A Fleet Management Algorithm for Automatic Taxi Operations

Giuseppe Sirigu
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Turin, Italy
giuseppe.sirigu@polito.it

Manuela Battipede, Piero Gili
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Turin, Italy
manuela.battipede@polito.it
piero.gili@polito.it

John-Paul Clarke
Daniel Guggenheim School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA, USA
johnpaul@gatech.edu

***Abstract*— The continuous growth of air traffic has resulted in congestion and long queues at airports, causing delays, pollution and loss of money for the airlines. In this paper, we present a new solution to perform just in time taxi operations using autonomous electric towbarless tractors. The purpose of this solution is to eliminate queues and to reduce the environmental and economic impact of ground operations, meeting the requirements for the future air traffic management (ATM). An algorithm for a tool that provides conflict-free schedules for the tractor autopilots will be presented; the generated schedule is meant to minimize the overall cost of the ground operations. The proposed algorithm is based on a hybrid particle swarm optimization (HPSO), hybridized with a hill-climb meta-heuristic, that defines an optimal set of path and speed for each flight in the flight schedule. The effectiveness of the proposed HPSO algorithm compared to the classical particle swarm optimization (PSO) was studied. Results showed that the proposed algorithm is more effective with respect to the classical PSO, even though the required computational time drastically increases.**

Keywords-Taxiing; Airport operations; Hybrid PSO; Particle swarm optimization; Hill-climbing; Combinatorial optimization

I. INTRODUCTION

Civil aviation is experiencing continuous growth in both passenger and cargo demand. The resulting congestion and long queues at airports (i.e. on taxiways and ramps) induce airplanes to burn an additional amount of fuel during ground operations, thereby causing pollution, noise and increasing costs to airlines.

An alternative solution to this problem is proposed and uses a fleet of autonomous electric towbarless tractors to move the aircraft between gates and runways, allowing the aircraft to maintain the engines turned off. Previous researches showed that this solution effectively reduces the environmental impact during the taxi [1]–[4].

The related PhD project introduces a futuristic approach for the future air traffic management: the aircraft will be delivered just in time and the taxi-in will start as soon as the aircraft lands. This allows to eliminate both outgoing and incoming queues at the runways. The aim of the PhD project is to define an algorithm for a tool that provides conflict-free schedules for the tractor autopilots that people upstream can meet, and people downstream can rely on. The generated schedule is meant to minimize the overall cost of the ground operations.

The proposed fleet management tool is composed of two main elements: a trajectory optimization algorithm and a task assignment algorithm. The purpose of the trajectory assignment algorithm is to select a combination of path and speed for each flight in the schedule, in order to minimize a cost function. The speed is conceived to be a discrete variable, thereby a combinatorial optimization problem must be solved. The task assignment algorithm takes as input the combinations of path and speed for each flight and allocates a tractor to each flight considering their state of charge (SoC). The objective of the task assignment algorithm is to balance the utilization of the tractors. Also in this case a combinatorial optimization problem has to be solved.

In literature, several algorithms have been proposed for combinatorial optimization for both convex and non-convex problems [5], [6]. In this paper, we propose a hybrid particle swarm optimization; a hill-climb meta-heuristic was used to hybridize the PSO. Kennedy and Eberhart were inspired by the social behavior of some animals, like bird flocking, to create the PSO [7]. In this population based algorithm, the individuals composing the swarm cooperate to find out the best solution in the problem search space. Each particle benefits of its experience (*pbest*) and of the swarm's one (*lbest* or *gbest*); *pbest* represents the best solution found by each particle so far. The swarm can be divided in neighborhoods of defined size (usually 15% of the entire swarm); in this case, each particle can move towards the best neighbor previous position *lbest* or can be headed for the actual best neighbor [8]. A different approach considers the best solution *gbest* found so far by any particle in the whole swarm; the global solution can be seen as a special case of the local one, where the neighborhood size equals the swarm size.

This paper provides a general overview of the overall system and the remainder of the paper is organized as follows: in (II), the fleet management system architecture is presented. The proposed hybrid particle swarm optimization for trajectory assignment purpose is introduced in (III), and the results for a test case are presented in (IV). Conclusions and future work are reported in the closing section.

II. SYSTEM ARCHITECTURE

The fleet management tool proposed in this paper has been designed to provide conflict-free schedules for automated tractors which minimize the total cost of the taxi operations. The tool was conceived to satisfy the following goals:

- Reconfigurability: the algorithm had to be easily adapted to different airports; respond quickly to variations in the flight schedule due to flight delays or weather changes;
- Reliability: the schedule must always be free of conflicts between the agents;
- The computational time required to process a time horizon of 1 hour had to be less than 15 minutes.

The structure of the fleet management system is reported in Fig. 1. In order to properly work, the algorithm requires a set of inputs specific for each airport, and also an airplane database containing the main features of the airplanes involved in the flight schedule (II.A). These data are used within the trajectory assignment to generate combinations of path and speed for each flight (II.B). Once the non-conflicting trajectories are assigned to each flight, the task assignment algorithm generates the schedules for the tractor autopilots (II.C). The system level is not analyzed in this paper and will be object of a second phase of the PhD project.

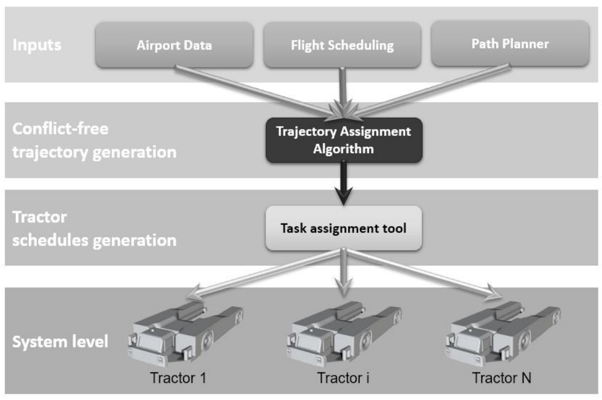

Figure 1. Fleet management system architecture.

A. *Inputs*

Three input sets have to be provided to the fleet management algorithm: the flight schedule, the airport data and the paths identified for each flight by means of a path planning algorithm. The airport data contain the tractor fleet composition in terms of number and features of the tractors. Furthermore, a discretization in waypoints of the airport is required by the path planning algorithm; a graphical user interface (GUI) was developed to simplify the discretization. It was realized in the MATLAB 8.1 environment and exploits the built-in *getpoint* function to get the waypoint location from the airport map. The GUI enables the user to define also the aircraft parking lots, the runway entry labels and the taxiway directions.

From the tractor point of view, each taxi mission has three phases: a central towing phase, in which the tractor moves the airplane between gate and runway and two repositioning phases, where it moves between the depot and the aircraft. A modified version of the Breadth-first search (BFS) algorithm was implemented; the proposed algorithm does not take into account the distance between the nodes during the path construction and provides paths visiting each node exactly once. This algorithm finds all the possible paths between two locations in the airport, in order to allow the trajectory assignment algorithm to choose a path different from the shortest one, if the latter is not available.

B. *Conflict-free trajectory generation*

In order to find the optimal set of non-conflicting trajectories, a combinatorial optimization problem had to be solved. A HPSO was implemented using a hill-climbing meta-heuristic to hybridize the PSO. For each particle in the swarm, the fitness value is a combination of a cost function and a penalty function taking into account the number of conflicts detected for that particular solution. Further details about the trajectory assignment algorithm will be provided in III.

C. *Tractor schedule generator*

The schedule for each tractor is built from the solution of the trajectory assignment algorithm by solving a combinatorial problem. The allocation of resources (tractor) to each activity (flight) will be based on the availability of the tractor and on its state of charge; the objective of the combinatorial optimization will be to assign a tractor to all the flights, balancing the utilization of the tractors among the whole fleet. The task assignment algorithm can be designed as an independent algorithm or it is possible to merge it with the trajectory assignment algorithm into a single algorithm.

III. Trajectory Assignment Algorithm

The search domain of the present problem is non-convex; therefore, linear programming techniques (e.g. Simplex) cannot be used. Heuristics are then required to find an optimal or near-optimal solution in a limited computational time. This section presents the details of the proposed HPSO and introduces the cost evaluation model and the problem constraints (i.e. conflict detection, gate utilization constraint).

A. *Hybryd Particle Swarm Optimization for Taxiing Problem*

Each particle of the swarm represents a position in the search space, which corresponds to a solution of the problem. A randomly generated family of *K* particles of *n* elements each and their velocities are used in order to spread as much as possible the particles among the search space, and to increase the probability of reaching the optimal solution. Once the fitness value for each particle in the swarm is computed, the *pbest* and *gbest* positions are determined. The velocity $v_{k,j}$ of each element *j* of the particle x_k are randomly generated, as described in (1).

$$v_{k,j} = W \cdot x_{k,j} + c_1 \cdot rand_1 \cdot (pbest_{k,j} - x_{k,j}) + c2 \cdot rand_2 \square (gbest_j - x_{k,j}) \quad (1)$$

The particle position is updated adding the correspondent velocity to each element of the particle. The algorithm pseudo-code is reported in Fig. 2; the *Count* variable is defined as the number of iterations without improvements in the *gbest* position. The *MaxIter* and *MaxCount* parameters are set by the user; a proper tuning of these values is fundamental to allow the algorithm finding an optimal or near-optimal solution, while preventing it from running for an unnecessary long time.

```
Generate $K$ random particles;
Initialize velocities $v_{kj}$ between 0 and 1;
while $Iter < MaxIter \wedge Count < MaxCount$ do
    Evaluate the fitness for each particle;
    Determine the best solution $pbest$ visited so far;
    Determine the global best solution $gbest$ visited so far;
    Compute the particle speeds;
    Update the particle position;
    Improve the solution by means of the hill-climbing heuristic;
end
```

Figure 2. Hybrid particle swarm optimization pseudocode.

At the end of each iteration, the algorithm improves each particle position by means of a hill-climbing heuristic (Fig. 3). For each particle, *n*/2 random indexes are extracted between 1 and *n*; they represent the elements of the particle *k* to be analyzed by the hill-climbing. The number of selected elements was chosen as a trade-off between the effectiveness of the approach and the required computational time. As the elements to be analyzed are chosen in random order, a broader search of the space around the actual particle position is granted, without a predefined search direction. The value of each examined element is sequentially substituted by each of the other possible values for that element, while the remaining *n*–1 are fixed. The modified particle characterized by the maximum fitness value replaces the original one.

```
for $k = 1 : K$ do
    Generate $\frac{n}{2}$ random integer numbers $idx_j$ within the $[1, n]$ set;
    for $j = 1 : \frac{n}{2}$ do
        Replace $x_{k,idx_j}$ with one of its possible values;
        Evaluate the fitness function for the modified $x_k$;
        if $new\_fitness_k > fitness_k$ then
            $x_k = x\_new_k$;
        end
        Repeat the steps above until all the possible values of $x_{k,idx_j}$ have
        been examined;
    end
end
```

Figure 3. Hill-climbing algorithm pseudocde.

Considering *NF* flights in the schedule, for each phase of the trajectory, the particles are composed as follows (Fig. 4):

- NF elements (in blue) represent the path index assigned to the phase *j* of the flight *i*, which is chosen among the all possible paths for that flight;
- NF elements (in red) contain the speeds associated to each path.

X =	Path Phase j Flight 1	Path Phase j Flight i	Path Phase j Flight NF	Speed Phase j Flight 1	Speed Phase j Flight i	Speed Phase j Flight NF

Figure 4. Particle structure.

The fitness value of each particle is computed as the reciprocal of the linear combination of the cost function and the penalty function (2). Further details on the terms composing the fitness function are provided in the remainder of this section.

$$f = 1/(k_1 \cdot C_{tot} + k_2 \cdot \; penalty \;) \tag{2}$$

B. *Cost Evaluation Model*

The total cost of the taxi operations was defined as the sum of the time related costs C_T and the costs related to the energy consumption C_E (3). The use of both the cost sources allows to balance between reducing the energy consumption and the time required to perform the taxi operations.

$$C_{tot} = C_T + C_E \tag{3}$$

The time related costs are modeled as a linear function of the taxi time (4), where *TRC* represents the hourly crew cost and was set to 550 $/h, based on the data available in [9], [10].

$$C_T = TRC \cdot t \tag{4}$$

The proposed approach uses electric propelled tractors to tow the airplanes. The mechanical power required to move the tractor at the speed V was computed considering the forces acting on the tractor [11]–[13]: the aerodynamic drag Fa (5), the rolling friction Fr (6), the inertia Fi (7) and the slope resistance Fs (8).

$$F_a = 1/2 \cdot \rho \cdot V^2 \cdot S \cdot C_D \tag{5}$$

$$F_r = m \cdot g \cdot f_r \cdot \cos(\alpha) \tag{6}$$

$$F_i = m \cdot a \tag{7}$$

$$F_s = m \cdot g \cdot \sin(\alpha) \tag{8}$$

The slope of the taxiways must be $\alpha \leq 3\%$ [14]; thus, the slope resistance can be neglected. Considering the tractor acceleration/deceleration a and the friction coefficient f_r, it is possible to compute the required mechanical power:

$$P = (F_a + F_r + F_i) \cdot V. \tag{9}$$

Each phase of the tractor trajectory was divided in three segments: an acceleration segment, a constant speed segment and a deceleration segment.

The energy consumption during accelerations and decelerations was computed by using a continuous model; this model considers also braking energy regeneration features. At each time step $\Delta t = 10^{-3}$s, the energy variation is computed as reported in (10). The total energy consumption is obtained integrating the value ΔE^{bat} for the acceleration/deceleration time.

$$\Delta E^{bat} = \begin{cases} \dfrac{\Delta t}{\eta_d \cdot \eta_{out} \cdot \eta_{EM} \cdot 3600} \cdot P & if\ P \geq 0 \\[2ex] \dfrac{\Delta t \cdot \eta_c \cdot \eta_{in} \cdot \eta_{EM} \cdot e_{rec}}{3600} \cdot P & if\ P < 0 \end{cases} \; . \tag{10}$$

The parameters η_d, η_c, η_{out}, η_{in} and η_{EM} in (10) are respectively the energy storage discharging/charging efficiency, the discharging/charging efficiency of the power electronics and the electrical motor efficiency. The coefficient e_{rec} represents the fraction of braking energy recovered.

The computation of the energy consumed during accelerations and decelerations would drastically increase the computational time required for the optimization; thus, a database containing the acceleration and deceleration time, space and energy consumed or regenerated was created. This database considers tractors with different specifications, which can be unloaded (the tractor is not towing any airplane) or loaded (the tractor is towing a specific airplane). In the loaded case, the characteristics of different airplanes are considered to compute the forces acting on the tractor for both take-off and landing.

Conversely, a piecewise-linear model function of the tractor speed was considered during the constant speed segments of length D_{const}. It is possible to write the mechanical power as a function of the speed:

$$P \propto P_1 \cdot V^3 + P_2 \cdot V. \tag{11}$$

P_1 and P_2 are stated respectively in (12) and (13).

$$P_1 = 1/2 \cdot \rho \cdot S \cdot C_D \tag{12}$$

$$P_2 = m \cdot g \cdot f_r \tag{13}$$

Considering a constant speed, it is possible to write the total travel time as $t = D_{const}/V$. The total energy consumed during this segment is:

$$E^{bat}(V) = E^{bat}(V - \Delta V) + \frac{dE^{bat}}{dV}\Delta V = \frac{D_{const}}{\eta_{EM} \cdot \eta_{out} \cdot \eta_d \cdot 3600}\{[P_1 \cdot (V - \Delta V)^2 + P_2] + 2 \cdot P_1 \cdot V \cdot \Delta V\} \tag{14}$$

C. *Conflict Detection*

A penalty was added to the fitness function to take into account the possible conflicts arising between the tractors and the constraint violations; an elliptical threat detection area (TDA) was defined around the tractor with the major axis aligned to the tractor speed (Fig. 5). The choice of an elliptical TDA is motivated by the fact that if a round TDA had been considered around each tractor, conflicts might be identified despite two tractors were traveling on parallel taxiways.

The minor semi-axis b, perpendicular to the taxiway centerline, is constant and defined in (15).

$$b = \begin{cases} w_{tractor} \cdot k & if\ unloaded \\ (w_{airplane} + \varepsilon) \cdot k & if\ loaded \end{cases} \tag{15}$$

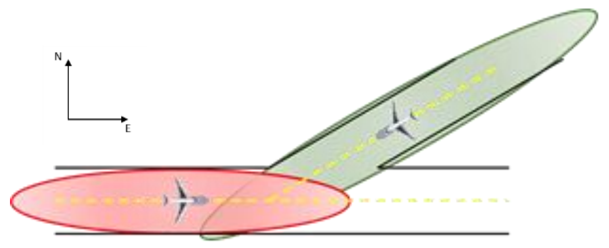

Figure 5. Elliptic threat detection area scheme.

$w_{tractor}$ and $w_{airplane}$ are respectively the tractor width and the airplane semi-wingspan; ε = 7m is the maximum tip clearance between a parked aircraft and any other object in the airport, as defined by the ICAO [14]. A safety factor k = 1.5 was also introduced.

The major semi-axis c value is composed of two parts: a constant term depending on the tractor/airplane length, and a variable term depending on the tractor speed (16).

$$c = \begin{cases} \left(l_{tractor} + \dfrac{V^2}{2 \cdot a}\right) \cdot k & if\ unloaded \\[2ex] \left(l_{airplane} + \dfrac{V^2}{2 \cdot a}\right) \cdot k & if\ loaded \end{cases} \tag{16}$$

A conflict is detected when the TDAs of two tractors are overlapped; a unit value is added to the *penalty* variable.

D. Gate utilization constraint

The gate utilization constraint imposes that the ground operations comply with the airport regulation on gate occupancy. The trajectories have to be optimized to prevent the airplanes occupying the gates beyond the maximum allowed gate occupancy (MGO), while completing the turnaround procedure. Moreover, both taxi-in and taxi-out trajectories are subject to the constraint specifying the minimum separation between two aircraft at the gate (MSG). In the present application, the values MGO = 2h, MSG = 15min were used [15]–[17].

IV. NUMERICAL RESULTS

This section presents the results of a convergence analysis of the trajectory assignment algorithm, performed on a test case in the Turin airport “Sandro Pertini”. The algorithm was implemented in C language and executed on a Windows 10 Pro platform supported by an Intel Core i7-4810MQ CPU and 8 GB RAM.

A. Test Case

The flight schedule used for the convergence analysis was extracted from a real flight schedule in the Turin airport and it aims at testing the algorithm for the computation horizon of one hour. The assigned gates and runway were defined by the authors as real data were not available for these parameters (Table 1). The total combinations of path and speed for the proposed test case is:

$$comb = \prod_{i=1}^{NF}\left(\prod_{j=1}^{3}\left(speeds_{i,j}\cdot k_{i,j}\right)\right) = 5.6843\cdot 10^{34}. \quad (17)$$

TABLE 1. TEST CASE: FLIGHT SCHEDULE

TO /LND	Flight	Dest/ Orig	Dep/Arr Time	Parking Lot	A\C	Rwy Entry
TO	LH 297	FRA	11:05:00	103	735	F
LND	KL 1555	AMS	11:10:00	111	F70	B
LND	AZ 1413	FCO	11:25:00	112	320	A
LND	FR 8628	TPS	11:30:00	113	320	C
TO	TK 1310	IST	11:35:00	114	73W	B
LND	VY 6514	BCN	11:40:00	114	320	E
TO	KL 1556	AMS	11:50:00	111	F70	G
TO	FR 8629	TPS	11:55:00	504	F70	F

The HPSO parameters are reported in Table 2. The optimal solution for the test case is reported in Table 3; the optimal total cost is C_{TOT} = 830.20 $.

TABLE 2. HPSO PARAMETERS

MaxIter	*MaxCount*	*W*	c_1	c_2	k_1	k_2
200	20	0.9	1	1	1	1000000

TABLE 3. TEST CASE: OPTIMAL SOLUTION

	Fl. 1	Fl. 2	Fl. 3	Fl. 4	Fl. 5	Fl. 6	Fl. 7	Fl. 8
Path 1	1	1	1	1	1	1	1	1
V 1	12.5	16	16	12.5	9.5	12.5	9.5	12.5
Path 2	1	2	2	2	1	2	1	1
V 2	11	16	16	12	11	14.5	8.5	11
Path 3	1	1	1	1	1	1	1	1
V 3	11	10.5	10.5	10.5	11	10.5	9.5	6.5

B. Convergence analysis

A converge analysis was carried out after 200 runs for different values of the swarm size *K*. The rate of convergence to the optimum for both algorithms is depicted in Fig. 6. It can be noticed that the PSO never finds the optimal solution, whereas for the HPSO $P(f_{opt}) \geq 70\%$.

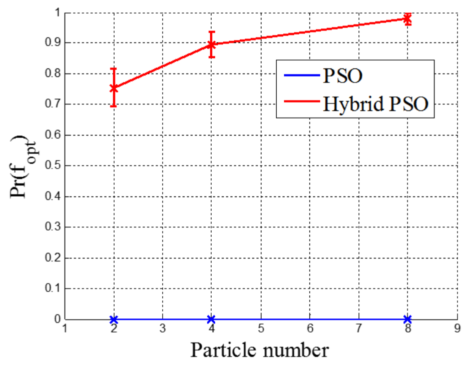

Figure 6. Convergence rate results.

The computational time required by the HPSO is considerably higher with respect to the one required using the PSO (Fig. 7); furthermore, it exceeds the computational time target of 15min for $K > 2$.

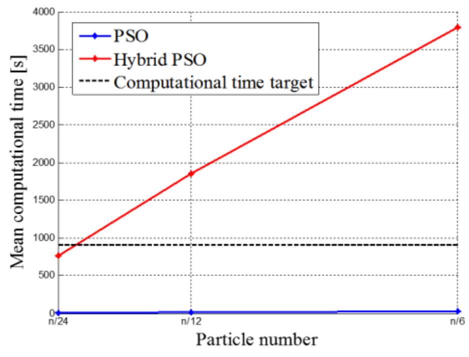

Figure 7. Mean computational time comparison.

V. CONCLUSIONS AND FUTURE WORK

A new solution to perform taxi operations using autonomous electric tractors was proposed; this solution aims at reducing the environmental and economic impact of the ground operations. The development of a fleet management algorithm is required to provide conflict-free schedules to the tractor autopilots.

In this paper, we presented a trajectory assignment algorithm, based on a hybrid particle swarm optimization; a modified version of the hill-climb metaheuristic was implemented to hybridize the PSO and improve the algorithm effectiveness. The cost arising from the trajectory assigned to each flight is composed of two terms: the energy consumed to perform the trajectory and the crew cost for the airlines.

A convergence analysis for different values of the particle number *K* was realized and the performance of the proposed HPSO were compared to the classical PSO in terms of probability of finding the optimum solution, and mean computational time. The analysis showed that the HPSO has a rate of convergence to the optimum significantly higher than the classical PSO for all the values of *K*. However, the HPSO requires considerably higher computational time to reach the optimum solution.

Future work will focus on the parallelization of the code, in order to speed up the computation. Furthermore, the task assignment algorithm will be developed and the proposed approach will be extended to different airports

REFERENCES

[1] M. Battipede, A. Della Corte, M. Vazzola, and D. Tancredi, “Innovative Airplane Ground Handling System for Green Operations,” in 27Th International Congress of the Aeronautical Sciences, 2010.

[2] K. Aktay, K. Aktay, and J. Y. Kim, “ANTS – Automated NextGen Taxi System,” 2009.

[3] R. Morris, M. L. Chang, R. Archer, E. V. C. Ii, S. Thompson, J. L. Franke, R. C. Garrett, K. Mcguire, and G. Hemann, “Self-Driving Aircraft Towing Vehicles : A Preliminary Report The Case for Autonomous Towing,” pp. 41–48, 2015.

[4] Lufthansa, “Innovative TaxiBot now used in real flight operations,” 2015. [Online]. Available: www.lufthansagroup.com/en/press/news-releases/singleview/archive/2015/february/20/article/3439.html. [Accessed: 18-Jan-2016].

[5] C. H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity. Courier Corporation, 1998.

[6] F. S. Hillier and G. J. Lieberman, Introduction to Operations Research, 9e ed. McGraw-Hill Higher Education, 2010.

[7] R. C. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in Proceedings of the sixth international symposium on micro machine and human science, 1995.

[8] M. Clerc, “Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem,” 2004.

[9] Group Transport Studies University of WestminsterLondon, “Aircraft crewing – marginal delay costs,” 2008.

[10] AIRBUS, “Getting to grips with the cost index,” 1998.

[11] J. Van Roy, N. Leemput, S. De Breucker, F. Geth, P. Tant, and J. Driesen, “An Availability Analysis and Energy Consumption Model for a Flemish Fleet of Electric Vehicles,” Eevc, pp. 1–12, 2011.

[12] J. Larminie and J. Lowry, Electric Vehicle Technology Explained Electric Vehicle Technology Explained. 2012.

[13] R. Maia, M. Silva, R. Araujo, and U. Nunes, “Electric vehicle simulator for energy consumption studies in electric mobility systems,” 2011 IEEE Forum Integr. Sustain. Transp. Syst. FISTS 2011, pp. 227–232, 2011.

[14] International Civil Aviation Organization, Aerodromes, Annex 14 to the Convention on International Civil Aviation, vol. I, no. July. ICAO, 2013.

[15] Denver International Airport, “Rules & Regulations Governing DEN,” 2010. [Online]. Available: www.flydenver.com/about/administration/rules_regulations. [Accessed: 11-Feb-2016].

[16] Oakland International Airport, “Oakland International Airport Gate Rules and Procedures,” 2006.

[17] Greater Orlando Aviation Authority, “Airlines operations procedures,” 2003.