33RD CONGRESS
OF THE INTERNATIONAL COUNCIL
OF THE AERONAUTICAL SCIENCES
STOCKHOLM, SWEDEN, 4-9 SEPTEMBER, 2022

ICAS 2022 SWEDEN

REAL-TIME CALCULATION AND ADAPTION OF CONFLICT-FREE AIRCRAFT GROUND TRAJECTORIES

Lennard Nöhren, Meilin Schaper, Lukas Tyburzy & Kathleen Muth

German Aerospace Center (DLR)
Institute of Flight Guidance
Braunschweig, Germany

Abstract

To improve the quality of airport surface operations and to pave the way for more autonomous systems, the calculation and adaptation of ground trajectories for aircraft is the backbone for every improvement. These trajectories should be conflict free and easily adaptable to changing conditions in real-time, but on the other hand optimized regarding configurable criteria. This paper describes how this can be achieved using artificial intelligence, especially a multiobjective A* algorithm coupled with a genetic algorithm. The genetic algorithm uses a flexible objective function that can be used to tune the resulting trajectories to the specific needs of the airport/air navigation service provider.

Keywords: Air traffic control, ground movement, taxi routing, multiobjective A* algorithm, genetic algorithm

1. Introduction

In the years before the Covid-19 pandemic the amount of air traffic was steadily rising. Based on the October 2021 forecast of Eurocontrol, air traffic will reach pre-pandemic levels between 2023 and 2027 and will keep rising afterwards [1]. This increasing demand in air traffic capacities will lead to more and more congestion not only in the air, but also on the ground at big airports all around the world. This congestion will lead to increasing delay [2]. Controller assistance systems can be introduced to reduce this problem and ensure the optimal processing of flights. Surface Management Systems (SMAN) are an important assistance tool for controller tasks regarding ground traffic. They can exist in different levels of assistance [3]. On the lower levels they focus on the creation of taxi routes and help the controller supervising the traffic using conformance monitoring functionalities. On the higher levels they support the controller with advisories or even send commands to the aircraft autonomously. Therefore, a high level SMAN needs to be able to generate trajectories of flights based on actual data that contain precise target times for in-block and take-off [3]. These trajectories need to be conflict free. In case a departure manager (DMAN) exists for optimising the departure sequence, the trajectories should ensure that the planned take-off sequence can be adhered to [4]. Since the real-world situation can change fast and often as controllers and pilots might not be able to follow calculated trajectories perfectly, the SMAN also needs to be able to adapt the trajectories fast and reliable. At the same time, the trajectories should be kept stable as long as possible not to increase the workload of the controller and pilots.

In future research full automation of ground traffic could also become more relevant. Complete trajectories that are conflict free and can be adapted easily and fast are deemed as absolutely necessary for a system that will manage aircraft ground traffic completely autonomous [3].

There has been some previous research in the area of multiobjective optimized, conflict free trajectory generation. Most of the works were based on a Mixed Integer Linear Programming (MILP) approach or genetic algorithms [5]. By applying MILP approaches it could be shown that a globally optimized solution can be found that leads to a large reduction in taxi time compared to a first come first served

scheduling approach. But the computation time to calculate such a solution can lie between few seconds and multiple minutes [6]. Therefore, this approach is not suitable for a real time system that needs to adapt to changing conditions with reasonable speed. Genetic algorithms on the other hand are heuristic algorithms specialized on a fast search within the most promising parts of the solution space. Although there is no guarantee to find the optimal solution, they have in general a much shorter run time, which makes them to a good choice for real time systems [5]. In [7] an approach using a genetic algorithm is proposed which guarantees to find a solution within 30 seconds. That time might still not be acceptable for a real time system at a busy airport. Other proposed solutions use a simplified model of the problem to reach better run time performance. In such models the possibilities for the algorithm to adapt the trajectories are often very limited or some constraints are not considered, which are supposed to be essential for realistic operational use. For example, in [8] trajectories can only be created from a fixed set of pre-computed routes with a fixed speed and they can only contain one hold point.

The work presented in this paper introduces the prototypical DLR SMAN TraMICS+ (Traffic Management Intrusion and Compliance System), which is able to generate and adapt trajectories for complex traffic scenarios. It could be classified as a Level 3 or Level 4 SMAN according to [3]. It uses a combination of a multiobjective A* algorithm for optimal route generation and a genetic algorithm for time-based conflict resolution and optimization. TraMICS+ can generate trajectories within seconds and the target function is adaptable to the needs of the user. The resulting trajectories are automatically adapted to any changes in the surroundings or non-conformance and can be manually changed by the controller if desired.

The paper is organized as follows: Chapter 2 contains a description of the general components of TraMICS+ that are needed for the trajectory generation and adaption. Chapter 3 introduces in detail the genetic algorithm, which is used for the optimization of the trajectories. Chapter 4 describes experiments, including the test and simulation environment of the TraMICS+. Chapter 5 presents and analyses the results of these tests and compares them for different configurations of the algorithms. Finally, in chapter 6 a conclusion about the work is drawn and some possible future steps are explored.

2. TraMICS+ Overview

TraMICS was initially designed to be a security tool using some safety-related functions like route conformance monitoring to determine the security situations indicator for a ground controller working position [9]. Being started with just routes, the enhancement to TraMICS+ includes conflict-free trajectory (route and time) calculation, optimization, monitoring and adaptation.

The calculation of conflict free trajectories is done in three steps: First the optimal route is calculated for the flight, using an A* algorithm (section 2.1). This route does not contain any temporal information yet. It is then enhanced to an initial trajectory by calculating the expected taxi times while not considering any other flights. This first trajectory might therefore not be conflict free to other trajectories. This is changed in the third step, where a genetic algorithm is used to resolve any conflicts in the newly generated trajectory and to optimize it based on a configurable target function (chapter 3).

Every assistance tool for trajectory generation on the ground needs the airport's topology as base for many of its computations, especially for the route calculation. The topology is stored internally as a graph that contains all taxiways, relevant points of the airport and also the topology of the apron. This enables trajectories from the runway to the parking position and vice versa, also containing a pushback if necessary. The nodes of the graph are called taxi points. They can be any taxiway intersection, stop bar, stand or runway entry/exit. The edges of the graph are the taxiways connecting the taxi points. These edges can be uni- or bi-directional and can be blocked if necessary. The topology currently used for experiments and implemented in TraMICS+ is a not simplified but meanwhile deprecated version of Hamburg airport. It consists of 733 taxi points and 1045 connecting edges. Figure 1 shows a schematic overview of the used topology. It is inserted into the TraMICS+ via an XML file and can be easily replaced by any other airport's topology if desired.

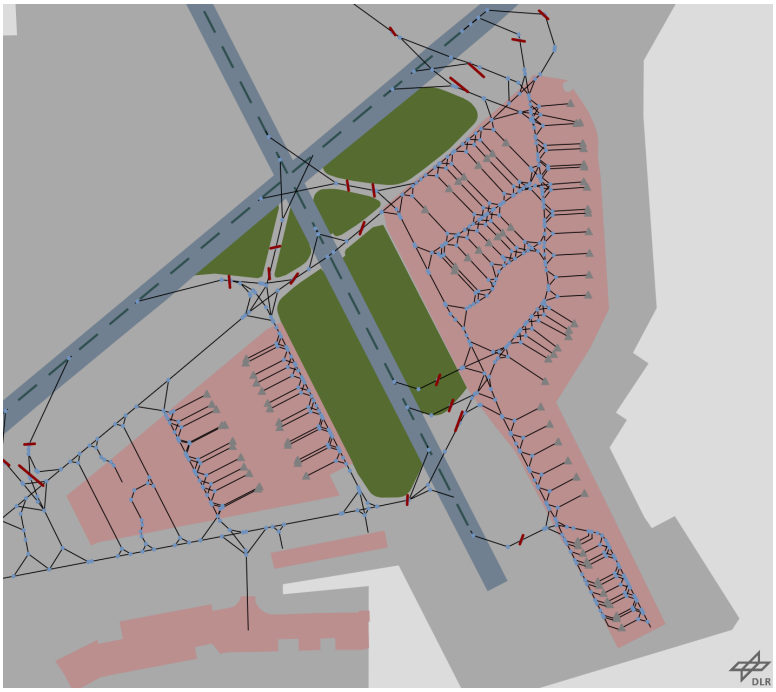

Figure 1 – Excerpt of the used topology. Blue dots mark taxi points of the graph, grey triangles mark stands and red lines mark stop bars.

2.1. Router

The topology graph is used to calculate the best route from the stand to the runway, or vice versa, for each flight. This is done with a multiobjective A* algorithm. In contrast to a regular A* algorithm, the multiobjective A* does not only use the distance between nodes in the graph as costs to calculate the lowest-cost route, but can also use additional parameters to calculate the cost of a route. In case of TraMICS+, two additional parameters are used. The first parameter is based on the difference in heading of two consecutive edges of the route in the topology graph, to ensure that the router prefers more straight routes instead of routes that contain many turns, causing the aircraft to break and accelerate more often. The second parameter is a penalty for too sharp angles between two following route edges in the topology graph. This prevents illegal turns (such as U-turns) in the routes, that the aircraft are unable to follow. It also ensures that the routes are planned in the direction of the current heading of the aircraft when recalculating routes.

To reduce calculation time, all lowest-cost routes between each stand and each runway are pre-computed when the system starts up. This way the router has to be called only when a flight deviates from its lowest-cost route or when a specific route is requested.

For departures that are parked at a stand requiring a pushback the lowest-cost route needs to be enhanced with a pushback segment. This is important for the planning, since departures block the taxiway on which they are pushing back for approximately two minutes while their engines are starting up. The pushback segment is created by determining the direction in which the route of the departure starts and inserting a new segment into the route that lies on the taxiway with the opposite direction. Figure 2 shows a route of a departure with a pushback segment.

2.2. Trajectories

Additional to the spatial information of the route, a trajectory contains temporal information, i.e. the times at which the flight reaches each taxi point. In TraMICS+ trajectories can contain a planned hold duration and additionally the speed at each point. They are also enhanced with further information, like the last passed taxi point or clearance information of the flight.

TraMICS+ generates an initial trajectory for each flight simply by taking its route, calculated by the router, and computing the times for each taxi point of the route assuming there is no interference with other trajectories. The computation is based on the estimated off block time (EOBT) or, if available,

the Target Off-block Time (TOBT) for departures and the estimated landing time (ELDT) for arrivals. It assumes a standard speed of 15 knots on straight taxiways. In curves the speed is reduced using a multiplier, that is calculated based on the turning angle of the curve. For departures a waiting time of 90 seconds is added at the end of the pushback as approximation for the time it takes to power on the engines of the aircraft. In case the flight has already started to move on the route when a trajectory is generated or adapted, the times are calculated based on the current time and position onwards.

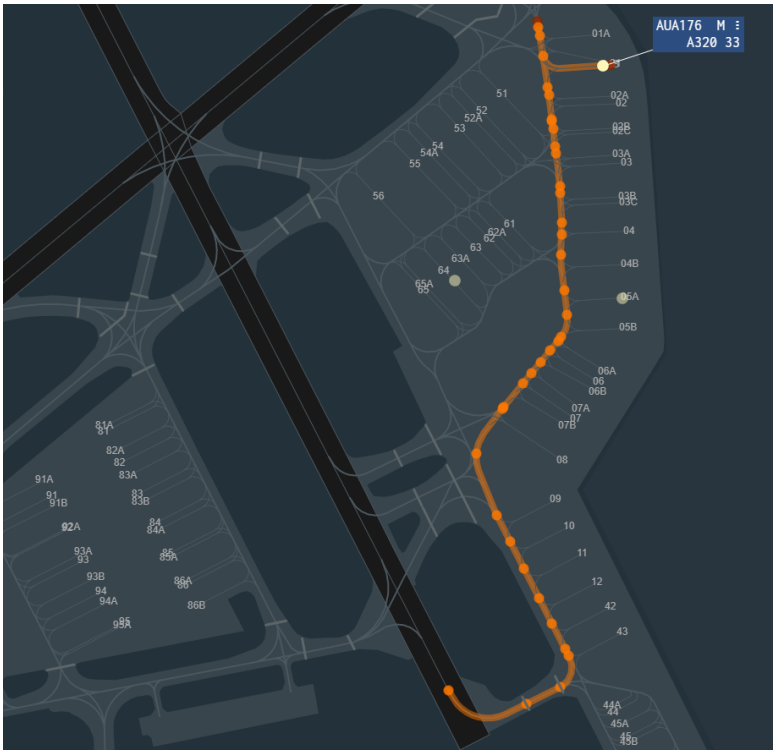

Figure 2 – Visualization of a trajectory planned by TraMICS+. Orange dots are possible hold points, red dots are points where an actual hold is planned.

There is no guarantee that the initial trajectories calculated in this manner are conflict free, since trajectories of other flights are not considered during the calculation. To achieve the desired conflict freeness the trajectories are adapted in a separate step. This step is described in detail in chapter 3. Figure 2 shows an example of a trajectory for a departure. It starts with a pushback, leading to a hold at the end of the pushback segment, followed by the taxi trajectory.
Trajectory generation can be triggered automatically (i.e. by the system) or manually (i.e. by the air traffic controller) in several ways. 15 minutes before the EOBT/ELDT of a flight the first trajectory is planned. This trajectory is then automatically updated when either a relevant field of the flight plan is changed, e.g. gate, runway, EOBT/ELDT, or when the conformance monitoring registers a non-conformance and issues a trajectory request for an active flight. Trajectory requests can also be issued manually by the controller if he wants to modify the planned route or insert a hold at any point of the trajectory. The rest of the trajectory is then adapted to accommodate the changes and any new conflicts that might have been created are resolved.

2.3. Conformance Monitoring

As described in chapter 2, the TraMICS was originally intended for security purpose and uses amongst others non-conformance detection to derive the security situation indicator. Therefore, the conformance monitoring is an essential functionality: It monitors the conformance to the given routes, clearances and, for TraMICS+, trajectories. This is crucial, since the trajectories shall always reflect the current situation. Multiple aspects of the flights are monitored and accordingly different types of non-conformance can be detected:

- Route deviation: The flight deviates from the planned route. This can have some special cases like pushback deviation (flight deviates from pushback route) or early/late exit (flight uses a different runway exit then anticipated). In any of these cases it is necessary to calculate a new route and consequently also a new trajectory, considering the new position of the flight.
- Time deviation: The flight is either too late or too early at the planned point. If a configured threshold is passed, the trajectory of the flight is re-calculated to ensure that the planning is still achievable and conflict free.
- Clearance violation: The flight moves without the correct clearance. Being the only non-conformance, this does not trigger a re-calculation of the trajectory but is shown to the controller as alert.

3. Genetic Algorithm

As described in section 2.2, the initially calculated trajectories might contain conflicts to each other. The term “initial trajectory” does not only indicate the first trajectory of a flight, but rather every trajectory that was generated or adapted for a flight as described in section 2.2 which is neither optimized nor assured to be conflict-free. To solve these potential conflicts the TraMICS+ uses a genetic algorithm. In the following sections the general functionality and the different components of the genetic algorithm are described in detail.

3.1. General Functionality

Genetic algorithms are inspired by genetics and evolution in nature. They hold a “population” of solutions described by a parameter set given as a so-called "chromosome". These solutions are adapted and improved to find an optimized result over multiple iterations, or “epochs”. At the beginning of the optimization an initial population is randomly generated. In each following epoch new solutions are generated out of existing solutions in two ways: “crossover” and “mutation”. In crossover two “parents” are selected from the population and their genetic information in the chromosomes is combined to create a new “child” solution. These children are then mutated with a certain probability. During mutation a random change is introduced to the solution. At the end of each epoch the solutions are rated with a “fitness” or “penalty” function. Solutions with the worst fitness or the highest penalty rating are removed. This is called “selection”. To calculate the penalty value TraMICS+ analyses the generated trajectories and performs a conflict detection. In our implementation half of the population is terminated during selection and in the following epoch new solutions are generated until the original population size is reached again. This scheme is repeated until a stopping criterion is reached. This might be when a certain number of epochs has been performed or when a solution is found that lies below a certain penalty threshold. In our implementation a moving penalty threshold is used: At the beginning the threshold is very low, to force the algorithm to run a certain minimal duration to find solutions with high quality. With every epoch the threshold is increased a little bit. This is done because it might be very hard or not even possible to find solutions with a low penalty in complex traffic situations. With this moving threshold, solutions with higher penalties can still be accepted if the algorithm was not able to find a better solution after a certain run time. This way the run time of the algorithm is bound and the probability to find any solution in complex situations at all is increased drastically.

3.2. Chromosomes

A solution of a genetic algorithm is often called a “chromosome”. Chromosomes are very simplified versions of the data structures that the algorithm shall optimize. This reduces memory load and simplifies the changing of the solutions. In our case the chromosomes are simplified representations of the trajectories containing a reduced list of trimmed taxi points, where each trimmed point has a speed value and a hold duration only. The speed value is given in three distinct levels: normal, fast, slow. This is done, since it is very hard to adhere to more precise taxi speeds in aircraft taxiing on ground, unless an e.g. electric taxi is used, which is not standard at most airports nowadays. Therefore, it would be unnecessary to advise exact speeds to the controller. It would increase complexity

and load for the controller without additional benefit. Internally the three speed levels correspond to 10, 15, and 20 knots.
The taxi points chosen to be in the chromosomes are only the relevant points of the trajectories, like stop bars and taxiway intersections. By constructing the chromosomes this way, the genetic algorithm is limited to change hold durations and taxi speeds at these relevant points, or change the route by swapping points. This is a big simplification of the complexity of the problem that reduces the number of possible solutions by a large amount, while still keeping enough room for the algorithm to find diversified solutions. The full trajectories can be easily reconstructed from a chromosome. This is needed e.g. for the conflict detection (section 3.5). Each chromosome not only contains the simplified trajectory of the flight for which the genetic algorithm was triggered, but also trajectory information of all other flights and results of the conflict detection. The genetic algorithm is able to adapt any of these trajectories, because in some cases this might lead to simpler solutions for conflicts. Nevertheless, it is desired that the algorithm mainly changes the trajectory for which the optimization was triggered. This is ensured with the described penalty function (section 3.6) which needs the stored conflict detection results as well.

3.3. Crossover

Crossover is the functionality in which two parents from the population are combined to create a new child solution. This is inspired by reproduction in nature, where parents pass on their positive traits to their children. There are many possibilities how to perform the crossover in genetic algorithms. The one used in our algorithm just splits the chromosomes of the parents in the middle and then combines the first half of the first parent with the last half of the second parent, or vice versa. In case the two parents have a different route, a common point between the two is searched first and the split is then done at that common point.
During testing it became apparent that the crossover operation had no big impact on the quality of the solutions found by our genetic algorithm. More often the algorithm even needed more epochs to find a good solution when crossover was used as opposed to when it was disabled. Therefore, the crossover operation was deactivated in our algorithm for the final evaluations.

3.4. Mutation

Mutation is next to crossover the second way how the genetic algorithm can modify chromosomes to find new solutions. A mutation is usually done with a low probability after the crossover. As described above we observed better performance when crossover was disabled and new solutions were created by mutating existing solutions instead only.
In each mutation only one of the simplified trajectories of a chromosome is modified. The trajectory to be changed is selected randomly, but the trajectory for which the optimization was triggered is selected with an increased probability. There are two different variants of mutations in our algorithm. The first variant can change the route of a trajectory. This is done by randomly selecting a known conflict and blocking the taxiway segment on which the conflict would occur. Then the A* algorithm for the route generation is applied again to generate the best route without the blocked segment. This route is then used as the new base for the chromosome. This is a very major mutation, since a changed route usually leads to a completely different trajectory and a smaller speed change might also be sufficient to solve the conflict. Because of that it is only done with a low probability. Route changes should only be done in case no other feasible solution is found, since they generate additional load for the controller and routes changed this way have a higher cost than the initially calculated route.
The second mutation variant just changes the speed or hold duration at a point of the simplified trajectory. Both, the point to be changed and the way how it is changed, i.e. hold duration or speed category, is selected randomly. Nevertheless, there are some parameters that can be used to force the mutation in a certain direction:

- For departures a parameter exists that controls the probability with which a hold at the stand is inserted. Holds at stands are in general a very desired way to solve conflicts, since the engines of the aircraft are not running at the stand, so no kerosene is wasted and additionally the flight does not block any taxiway while it is waiting.

- Another parameter controls if the mutation shall be performed directly in front of a previously detected conflict. This can help the algorithm to find better solutions faster, because if it would just randomly select the point to be changed it might select a point behind the conflict, which won't have any benefits.
- The third option is a configurable parameter for the probability that an existing hold point is adapted instead of inserting a new hold point when a hold mutation is done. Changing an existing hold instead of inserting a new one should be preferred if possible, because it reduces the amount of brake and acceleration manoeuvres the aircraft has to perform.

When these three probabilities are set to high values the algorithm is restricted further in searching solutions which might lead to less variable solutions, but also to much shorter optimization times.

3.5. Conflict Detection

The main goal of the genetic algorithm is to solve conflicts, the initial trajectories might have. Therefore, the conflict detection algorithm is a very important component. A conflict between two flights shall be detected if they are spatially and temporally closer to each other than configured thresholds at any point of their trajectories. We have chosen 50 meters and 20 seconds as default separations. Additionally, there are some special cases: runways that are currently occupied by a departing or landing aircraft need to be blocked longer, so there is a larger conflict window on runways. Aircraft that are currently holding also need to be considered differently. Multiple aircraft may hold directly behind each other to form a queue, for example in front of a runway. These aircraft get closer to each other than the conflict thresholds, so in such a case different checks are performed, e.g. verifying that the order in which the aircraft arrive and leave the point is the same.
Every time a trajectory is adapted in any way all these checks need to be carried out for every other flight with a trajectory. This will lead to massive performance costs if it is not done in an optimal way. By far the biggest part of the runtime of the genetic algorithm is spent with conflict detection in our implementation. To achieve low computational costs a special data structure is used: It contains the occupation times of every flight at any taxi point of the topology, i.e. the time at which the point is blocked because an aircraft is standing there or taxiing by. Additionally, for every taxi point two lists are stored. One of these lists contains which other taxi points are adjacent in the topology to be able to check all points within the spatial conflict threshold. The other list contains taxi points that are connected via a link segment, that is longer than the conflict threshold. This is needed to ensure no conflicts exist on such a long segment. For example, if a faster flight tries to pass a slower flight in the middle of the segment or if two flights taxi on the segment in the opposing direction and meet in the middle. Whenever a trajectory is created or adapted the occupation time windows at each trajectory point are inserted into the data structure. Any conflicts to other flights can then directly be detected by checking if the newly inserted occupation time overlaps with any existing one.
With these data structures used in the described manner, the time spent with conflict detection could almost be reduced by a factor of 10 compared to a naive approach in which a new trajectory is just compared to each existing trajectory serially, point by point.

3.6. Penalty Function

As described before, the penalty function is used to evaluate the solutions of the algorithm for each epoch. That means it controls which solutions are discarded in the selection, when the stopping criterion of the genetic algorithm is reached and which solution is returned as the final result. As even small changes in the penalty function can change the results of the algorithm very drastically, it is very important to select a penalty function that is suitable for the given problem and leads to the desired results. We have chosen the penalty function as a weighted sum of the following properties of a solution:

- W1 – Conflict weight: This is the weight for all remaining conflicts that exist in the solution. Each conflict has a severity between one and two which are summarized and afterwards multiplied

with this weight. The severity of a conflict is defined as follows:

$$S = \left(1 - \frac{sDist_C}{sDist_T} + 1 - \frac{tDist_C}{tDist_T} + 2\right)/2 \tag{1}$$

Where $sDist_C$ is the spatial distance of the two flights in the conflict, $sDist_T$ is the spatial distance threshold, $tDist_C$ is the temporal distance in the conflict and $tDist_T$ is the temporal distance threshold for conflicts. This is supposed to help the algorithm to add changes in the right direction. A small change might not be enough to solve a conflict, but if it at least reduces the conflict severity, it is a step in the right direction. In general this is configured as the largest weight, since the removal of conflicts is the main goal of the genetic algorithm and a solution should never be acceptable as long as it still contains a conflict.

- W2 – Duration weight: This weight is multiplied with the increase of the taxi duration compared to the initial trajectory. The increased taxi duration is only an approximation based on the total hold duration (except stand holds) and the increased taxi distance from route changes. An increased taxi duration is penalized, because it costs more kerosene and may lead to more aircraft that are taxiing at the same time.

- W3 – Hold weight: The number of holds that were inserted in the solution is multiplied with this weight. Holds that are operationally necessary, like the hold at the end of the push back until all engines are up, are not considered. As stated before, each hold leads to a large cost, because the flight needs to break and accelerate every time. Therefore, the number of holds should be minimal.

- W4 – Speed change weight: This weight is added for every time the speed of the flight changes. Every speed change adds work load to the controller, since he needs to communicate it to the pilot. This does not include necessary speed changes for curves, but rather just changes to the base speed in the chromosome.

- W5 – Fast point weight: The weight for every point in the trajectory where the flight should move fast. When the aircraft is moving faster it burns more kerosene.

- W6 – Slow point weight: The weight for every point in the trajectory where the flight should move slow. If the aircraft taxis slower it increases the total taxi duration, can block following aircraft and may lead to more aircraft that are taxiing at the same time in general.

- W7 – Existing trajectory weight: As described in section 3.2, the chromosomes contain the existing trajectories for all active flights. This weight is applied for every previously calculated trajectory that was modified by the genetic algorithm to create this solution. This is done to reduce the number of active trajectories that are adapted. Changing trajectories may lead to a lot of additional load for the controller, because he may have given clearances already that he needs to revoke or change if the trajectory is changed afterwards.

- W8 – Route change weight: This weight is multiplied with the number of flights for which the route was changed during the optimization. The initial route is always the best route, as calculated by the multiobjective A* algorithm, so a route change leads to a longer taxi distance and it also may increase the work load for the controller, because flights may have already cleared routes or, if the aircraft is still at the stand, the controller might already have mentally agreed to the former proposed route. This means route changes should only be advised if there is no other solution, or any other solution would be very expensive as well.

- W9 – Stand hold weight: This weight is multiplied with the sum of additional waiting times at the stand of departures. Holding departures longer at stands is in general a very desired way to solve conflicts, since it does not burn any kerosene or block any taxiways. Nevertheless, a small weight for this is needed, because otherwise flights might be held at their stand much longer than necessary.

- W10 – Big delay weight: A threshold can be configured at which the taxi delay calculated for W2 and W9 is considered as “big delay”, default is 300 seconds. For every delay exceeding this threshold the big delay weight is added to the penalty. On the one hand this is done to penalize very large delays harder and to force the algorithm to search for other solutions. On the other hand, this weight should help to distribute delay better between multiple flights, instead of just delaying a single flight by a very large amount.

Each of these weights is set to a default value that was determined by parameter tuning with the goal of enabling the algorithm to find operationally practical solutions with a high reliability. The used process is described in more detail in chapter 4 and the default values are listed in table 1. It is possible to change each weight, as the values are configurable. This way the user can tune the algorithm to generate results that are fitting for his own needs and use cases. Some different configurations of the penalty function are compared in the next chapters.

4. Experiments

TraMICS+ and its algorithms were evaluated with simulations using the NLR ATC Research Simulator (Narsim) [10]. This simulator can be used with recorded traffic scenarios as well as for interactive traffic simulations using pseudo pilots. For all tests the above-mentioned topology was used with two 50 minutes traffic scenarios. One scenario contains 22 flights, the other one 34. The smaller scenario represents a typical traffic situation for Hamburg airport. The larger scenario represents an extremely busy situation to further challenge the algorithms [11]. Two types of tests were performed: Human-in-the-loop (HITL) tests and replays. During HITL tests, humans acted as air traffic controller, controlling the ground traffic based on the advisories of the TraMICS+ and pseudo-pilots answered the controller’s commands and moved the aircraft in the simulator accordingly. This way the usability of the trajectories and the advisories of the TraMICS+ could be tested in a realistic environment.
The other type of tests used previously recorded traffic data, that was replayed with the Narsim for the first ten minutes. Afterwards the resulting trajectories were analysed. These tests could be run multiple times with exactly the same inputs in relatively short time. Since the genetic algorithm is not deterministic it is necessary to repeat any tests regarding the trajectories to reduce the impact of random variations. With these repeated tests a parameter tuning of the algorithm was performed as well. The genetic algorithm has a lot of parameters that have a large impact on the performance: The 10 weights of the penalty function (section 3.6), for which some reasonable default values are needed, and some parameters for the general functionality of the algorithm like population size, the maximal number of epochs, the stopping penalty and stopping penalty increase, whether crossover is enabled or not and mutation probability.
After the first parameter tuning was finished, more extensive tests for a precise analysis of the trajectories and general performance of the genetic algorithm could be performed. For these tests’ different profiles (i.e. different value settings) for the penalty function were created. These profiles have different goals for the optimization that could be used for different types of applications:

- Standard: With this profile the genetic algorithm should find solutions that are generally acceptable to experts and be reliable in solving any conflicts.

- Green: This profile tries to reduce the environmental impact of the traffic. The trajectories should be optimized in such a way that the kerosene usage and thereby the damage to the environment is minimized. We assume that the most important factor for that is reducing unnecessary accelerations and brake procedures.

- Consistent: This profile should reduce the work load on the controller, by reducing the number of changes to already cleared trajectories and routes.

- Earliest: The goal of this configuration is to generate trajectories with which the aircraft arrive at their destination as early as possible. That means that the taxi time including holds should be minimized and an increased taxi speed is preferred as opposed to reduced speed. This way the throughput of the airport could be increased. Many models that are currently used to calculate

kerosene consumption and with that the environmental impact on the ground, are only based on the taxi duration [12].

The precise configuration for each of these profiles can be seen in table 1.
For the performance analysis of the genetic algorithm all tests were run on the same Hardware: A modern laptop using a 6-Core Intel i7 CPU [13] and 32 GB of RAM. No high-performance Hardware was used. So, all results of these tests can be achieved with hardware that would be feasible to use in an operational environment, without additional costs.

Table 1 – Overview of the weights of the penalty function for the different profiles. Values in bold differ from the standard configuration.

Penalty Weight	**Standard**	**Green**	**Consistent**	**Earliest**
W1 – Conflict weight	10000	10000	10000	10000
W2 – Duration weight	10	**5**	10	**20**
W3 – Hold weight	500	**2000**	500	500
W4 – Speed change weight	200	200	200	200
W5 – Fast point weight	25	**50**	25	25
W6 – Slow point weight	25	25	25	**50**
W7 – Existing trajectory weight	1000	1000	**2500**	1000
W8 – Route change weight	1000	1000	**4000**	1000
W9 – Stand hold weight	5	**2**	5	**10**
W10 – Big delay weight	1000	**500**	**500**	**2000**

5. Results

The parameter tuning of the genetic algorithm was a very important step before using the algorithm for any further evaluations. With the initial parameters, which were just selected by intuition, the algorithm was not able to solve many conflicts in complex traffic situations. On average 0.75 conflicts remained per trajectory after the optimization in the large scenario. Such a result would not be acceptable for operational use, since the advisories based on these trajectories could not be trusted in most cases.
The largest improvements during parameter tuning could be achieved by deactivating crossover, increasing the probability for insertion of holds at the stands and by changing the stopping criteria of the algorithm to allow a longer runtime. Based on these improvements a new standard configuration was selected for the algorithm that was used as baseline for all further test and evaluations.

Table 2 – Optimization statistics for the two recorded test scenarios with the standard configuration.

Scenario	**Avg. calculation time per Trajectory [ms]**	**Max. calculation time [ms]**	**Avg. remaining conflicts per trajectory**
Small Scenario (22 flights)	111.23	1296	0.0083
Large Scenario (34 flights)	400.98	2092	0.0731

Table 2 shows the results of the tests with the two recorded scenarios using the standard configuration that was determined during parameter tuning. It can be seen that the algorithm is able to solve almost all conflicts in the small scenario. Even in the large scenario, which reflects an extremely complex traffic situation, less than 10% of the trajectories contain a conflict to another trajectory. During the simulations it could also be noticed that most of the conflicts that remained after an optimization, were automatically solved when the genetic algorithm was called the next time. In such a complex scenario the algorithm is triggered every few seconds, that means the conflicts usually exist for a few

seconds only, until they are resolved. In case the amount of remaining conflicts is not acceptable for a certain use case, the parameters of the algorithm, especially the stopping criteria, could be further adapted. A trade-off between the quality of the results and the computation time can be made. In this work the goal was to reach real-time capabilities, so a few remaining conflicts were acceptable.
Observing the computation time of trajectories, it can be seen, that on average less than half of a second is needed, even in the complex scenario. In some cases, it might take up to two seconds to calculate or adapt a trajectory, but even that is still an acceptable computation time in a real time system, since the calculations are done in the background and the remaining system can be used normally.
For the analysis of the effectiveness of the different penalty profiles, as presented in table 1, the trajectories were examined in more detail. Mostly the attributes which are integrated into the penalty function were chosen as metrics, since they reflect the different possible requirements for a trajectory. Table 3 shows the average values for each metric after 10 runs of the small scenario per configuration, using the same algorithm parameters like population size, etc..

Table 3 – Test results of the different profiles. The average taxi duration of the aircraft and most of the weights of the penalty function were analysed. The best result of each metric is marked in bold.

Penalty Weight	**Standard**	**Green**	**Consistent**	**Earliest**
Taxi duration [s]	319.5	329.6	321.9	**313.1**
Increased taxi duration [s] compared to initial trajectory (W2)	25.7	25.6	25.1	**16.9**
Nr. of holds (W3)	0.204	**0.05**	0.133	0.2
Nr. of speed changes (W4)	0.496	**0.475**	0.508	0.508
Nr. of fast points (W5)	2.295	**0.975**	2.841	2.541
Nr. of slow points (W6)	1.537	2.216	1.408	**1.217**
Nr. of changed existing traj. (W7)	0.171	**0.125**	0.133	0.167
Nr. of route changes (W8)	0.037	0.033	**0.008**	**0.008**
Stand hold duration [s] (W9)	34.9	40.7	44.1	**29.6**
Nr. of big delay exceeds (W10)	0.013	0.058	0.058	**0**

In the “green” profile the number of holds could be reduced from an average of 0.2 per trajectory to 0.05 and the amount of fast points was more than halved. As described in chapter 4, this was the main goal of that profile. In future research these results could be further analysed to see how big the environmental impact of such changes is. The “consistent” profile was supposed to reduce the number of route changes and changes to existing trajectories. Both values were reduced in the tests, albeit only by a small amount. It may be possible to reduce these values further by increasing the corresponding penalties even more, but that might reduce the quality of the solutions of the genetic algorithm. Lastly the “earliest” profile was able to save an average of approximately 6 seconds of the average taxi duration of the flight. This might not be a lot, but the trajectories to which these numbers are compared were already highly optimized, using the standard configuration.

6. Conclusion and Future Work

In this work the prototypical SMAN TraMICS+ was presented and the algorithms for the calculation of conflict free, realistic ground trajectories were introduced. These trajectories are automatically adapted to changed conditions and their usability was pre-evaluated in human-in-the-loop simulations. It was shown that the chosen algorithms were suitable for a real-time-system, since the calculation or adaption of any trajectory took on average less than half a second, even in a very complex traffic scenario. The presented algorithms can be configured in many ways leading to different solutions. For our setting a well-fitted configuration for the algorithms was chosen. In simulations with pre-recorded traffic it was verified that different profiles (i.e. parameter settings for the penalty function with different focus) lead to solutions with the desired substantial differences.
Although the proposed algorithms are well suited, configured and tuned there is still potential for

future work. For example, some parts of the penalty function could be further refined: The penalty values multiplied with W2 (increased taxi duration compared to initial trajectory), W5 (number of fast points) and W6 (number of slow points) are currently only rough approximations of their respective metrics. The term for W7 (number of changed existing trajectories) could also be further refined by differentiating between flights that already received clearances from the controller and flights whose trajectories were not yet approved by the controller. By fine tuning the penalty profiles it is most probably possible to find a parameter setting that lead to even better results than in the here performed more general experiments as well.
The crossover function in the implemented standard version did not lead to any improvements during testing and was therefore disabled. In classic genetic algorithms, crossover is a very important component and has a big positive impact on the optimization. So some more work will be done to improve the crossover function in a more problem specific way and make it more beneficial for the overall algorithm. A possible approach to refine it could be to consider known conflicts when selecting a position at which the two parents are merged.
In the current implementation of TraMICS+, the trajectories for departures are based on the EOBT/ TOBT only. No runway scheduling is performed yet and no take off time is considered. The focus for the studies presented here was laid on optimal trajectories for TraMICS+. This can lead to line-up queues in case the runways are occupied when the departures arrive there. In the future TraMICS+ should either be coupled with a Departure Manager (DMAN) or be enhanced to contain some DMAN functionalities. This would enable the calculation of trajectories in such a way that an optimized departure sequence is adhered to, similar to [3]. It is expected, that this would further improve environmentally friendliness, because runway queues could be better prevented.
To evaluate TraMICS+ operational usability validation trials with air traffic controllers will be performed in June and July 2022.
A precise fuel calculation will be implemented in the research project GreAT [14], which could be beneficial to TraMICS+, as the profiles or even the penalty function itself could be enhanced.
TraMICS+ and with it the described algorithms may serve as a basic component for future research on full automation on ground, which requires a reliable, fast and feasible calculation and adaptation of trajectories.

The project on which the presented research is based is funded by the German Federal Ministry of Education and Research (BMBF) as part of the call „Zivile Sicherheit – Kritische Strukturen und Prozesse in Produktion und Logistik“ with the funding number 13N15104. The responsibility for the content of this publication lies with the authors.

7. Contact Author Email Address

Lennard.Noehren@DLR.de
Meilin.Schaper@DLR.de
Lukas.Tyburzy@DLR.de
Kathleen.Muth@DLR.de

8. Copyright Statement

References

[1] Eurocontrol. *Seven-Year Forecast 2021 – 2027 Main report*. October 2021.
[2] Eurocontrol. *European Aviation in 2040 Challenges of Growth – Annex 4 Network congestion*. 2018.
[3] Schaper M und Gerdes I. Trajectory Based Ground Operations And Their Coordination With Departure Management. *32nd DASC*, Syracuse, NY, USA, October 2013.

[4] Gerdes I and Schaper M. Management of Time Based Taxi Trajectories coupling Departure and Surface Management Systems. *11th ATM Seminar 2015*, Lissabon, Portugal, 2015.

[5] Atkin J, Burke E and Ravizza S. The Airport Ground Movement Problem: Past and Current Research and Future Directions. *4th International Conference on Research in Air Transportation*, Budapest, pp 131-138, 2010.

[6] Clare G and Richards A. Optimization of Taxiway Routing and Runway Scheduling. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*, Vol. 12, No. 4, pp 1000-1013, December 2011.

[7] Gerdes I and Temme A. Taxi routing for aircraft: Creation and Controlling. *Second SESAR Innovation Days*, November 2012.

[8] Pesic B, Durand N and Alliot J. Aircraft ground traffic optimisation using a genetic algorithm. *GECCO 2001, Genetic and Evolutionary Computation Conference*, San Francisco, United States, July 2001.

[9] Schaper M, Gluchshenko O, Muth K, Tyburzy L, Rusko M and Trnka M. The Traffic Management Intrusion and Compliance System as Security Situation Assessment System at an Air Traffic Controller's Working Position. *31st European Safety and Reliability Conference*, pp 2825-2831, 2013.

[10] NLR. *NARSIM*. `www.narsim.org`. accessed 20.05.2022.

[11] Hamburg Airport. *Verkehrszahlen 2018: Luftverkehr wird immer effizienter*. `www.hamburg-airport.de/de/unternehmen/presse/pressemitteilungen/archiv/verkehrszahlen-luftverkehr-2018-1282`. accessed 20.05.2022.

[12] Eurocontrol. *Advanced Emission Model*. `www.eurocontrol.int/model/advanced-emission-model`. accessed 20.05.2022

[13] Intel. *Intel Core i7-8750H Processor*. `www.intel.com/content/www/us/en/products/sku/134906/intel-core-i78750h-processor-9m-cache-up-to-4-10-ghz/specifications.html`. accessed 20.05.2022

[14] GreAT. *Greener Air Traffic Operations*. `www.project-great.eu/`. accessed 20.05.2022.