cruise and the altitude of the aircraft. C is the difference between the desired airspeed and actual airspeed of the aircraft. The sup and inf of A are approximated as the maximum deflection from the desired bearing, 180 degrees and the minimum deflection 0 degrees respectively. Similarly for B the sup is the maximum distance the aircraft can be from its target altitude given its current altitude, obtained by either climbing or descending with $\gamma_{\max}$ and $V_{\max}$. The inf is then the minimum distance the aircraft can be from its target altitude similarly using the $\gamma_{\max}$ and $V_{\max}$ until it is at the desired altitude at which point it would be 0. By using this formulation, the cost for aircraft i which is on its desired bearing for all steps of the horizon would have a cost of 1 for $J^D_{1,i}$, while an aircraft j which is 90 degrees away from its desired bearing for all steps of the horizon would have a cost of 0.5 for $J^D_{1,j}$. The cost J^D_2 is the fuel minimisation recast as a maximisation; $\delta t H T_{\max} \eta$ is the maximum amount of fuel that could be used in a step, the minimum is approximated to be 0.

The four parts of the cost J^D are all normalised and then scaled in importance by the weighting factors α_j. These weighting coefficients have been tuned empirically and have a large effect on the behaviour of aircraft. Choice of α_j affects policy e.g. fuel use, noise abatement etc. Example values of α_j used for the simulations in this paper are given in part V.

$$J^D_{T,i}(k:k+H) = \textstyle\sum_{j=1}^{4} \frac{\alpha_j J^D_{j,i}(k:k+H)}{H} \tag{8a}$$

$$0 \le J^D_{T,i}(k:k+H) \le 1 \tag{8b}$$

$$\sum_{j=1}^{4} \alpha_j = 1 \tag{8c}$$

III.D.2. Arrival

In contrast to departures, arriving aircraft have far stricter terminal constraints with a specific requirement that to land the aircraft must be: travelling towards the runway; be in line with the runway given a heading tolerance; within a certain distance of the runway; travelling below a given speed; and be within a set altitude trajectory to accomplish landing. The model does not extend to fully landing the plane and continues to assume pilot control for this aspect of the flight, similar to that of the initial take off.

Arriving aircraft will head towards the airport following an objective function made up of 3 elements. The first is a cost on the fuel used to encourage minimal fuel use; the second rewards the aircraft for following a nominal altitude descent trajectory; and the third rewards the aircraft for keeping its heading close to a nominal flow field designed to encourage the aircraft to turn smoothly before approaching the runway. Such a flow field is entirely designable based on the airport or goals of the ATC for the airport. For this work the airport flow field has been formulated as shown in Figure 3 and the cost function penalises aircraft for their heading not aligning with the flow field. The use of a nominal altitude descent trajectory rather than relying purely on the minimisation of fuel or a target altitude is justified by the finite horizon of planning which might not be long enough for an aircraft to reach the landing sector. By providing a nominal trajectory the aircraft's descent is smoother and less likely to drop in altitude sharply before having to fly close to the ground till it reaches the landing sector. The arrival problem is then optimised for the maximisation of the

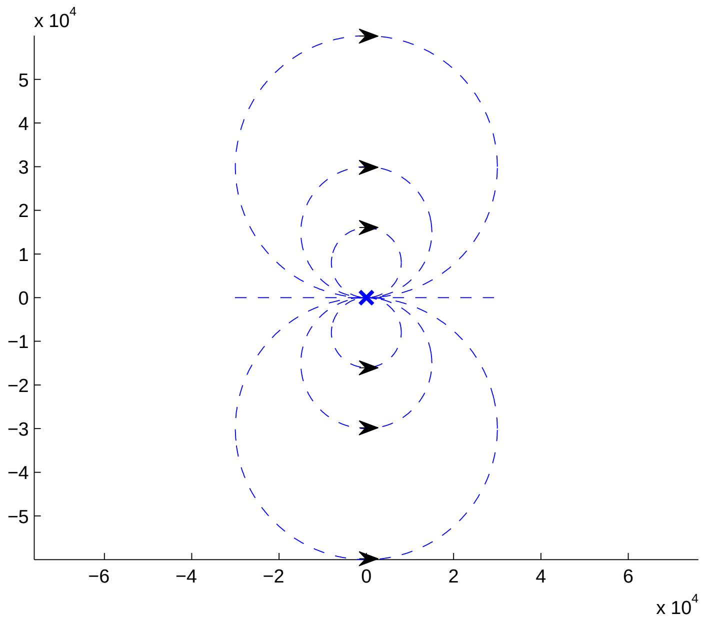

Figure 3. Example flowfield for a single runway airport with East to West landing pattern

cost J_T^A which is the sum of the weighted costs for heading offset from flowfield, fuel usage and nominal altitude descent trajectory following:

$$J_{1,i}^A(k:k+H) = \frac{-D_i(k{:}k{+}H)+\sup D_i(k{:}k{+}H)}{\sup D_i(k{:}k{+}H)-\inf D_i(k{:}k{+}H)} \tag{9a}$$

$$J_{2,i}^A(k:k+H) = \frac{(m_i(k)-m_i(k+H))+\delta t H T_{\max}\eta}{\delta t H T_{\max}\eta} \tag{9b}$$

$$J_{3,i}^A(k:k+H) = \frac{-E_i(k{:}k{+}H)+\sup E_i(k{:}k{+}H)}{\sup E_i(k{:}k{+}H)-\inf E_i(k{:}k{+}H)} \tag{9c}$$

where:

$$D_i(k:k+H) = \sum_{j=k}^{k+H} \frac{|\chi_i(j) - \hat{\chi}_{i,f}(x_i(j), y_i(j))|}{H} \tag{10a}$$

$$E_i(k:k+H) = \sum_{j=k}^{k+H} \frac{|\beta_i(x_i(j), y_i(j), z_i(j)) - \beta_{i,f}|}{H} \tag{10b}$$

and

$$\hat{\chi}_{i,f}(x_i(j), y_i(j)) = 2\arctan\left(\frac{x_i(j)}{y_i(j)}\right) + 90 \tag{11}$$

$$\beta_i(x_i(j), y_i(j), z_i(j)) = \arctan\left(\frac{z_i(j)}{180 - 2\arctan\left(\frac{x_i(j)}{y_i(j)}\right)\frac{||x_i(j), y_i(j)||}{\cos(\arctan(x_i(j)/y_i(j)))}}\right) \tag{12}$$

D is the normalised difference between the flow field's heading and the aircraft's, E is the normalised difference between the angle to the ground origin for the aircraft and the desired angle of approach. The geometric terms of β and $\hat{\chi}$ are based on the flow field shown in Figure 3 with an east west runway centred on the origin with arrival flights landing on the eastern end of the runway. This example flow field is used throughout the paper for its elegant geometric properties. It is simple to change the flow field and would result in a change of equation 12. The distance from the aircraft to the origin for E is measured not as the Euclidean distance but rather as the distance remaining on the arc of the flow field the aircraft is currently on. The point of this design choice is to encourage a constant descent angle for the aircraft throughout its travel to the runway. The Euclidean distance would have lead to variations in the angle of descent over the full path for aircraft starting at positions greater than 90 degrees to the desired landing bearing.

Like the departures, the arrivals objective function is designed to be non-negative and the total cost J_T^A and all its constituent costs are normalised between 0 and 1. The weightings on the terms of the cost function $\tilde{\alpha}$ are subject to priorities of the desired trajectory and example values are included in Section V.

$$J_{T,i}^A(k:k+H) = \sum_{j=1}^{3} \frac{\tilde{\alpha}_j J_{j,i}^A(k:k+H)}{H} \tag{13a}$$

$$0 \leq J_{T,i}^A(k:k+H) \leq 1 \tag{13b}$$

$$\sum_{j=1}^{3} \tilde{\alpha}_j = 1 \tag{13c}$$

III.E. Weight Scaling

As previously mentioned in subsection III.A each aircraft in a particle has a weight associated with it. This weight records the degree of success an aircraft has in the simulations with various noise realisations. To do this in line 16 the cost of aircraft i in particle l is multiplied by the weight of aircraft i in particle l each time the inner loop 9-18 is executed. An aircraft only needs to violate one constraint in any execution of the inner loop to have its weighting set to 0. This weighting would remain as 0 until resampling (described in the next section) removes that aircraft's proposed controls from the population.

Ideally at each iteration of the inner loop the sum across all particles L of weights associated with aircraft i would be normalised to 1. This would reduce numerical issues as the inner loop progresses. Without such a normalisation step the weight would always be decreasing

$$W_{i,l}^{j+1} = W_{i,l}^{j} J_{T,i,l}^{j} \tag{14a}$$

$$W_{i,l}^{0} = 1/L \tag{14b}$$

$$0 \leq J_{T,i,l}^{j} \leq 1 \tag{14c}$$

where W_i^j represents the weight and $J_{T,i,l}^j$ the objective function value of aircraft i in particle l at inner loop iteration j and L is the number of particles. The cost is always between 0 and 1 and the weight starts at less than 1.

III.F. Particle Resampling and Perturbation

In lines 22 - 23 the aircraft are individually resampled to generate a new population of particles. These new particles have their aircraft controls perturbed by Gaussian white noise to separate particles with similar controls across the local search space. This perturbation also allows particles to explore away from points which were in the original randomly sampled population of controls.

Resampling is done on the basis of sequential importance sampling using a method such as Kitagawa resampling.[21] The higher a weight aircraft i has in particle l compared to the weight of aircraft i in all other particles, the more likely it is to have its controls resampled into the new population.

As the aircraft are resampled separately to form the new particles no conflict avoidance guarantees can hold once resampling has taken place. This arises from the phenomena that the controls from aircraft i from particle l could now be in a new particle with the controls from aircraft j from particle m. This again arises from the departure from the generic method of one weight per particle mentioned in Section III.C, justified by the significant simplification of dimensionality of the search space and the effect on both computation time and particle populations needed. As long as the final sample from the distribution of particles is drawn correctly with regards to the binary conflict constraints the lack of conflict guarantees at the beginning of each inner loop is not problematic.

III.G. Final Sample Selection

This is the final step of the SMC optimisation (line 27) before it hands back to the MPC update process. This step determines the controls to be used by all aircraft for the next update step. In many generic SMC applications this final sample is taken as the mean of the values of the particles. In multi modal distributions this would give an inaccurate estimation of the global optimisers (consider the mean of a control where half of the population steers left around an obstacle and the other half right). An alternative statistical measure to offset this problem would be to select the mode of the distribution. However this is also unsuitable in our implementation. The presence of binary constraints across aircraft control distributions could lead to constraint violation. Therefore throughout this paper we select the best performing particle in the final iteration as our estimate of the maximiser.

To find the best performing particle the weights of all aircraft in the particle are multiplied together and the particle with the greatest overall weight is selected as the estimator of the maximiser

$$\max_{l} \prod_{i=0}^{N} W_{i,l}. \tag{15}$$

Any particle where an aircraft has failed any constraint in simulation would have a weight 0 associated with the failing aircraft and thus could not be selected as the best performing particle.

III.H. Rolling Window

To allow for longer scenarios where aircraft both enter and leave the problem, as would happen in a real airport, the whole application is considered in the form of a rolling window. The length of the window is the same length of the horizon H for the MPC and the window moves with every completion of an MPC update. Any aircraft entering the problem (either through take off or landing) does so at a pre-determined time step, though this can be relaxed to consider uncertainty on arrival into the problem. Only aircraft which are considered active in the problem have their trajectories optimised by the algorithm previously described.

As we are using MPC, aircraft may become active in the problem midway through a horizon. Their trajectories are optimised from the point they enter the problem. Similarly for arrival aircraft there is the chance for an aircraft to complete mid horizon, thus leaving the problem. During the algorithm when trajectories are planned (lines 10-11) arrival aircraft active in the problem are tested to see if they've entered the landing sector. If they have they will be noted as finished and future time steps within that horizon will not be planned for that aircraft. Departure aircraft are less constrained and not tested for early completion during the trajectory planning stage. Arrival aircraft which have met their landing constraints mid horizon are given the best possible cost, 1, for all remaining steps of the horizon as a bonus to that aircraft for landing.

Figure 4 shows an example problem where aircraft enter and leave the problem during simulation. Active aircraft are shown in light blue, finished aircraft which have left the problem are shown in dark blue and aircraft yet to enter the problem are shown in hatched lines. Aircraft will not be deactivated until they no longer appear within the bounds of the rolling window (MPC horizon), likewise they are not activated until they have appeared in the MPC horizon. The aircraft which have not finished are shown planning for an unknown time into the future as finishing can only be detected within the MPC horizon.

III.I. Wind Model

Large portions of the uncertainty about aircraft flight plans results from the inherent uncertainty in meteorological forecasts. Wind speed and direction is probably the most important meteorological factor affecting aircraft trajectories in the TMA. The wind disturbance is assumed to be made up of two components, a nominal component representing the wind forecast and a stochastic component representing the forecast errors. This paper uses a spatio-temporal wind model for the forecast errors based on the model used by Lymperopoulos and Lygeros.[22] Their model approximations were designed for wind prediction over a far larger geographical scale than that used by the TMA example. However, they appear to be suitable for our application, following experimentation.

The model of Lymperopoulos and Lygeros assumes an isotropic wind-field (invarient under rotations) with uncorrelated wind speeds in the South-North and East-West directions. Vertical wind is neglected in the model however correlation between wind at different altitudes is considered. The wind field is generated from a random field where each point is Gaussian with zero mean and a covariance matrix $R(t, P, t', P')$

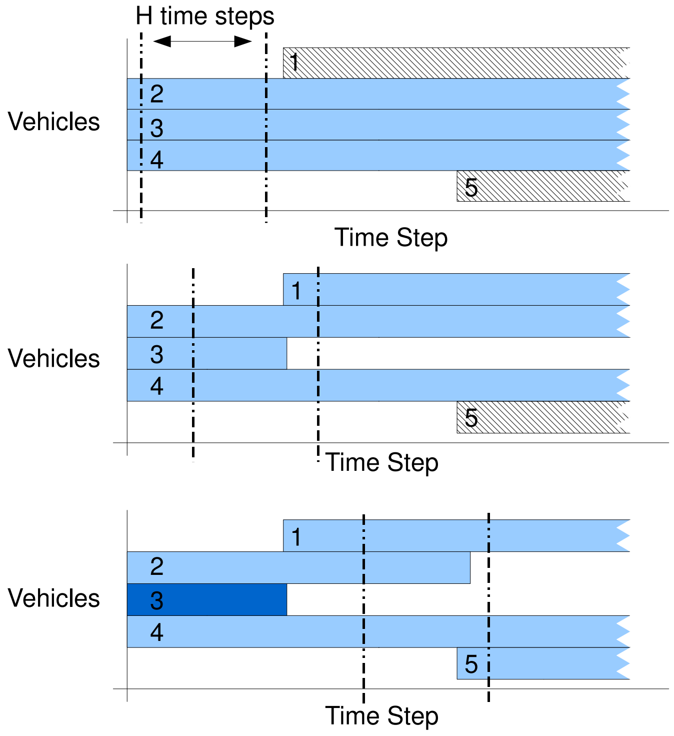

Figure 4. Graphical representation of active aircraft (light blue), completed aircraft (dark blue) and aircraft which haven't joined the problem (hatched) for rolling window simulation at three different points in simulation

where t and t' are the points in time and P and P' are the points in three dimensional space. Through these assumptions and by using an exponential function the correlation can be approximated as:

$$\rho_{xy}(t,P,t',P') = \sigma(z)\sigma(z')\text{exp}(-\lambda|t-t'|)\text{exp}\left(-\beta\left|\begin{matrix} x-x' \\ y-y' \end{matrix}\right|\right)\text{exp}(-\gamma|z-z'|). \quad (16)$$

Where $\sigma(z)$ is the standard deviation of the wind error in m/s at altitude z. The figures used for λ, β and γ used by Lygeros and Glover[23] were $\lambda = 6 * 10^{-6}m^{-1}$, $\beta = 1.6 * 10^{-6}m^{-1}$ and $\gamma = 1.5 * 10^{-5}m^{-1}$ along with $\sigma(z)$ are based on the data reported in 24.

To use the correlation function above Lymperopoulos and Lygeros proposed the following method for generating wind realisations. The wind field is divided into a grid with N_x points in the South-North direction, N_y points in the East-West direction and N_z vertically. For each point in the three dimensional grid 2 random numbers are generated from a zero mean Gaussian distribution. The first of these random numbers in each pair relates to the South-North wind error and the second to the East-West wind error, these numbers are stored in two vectors $W_X(k)$ and $W_Y(k)$ at time step k.

The covariance matrix $\hat{R} \in \mathbb{R}^{N_X N_Y N_Z \times N_X N_Y N_Z}$ where each entry is a comparison between two grid points using equation 16 is the covariance matrix for both $W_X(k)$ and $W_Y(k)$ given the isotropic assumption. Lygeros and Lymperopoulos assume this covariance matrix to be constant in time and thus generate wind samples using the following linear Gaussian model:

$$\begin{aligned} W_X(0) &= \hat{Q}v_X(0), \quad W_X(k+1) = aW_X(k) + Qv_X(k+1), \\ W_Y(0) &= \hat{Q}v_Y(0), \quad W_Y(k+1) = aW_Y(k) + Qv_Y(k+1), \end{aligned} \quad (17)$$

where $v_X(k), v_Y(k) \in \mathbb{R}^{N_X N_Y N_Z}$ are standard independent Gaussian random variables, Q and $\hat{Q}$ are derived by Cholesky Decomposition from the covariance matrix $\hat{R}$ using

$$QQ^T = (1-a^2)\hat{R} \quad \text{and} \quad \hat{Q}\hat{Q}^T = \hat{R} \quad (18)$$

where $a = e^{-\delta t/G_t}$ with G_t a parameter of time correlation obtained from 24. The wind error for individual aircraft is then obtained by tri-linear interpolation between the grid points.

IV. Parallel Implementation

It had been previously been proposed there is a large degree of speed up that can be obtained by parallelising the SMC algorithm.[16–19] The authors previous work[20] focused on implementation on a graphics processing unit (GPU) using the CUDA programming language provided by NVIDIA and demonstrated a 98% computational improvement. CUDA is a scalable parallel programming language similar to C and C++ with the libraries and utility to design kernels to implement on a GPU. These kernels are repeatable functions representing the code that the user wishes to be executed in parallel with different data. GPUs can use thousands of threads at the same time, whereas a conventional computer will only use a few. The execution and near instantaneous switching between threads is how a GPU achieves high efficiency on parallel applications.

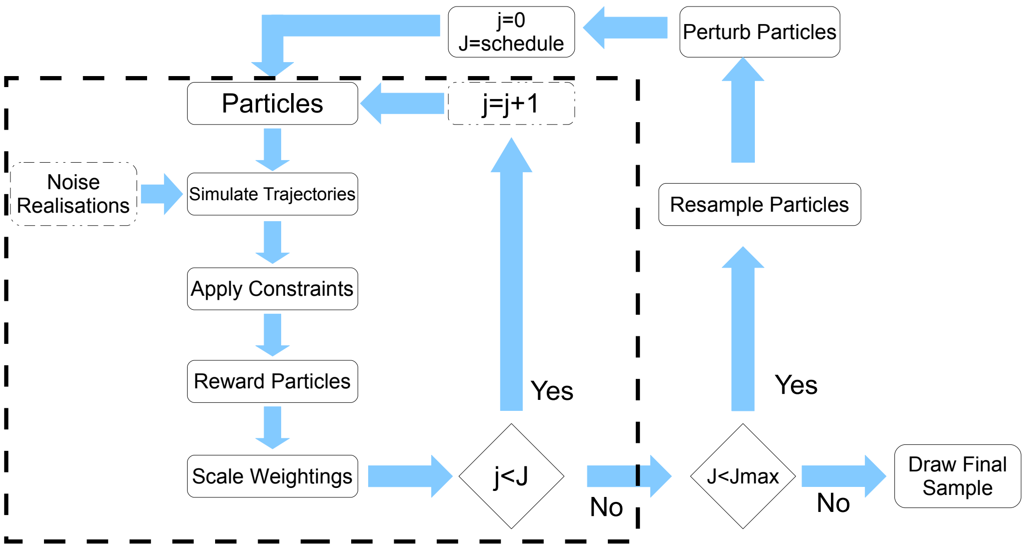

Figure 5. Graphical Representation of SMC Algorithm with Kernel Highlighted

Figure 5 shows a simplified graphical representation of the SMC algorithm focusing on the operations taking place in a single computation step of the MPC. There is also a boxed region of the flowchart demonstrating the GPU kernel. This kernel incorporates the inner loop of simulation specifically the lines 9-18 of the algorithm. This code was a bottleneck in a sequential implementation as it must be executed multiple times for each particle where the only difference between each execution is the particle's individual data. This made it a perfect candidate for parallelisation with one particle per thread, as there is minimal thread interaction and use of shared memory.

In the algorithm used there is no need for the threads to synchronise or communicate and thus the problem can be considered as 'embarrassingly parallelisable'. The remaining implementation decision was to select an appropriate random number generator for use within the kernel. In this paper's implementation we have focused on using the NVIDIA provided library CURAND, specifically the XORWOW (Xor Shift added with Weyl sequence generator[25]) which seeds each thread and maintains the state of each random generator between kernel calls. This generator was used for its superior speed compared to that of the others included in CURAND.

V. Simulations and Results

This section presents the results of simulations performed on the previously outlined algorithm and application setup. These simulations vary in complexity from single aircraft paths, used to tune the performance of the objective function coefficients to tests designed to see how far we can push the implementation with our current setup. The final section of the results discusses the computational speed of the method compared to that of a previous implementation.[20]

V.A. Simulation Setup

For all simulations within this paper the following parameters were kept constant. MPC time steps were 10 seconds in length ($\delta t = 10$) and had a horizon length of $H = 6$ time steps. δt is arguably far shorter than needed in the application where standard ATC would be looking at 30 seconds to 60 seconds updates. For simplicity aircraft have been standardised with the aerodynamic properties of an Airbus A320 obtained from BADA.[26] The TMA is considered as a circle of radius 30km where there is a single runway at the origin. Aircraft both land and depart East to West from this one runway. There is no additional runway scheduler running alongside the simulations so separation near the landing envelope is maintained by the distance separation used for conflict avoidance from Equation 5.

The algorithm used a $J_{\max} = 100$, with a schedule function of $\text{SampleSchedule}(J) = \lfloor(3 + 5e^{0.05J})\rfloor$ and $L = 10240$ particles (which have been naively arranged as 80 blocks of 128 threads for the GPU implementation). The simulations were performed on an NVIDIA Tesla C2070, which has 448 cores.

The disturbances from wind for all scenarios were simulated using the wind model outlined in Section III.I. A three dimensional grid of eight points was used to generate the wind at the outlying reaches of the TMA before trilinear interpolation found the disturbances at the aircrafts' individual locations. No prevailing wind was used in the scenarios so the aircraft are only subject to the random disturbances of the wind field itself. A denser grid of points can be used for sampling the wind speed however this slows computation down significantly with the need to generate many more random numbers within each inner loop of the GPU kernel.

V.B. Single Aircraft Trajectories

The presented results were used to tune the coefficients in the objective functions for both departure and arrival aircraft and evaluate the behaviour of arrival aircraft when exposed to the flow field. In each case for departures and arrivals 12 aircraft have been independently simulated and then plotted together on Figures 6 and 7 to give an overview on behaviour. The arrival aircraft were started at different places around the airport and simulated until they reached the same landing zone. Departing aircraft were given different desired bearings and simulated until they reached the edge of the TMA zone.

The effect of the flow field on arrival aircraft is clear in Figure 7, on aircraft approaching from the Eastern side of the runway minimal diversion is taken prior to arrival. However the vehicle approaching from due west of the runway is given a long diversion outside the TMA before re-approaching from a more suitable bearing. In reality aircraft are allowed to approach an airport from a restricted number of directions and flying directly into departing aircraft traffic would be unlikely.

From this tuning the weightings shown in Table 1 were used for landing and take off aircraft in the remainder of the scenarios in the paper. Tuning these parameters differently will have significant effects on the trajectories generated.

V.C. Standard Arrivals and Departures

The simulations presented here demonstrate the unified running of the departure and arrival aircraft trajectories for the airport. Scenarios were randomly generated such that both arrival and departure aircraft joined the simulation at fixed times evenly spread across the entire simulation interval with a variance of a few time steps. The simulations were run for 100 time step updates (total scenario length of around 15 minutes). Between 5-10 vehicles of both arrival and departure types were included in each scenario. Figure 8 shows an example of final trajectories for a 10 landing 10 departure scenario. Not all scenarios completed successfully. In cases where departing aircraft joined the problem when a landing aircraft was due to enter

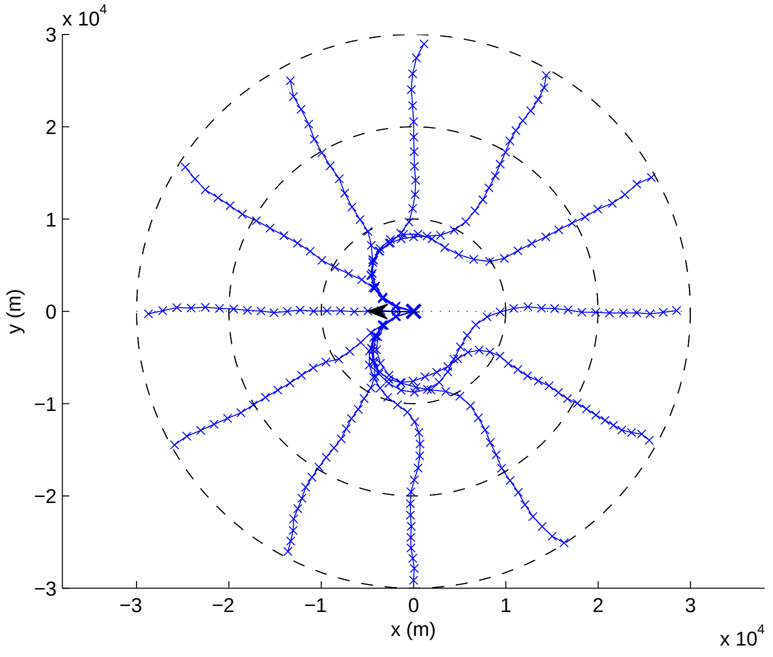

Figure 6. Departure trajectories for 12 single aircraft taking off from a runway at the origin in a Westerly direction before heading to their desired bearing

Aircraft Type	Coefficient	Value	Associated Cost
Departure	α_1	0.4	Difference in bearing from desired to current
	α_2	0.1	Fuel minimisation
	α_3	0.25	Difference in altitude from desired to current
	α_4	0.25	Difference in airspeed from desired to current
Arrival	$\tilde{\alpha}_1$	0.25	Difference from aircraft heading to flowfield
	$\tilde{\alpha}_2$	0.65	Difference from aircraft altitude to nominal altitude
	$\tilde{\alpha}_3$	0.1	Fuel minimisation

Table 1. Example objective function weightings used for optimisation

the landing envelope separation constraints were violated and no valid solution could be found. This is one of the restrictions of having fixed departure times. These could be amended by allowing flexibility in scheduling like a real airport. Whether this flexibility can be directly implemented in the SMC optimisation without an outside scheduling agent is a topic of future work.

V.D. Low Fuel Scenario

In the low fuel scenario two arriving aircraft with the same distance till landing are initialised, one with plenty of fuel and the other with very little fuel remaining. This scenario is then run 20 times till both aircraft have landed to see how the aircraft behave. In each run of the scenario the aircraft are subject to a different set of disturbances from wind as well as the stochastic nature of the algorithm. Figure 9 shows one of the generated trajectories. In the case of a human operator we would expect the aircraft with low fuel to be prioritised for landing with the aircraft with more fuel being diverted or held. However as Table 2 shows the order in which the aircraft landed was actually more split. The cases where the low fuel aircraft landed second was due to the aircraft using its fuel sparingly for acceleration; most likely due to the fuel minimisation term in the objective function. Thus the high fuel aircraft proceeded faster towards the landing envelope and landed first. In both situations the low fuel aircraft lands before it runs out of fuel. To alter

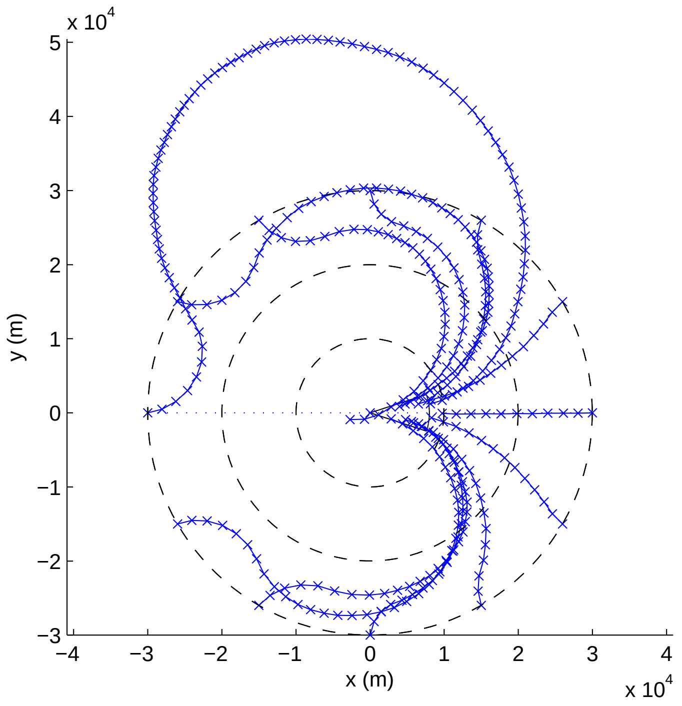

Figure 7. Arrival trajectories for 12 single aircraft entering the TMA at different angles before proceeding to the landing envelope to land on an runway from the East

this behaviour an outside observer with some form of priorities would be needed to be added to the system to schedule the order which aircraft may enter the landing envelope.

Aircraft which landed first	Number of times	Average fuel (kg) remaining on low fuel aircraft
High Fuel	11	22.6
Low Fuel	9	6.2

Table 2. Breakdown of which aircraft landed first in low fuel scenario and average fuel remaining

V.E. Congested Arrival Schedule

The congested arrival schedule is a demonstration of the number of aircraft the system can handle concurrently. Previous work by the authors did demonstrate the method for up to 20 aircraft simultaneously in cruise-like situations. However with the limited resource of the single runway it is not necessarily sensible to initialise 20 aircraft in the first time step and run until all have landed. Instead aircraft are introduced to the scenario gradually as would happen in a real airport and removed from the scenario as they land. No departure aircraft are considered in this scenario as this would necessitate flexible scheduling of take offs which is currently neglected for simplicity (but as mentioned is the subject of future work). Figure 10 shows 24 arrival aircraft landing trajectories over time. The maximum concurrency observed in this scenario was 8 vehicles.

V.F. Computation Speed

Previously the MPC-SMC method was demonstrated for fast computation of many aircraft problems in cruise-like conditions[20] with computation time per MPC update step varying up to 33 seconds for a 20 vehicle problem. This implementation had the simplification of fixed scenario lengths with all aircraft entering the problem at the start and no aircraft leaving the problem before the end. As such all aircraft were active during the entire scenario and computation time per step of the algorithm was deterministic

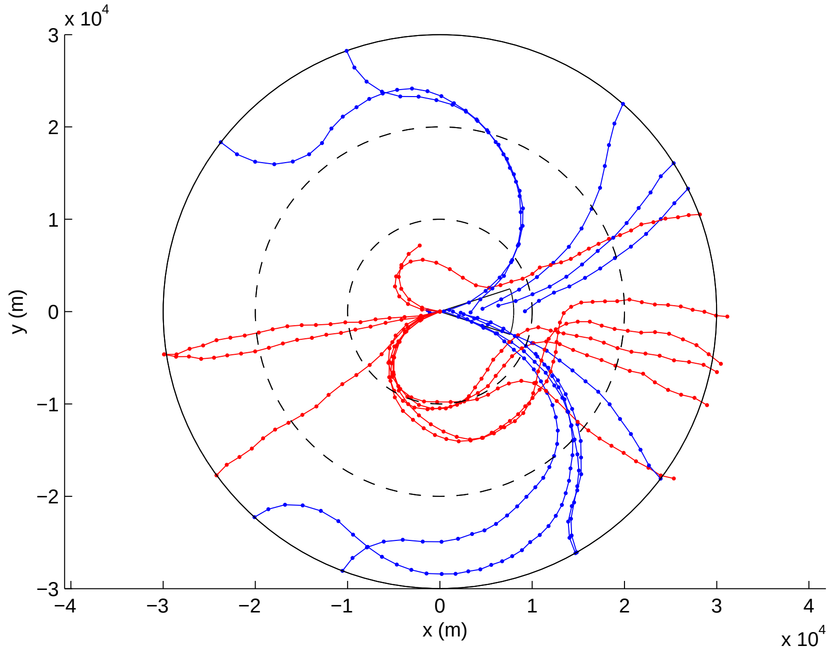

Figure 8. Trajectories of 10 arrival (blue) and 10 departure (red) vehicles over a 100 time step simulation

based on the number of aircraft in the problem. This is not true for this paper's implementation where the number of active aircraft in the problem varies step by step as aircraft both enter and leave the scenario.

Figure 11 shows the comparison of our previous implementation's time per update step compared to the average of this paper; fewer points are available for this paper's implementation due to the nature of the scenarios being solved however it offers an insight into the magnitudes involved.

This implementation is significantly slower than the previous incarnation, most notably due to the 10-fold increase in number of particles needed to adequately represent the search space. The problem of landing vehicles into a small terminal area whilst obeying all flight constraints is significantly harder than the previous cruise like situation and 1024 particles was not enough to reliably find feasible solutions. A larger or faster GPU would offer direct benefit to the computation speed through allowing more cores to process the particle threads and each core to work faster.

VI. Conclusions

This paper has discussed the implementation of a parallelised Model Predictive Control, Sequential Monte Carlo algorithm for solving the problem of conflict resolution and aircraft trajectory control in air traffic management specifically around the terminal manoeuvring area of an airport. The target problem was nonlinear in both dynamics and objectives, highly constrained, non-convex and uses a single decision-maker with multiple aircraft. The implementation included a spatio-temporal wind model and rolling window simulations for realistic ongoing scenarios. The method was shown to be capable of handling both arriving and departing aircraft including some with very low fuel remaining. The aircraft throughout were optimised for multi priority objective functions dependant on whether they are arriving or departing; the tuning of this was demonstrated. A novel flow field was proposed to smooth the approach trajectories for arriving aircraft and its effect on behaviour of aircraft was demonstrated. This led to some restrictions on the approach angles of aircraft arriving to the single runway such that they did not approach the runway directly into oncoming departure traffic.

Avenues for further work have been identified as allowing flexible times for departure aircraft take off and a more formalised approach of separating landing aircraft in the landing envelope other than the standard spatial conflict detection. There is scope for speeding up the computation speed of the method through

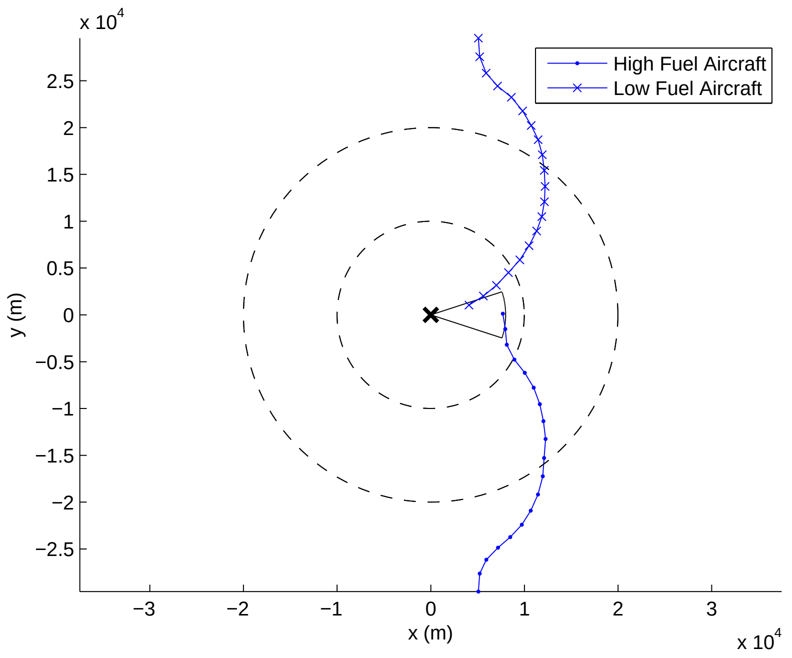

Figure 9. Example trajectory for an aircraft with low fuel and an aircraft with high fuel approaching the landing envelope at the same time (low fuel landing first)

either transferring to a higher spec GPU or optimisation of the existing code. This would allow for more complexities to be included in the GPU kernel such as constraints on vertical acceleration for passenger comfort and an external agent weighting the priorities of aircraft against one another for landing.

References

[1]Atkin, J. A. D., Burke, E. K., Greenwood, J. S., and Reeson, D., "On-line decision support for take-off runway scheduling with uncertain taxi times at London Heathrow airport," *Journal of Scheduling*, Vol. 11, No. 5, 2008, pp. 323–346.

[2]NATS, "Acting Responsibly: NATS and the Environment 2009," 2009.

[3]Hu, J., Pradini, M., and Sastry, S., "Optimal Coordinated Maneuvers for Three Dimensional Aircraft Conflict Resolution," *AIAA Journal of Guidance, Control and Dynamics*, Vol. 25, 2002, pp. 888–900.

[4]Ghosh, R. and Tomlin, C., "Maneuver Design for Multiple Aircraft Conflict Resolution," *Proceedings of the American Control Conference*, Chicago, Illinois, June 2000, pp. 672–676.

[5]Bicchi, A. and Pallottino, L., "On Optimal Cooperative Conflict Resolution for Air Traffic Management Systems," *IEEE Transactions on Intelligent Transporation Systems*, Vol. 1, No. 4, 2000, pp. 221–232.

[6]Prete, J., Krozel, J., Mitchell, J., Kim, J., and Zou, J., "Flexible, Performance-based Route Planning for Super Dense Operations," *In Proceedings AIAA Guidance, Navigation and Control Conference and Exhibit*, Hawaii, August 2008.

[7]Richards, A. and How, J., "Aircraft Trajectory Planning with Collision Avoidance using Mixed Integer Linear Programming," *Proceedings of the American Control Conference*, Vol. 3, Anchorage, Alaska, May 2002, pp. 1936 – 1941.

[8]Mesgarpour, M., Potts, C., and Bennell, J., "Models for aircraft landing optimization." *Proceeding of the 4th international conference on research in air transportation*, Budapest, Hungary, 2010.

[9]Sölveling, G., *Stochastic Programming Methods For Scheduling Of Airport Runway Operations Under Uncertainty*, Ph.D. thesis, Georgia Institute of Technology, 2012.

[10]Khardi, S., "Aircraft Flight Path Optimization The Hamilton-Jacobi-Bellman Considerations," *Applied Mathematical Sciences*, Vol. 6, No. 25, 2012, pp. 1221 – 1249.

[11]Maciejowski, J., *Predictive Control with Constraints*, Pearson, Prentice Hall, 2002.

[12]Doucet, A., de Freitas, N., and Gordon, N., editors, *Sequential Monte Carlo Methods in Practice*, Springer, 2001.

[13]Kantas, N., Maciejowski, J., and Lecchini-Visintini, A., "Sequential Monte Carlo for Model Predictive Control," *International Workshop on Assessment and Future Directions of NMPC*, Pavia, Italy, September 2008.

[14]Eele, A. and Maciejowski, J., "Comparison of Stochastic Methods for Control in Air Traffic Management," *International Federation of Automatic Control (IFAC) World Congress*, Milan, September 2011.

[15]Villiers, J. P., Godsill, S., and Singh, S., "Particle Predictive Control," *Journal of Statistical Planning and Inference*, Vol. 141, 2011, pp. 1753–1763.

[16]Lee, A., Yau, C., Giles, M., Doucet, A., and Holmes, C., "On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods," *Journal of Computational & Graphical Statistics*, Vol. 19:4, 2010.

[17]Fernández-Villaverde, J. and Rubio-Ramirez, J., "Estimating Macroeconomic Models: A Likelihood Approach," *Review of Economic Studies*, Vol. 74, No. 4, 2007, pp. 1059–1087.

[18]Rosenthal, J., "Parallel computing and Monte Carlo algorithms," *Far East Journal of Theoretical Statistics*, Vol. 4, 2000, pp. 207–236.

[19]Stone, J. E., Phillips, J. C., Freddolino, P. L., Hardy, D. J., Trabuco, L. G., and Schulten, K., "Accelerating Molecular Modeling Applications with Graphics Processors." *Journal of Computational Chemistry*, Vol. 28, No. 16, 2007.

[20]Eele, A., Maciejowski, J., Chau, T., and Luk, W., "Parallelisation of Sequential Monte Carlo for Real-Time Control in Air Traffic Management," *Submitted to the Proceedings of the Conference on Decision and Control*, 2013.

[21]Kitagawa, G., "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models," *Journal of Computational and Graphical Statistics*, Vol. 5, 1996, pp. 1–25.

[22]Lymperopoulos, I. and Lygeros, J., "Improved Multi-Aircraft Ground Trajectory Prediction for Air Traffic Control," *Journal of Guidance, Control and Dynamics*, Vol. 33, No. 2, 2010, pp. 347–362.

[23]Lygeros, J. and Glover, W., "A Multi-Aircraft Model for Conflict Detection and Resolution Algorithm Evaluation," Tech. rep., Hybridge IST-2001-32460, 2004.

[24]Cole, R., Richard, C., Kim, S., and Bailey, D., "An Assessment of the 60km Rapid Update Cycle (RUC) with Near Real-Time Aircraft Reports," Tech. rep., Lincoln Laboratory, Massachusetts Institute of Technology, 1998.

[25]Marsaglia, G., "Xorshift Random Number Generators," *Journal of Statistical Software*, Vol. 8, No. 14, 2003.

[26]Eurocontrol, *User Manual for BADA 3.10*, April 2012.

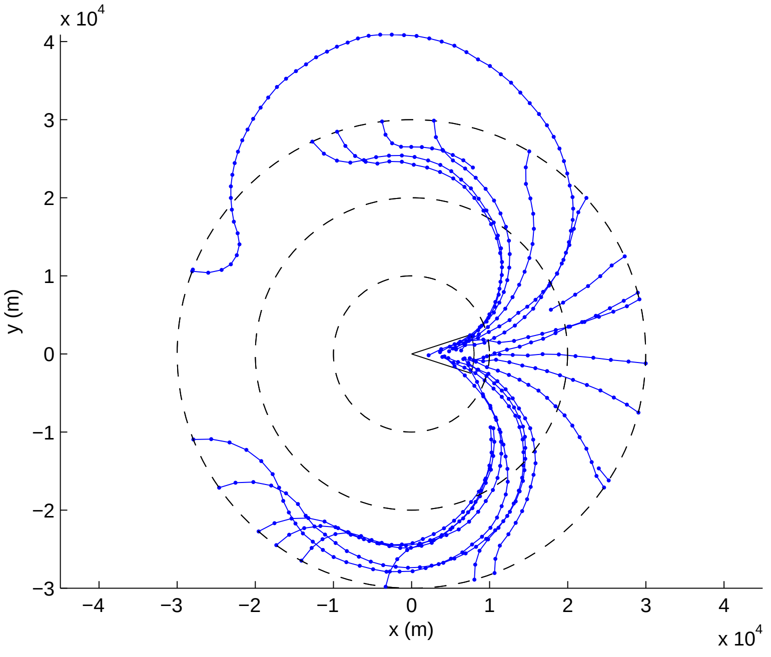

Figure 10. Aircraft trajectories from a 100 step simulation with 24 arriving aircraft from random locations

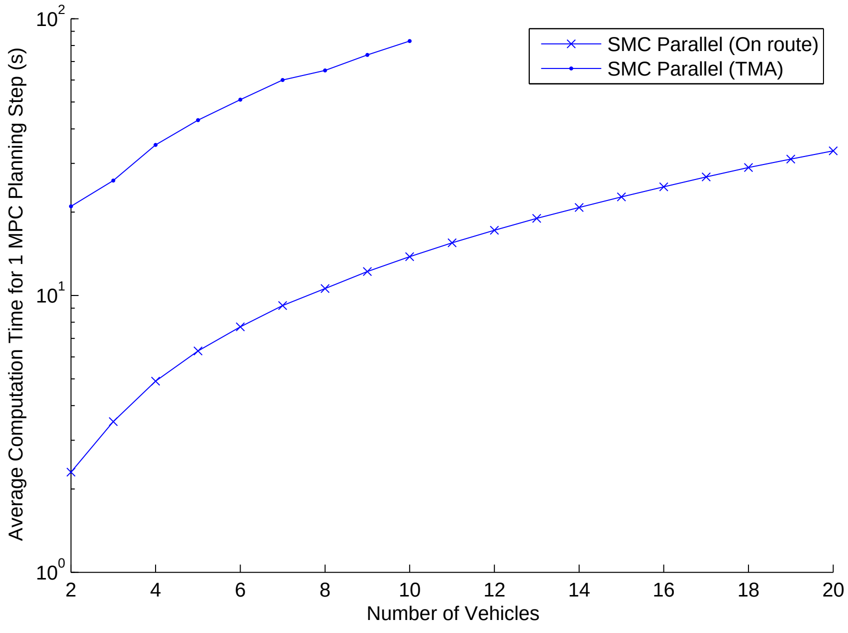

Figure 11. Comparison of previous and current computation time spent per time step for problems with different number of vehicles