2023 International Conference on Unmanned Aircraft Systems (ICUAS)
June 6-9, 2023 | Lazarski University, Warsaw, Poland

Wind-Aware Path Optimization for an Aerobot in the Atmosphere of Venus Using Genetic Algorithms

Anna Puigvert I Juan, Bernardo Martinez Rocamora Jr., and Guilherme A. S. Pereira

***Abstract*— This paper presents a path optimization solution for an autonomous aerial robot (aerobot) in the windy atmosphere of Venus. The aircraft is required to travel from its current position to a goal position by following minimum energy paths. The approach proposed in this paper uses genetic algorithms, a heuristic search that, based on a population of initially feasible paths and a set of biologically inspired operations, finds a low-cost path. The proposed cost function accounts for energy expenditure, such as thrust or drag, and also energy accumulation, such as charging with solar panels and gains from potential energy (e.g., due to upward directional winds). Path feasibility is assured by computing local reachability regions based on the wind velocity and the maximum speed of the aerobot. The method is illustrated through a series of simulations that show our results as a function of the number of iterations and path population sizes. A comparison with a previous algorithm is also made.**

I. INTRODUCTION

Venus has geophysical characteristics comparable to those on Earth, such as composition, gravity, mass, and radius. Researchers believe that Venus, two billion years ago, could have had oceans and a livable atmosphere, similar to Earth's [1]. Past explorations in Venus concluded that the planet has changed drastically due to the greenhouse effect, which may have turned its surface into a hot and inhospitable place [2]. Venus has a surface temperature around $450\,^{\circ}\mathrm{C}$, more than the melting point of lead, and a pressure around $93\,\mathrm{bar}$ [3], which complicates the deployment of scientific missions in the surface and lower atmosphere.

However, studying Venus is valuable because it can extend the understanding of the planetary and climate evolution of our own planet Earth. One of the missions proposed to study Venus consists of deploying a semi-buoyant unmanned propelled aerobot [4]. It will have an aeroshell-less hypersonic entry and will then transition to a semi-buoyant, maneuverable, solar-powered aerobot that will fly between $50\,\mathrm{km}$ and $70\,\mathrm{km}$ of altitude above the Venusian surface, where the atmospheric conditions are similar to Earth's [5] and where scientists found evidence of the presence of phosphine, a substance that suggests the existence of organic life-forms [6], [7].

This research was made possible with support from the NASA Established Program to Stimulate Competitive Research, Grant #WV-80NSSC21M0145, and the Benjamin M. Statler Fellowship.

The authors are with the Department of Mechanical and Aerospace Engineering, Benjamin M. Statler College of Engineering and Mineral Resources, West Virginia University. Morgantown, WV, 26501 USA.
Emails: ap00063@mix.wvu.edu, bm00002@mix.wvu.edu, guilherme.pereira@mail.wvu.edu

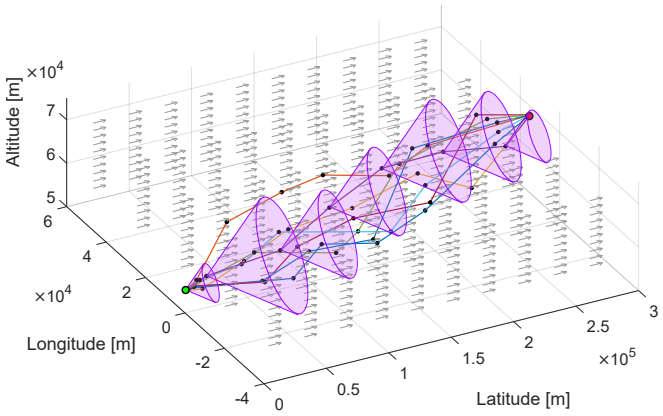

Fig. 1: Initial random population of paths used by the proposed genetic algorithm path planner. Each color represents a path and black dots are the waypoints points of each path. The green and red dots are the start and goal positions, respectively. Purple cones show the feasible regions of each waypoint of the purple path. This is a function of the wind (gray arrows) and the vehicle speed. Feasible paths are paths where each waypoint is inside the feasibility region of the previous one.

Our goal is to develop an optimal motion planner for an aerobot in Venus that 1) creates minimum energy cost paths, that allow the vehicle to fly for long periods of time without running out of battery; 2) accounts for in-flux and out-flux of the battery to determine battery level; and 3) accounts for the strong winds present in the area of exploration, which is faster than the aerobot's maximum airspeed.

Some previous works solved similar problems on earth. In [8], [9], a tree-based kino-dynamic motion planner considering the wind was developed for a fixed-wing aircraft with temporal and spatial changes in the wind field. This work did not consider energy expenditure with the propulsive system or any sort of charging with solar panels. Additionally, the authors assume that the aircraft's velocity relative to the wind is larger than the wind velocities, which is not the case in our problem. In our previous work, we proposed a sampling-based planner to find energy-efficient paths to navigate in the atmosphere of Venus [10], where the wind velocities cannot be compensated by the aircraft. However, the winds were only considered indirectly, deforming the local Dubins paths between nodes of the graph constructed. While guaranteeing feasibility, our previous work was not able to find optimal paths, which motivated the current work. Besides searching for optimal paths, the Genetic Algorithm method used in

this work resulted in a more efficient and easy-to-implement algorithm, when compared to [8], [9] and [10].

In this paper, we present a sampling-based genetic algorithm path planner for a fixed-wing aerobot under the influence of the strong winds of Venus. The motion planner optimizes a cost function (fitness function) based on in-flux and out-flux of energy. This cost function, proposed in our previous work [10], accounts for the expenditure of energy, such as thrust or drag, but also energy accumulation, such as charging with solar panels and gains from potential energy (e.g., upward directional winds). It is important to note that the naive inclusion of such in-flow energy might result in negative costs when calculating the energy cost of the path, which would contradict most optimizers' criteria [11]. We then employ the concept of "opportunity cost" [12] to account for the loss of gain generated by failing to pursue the most beneficial path. In order to calculate the energy cost from the solar panels, we account for the altitude that the aerobot is flying, being $70\,\mathrm{km}$ above the surface the maximum gain of energy in our environment and $55\,\mathrm{km}$ the less favorable altitude, where the different cloud layers attenuate the solar intensity [13]. Besides finding low-cost paths in three dimensions (3D), our method also guarantees the feasibility of the paths by using the wind speed and the maximum aerobot velocity to compute local feasibility regions for each waypoint of the path. An illustration of the path feasibility strategy used in the paper is shown in Fig. 1.

Therefore, the main contributions of this paper are:

1) A genetic algorithm-based path planner that finds energy-efficient paths for an aerobot that i) accounts for path constraints due to wind, ii) includes the energy model of the semi-buoyant aerobot, and iii) includes Venusian atmospheric conditions.
2) A method for generating feasible paths in 3D windy environments using random sampling inside the reachable region defined by the local wind vector and the vehicle's maximum airspeed.

The rest of this paper is organized as follows. Problem definition and background are presented in Sect. II and III, respectively. Our heuristic approach is explained in Sect. IV. Then, numerical results are presented in Sect. V. Finally, conclusions and future work are presented in Sect. VI.

II. PROBLEM DEFINITION

This section provides the information that is relevant to describe the problem solved in this paper. A genetic algorithm (GA) was developed to find an optimal path for an aerobot in the atmosphere of Venus. We provide environmental and vehicle models that are necessary to constrain the algorithm.

A. Problem Statement

In this work, the desire is to move an aerobot from the start position, p_{start}, to the goal position, p_{goal}, in the atmosphere of Venus, where strong winds are present, with the lowest energy consumption possible while avoiding running out of battery power. Each coordinate of the aerobot is represented in the three-dimensional space by (x, y, z), where x and y represent latitude and longitude respectively and z represents the altitude. Specific constraints and assumptions about the environment and the aerobot are discussed in the next subsections.

B. Environment

The flyable region of exploration for an aerobot in the Venusian atmosphere is within $55\,\mathrm{km}$ and $70\,\mathrm{km}$ of altitude with respect to its surface. In this region, Venus has similar climate conditions as the ones on Earth. The air density, $\rho(z)$, gravitational acceleration, $g(z)$, pressure, $p(z)$, temperature, $\Theta(z)$, and wind field, $\mathbf{w}(x, y, z)$, can be obtained from data tables presented in [10]. These tables provide simplified models of the atmosphere, using data from several past missions to Venus [3], [14]. For simplicity, all the variables mentioned before are assumed to be time-invariant. Furthermore, the wind field is assumed to be dominated by the super-rotation winds, compatible with the equatorial region observations [15]. As a summary, two main points are important in the Venusian atmosphere: (i) the farther from the surface the lower the density, temperature, and pressure; (ii) the strong winds range from $61 \pm 25\,\mathrm{m/s}$ at lower altitudes to $94 \pm 30\,\mathrm{m/s}$ at higher altitude. Graphs and tables illustrating this behavior can be found in our previous paper [10]. The last environmental factor included in our problem is the solar intensity ($I_{solar}(z)$) which provides power to the aerobot by charging the batteries. Venus provides a solar flux almost double as Earth's, being its exoatmospheric solar flux $2600\,\mathrm{W/m^2}$. However, due to the dense clouds present in the exploration region, the available solar intensity varies with the altitude. The ratio of available solar intensity is 95% at the higher flyable region ($65\,\mathrm{km}$) and between 20 and 50% at lower altitudes ($40\,\mathrm{km}$). A model for the solar intensity in the Venusian atmosphere is also shown in [10].

C. Aerobot

This paper considers an aerobot based on the concept proposed in [4]. A 3D visualization of this concept is shown in Fig. 2. The vehicle is a semi-buoyant unmanned propelled

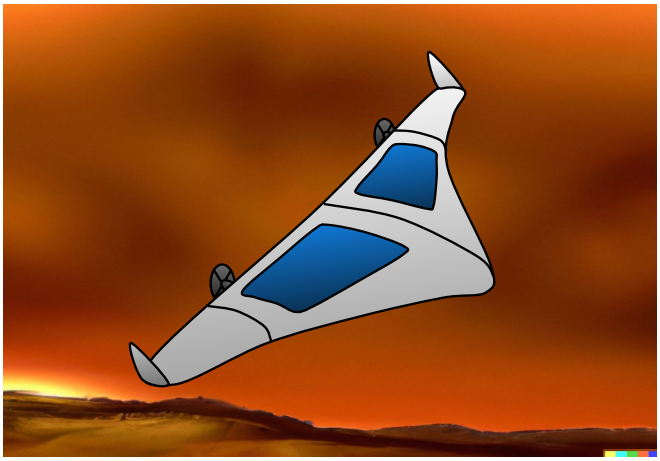

Fig. 2: Aerobot deployed in the Venus clouds. The background of this image was created with the assistance of AI.

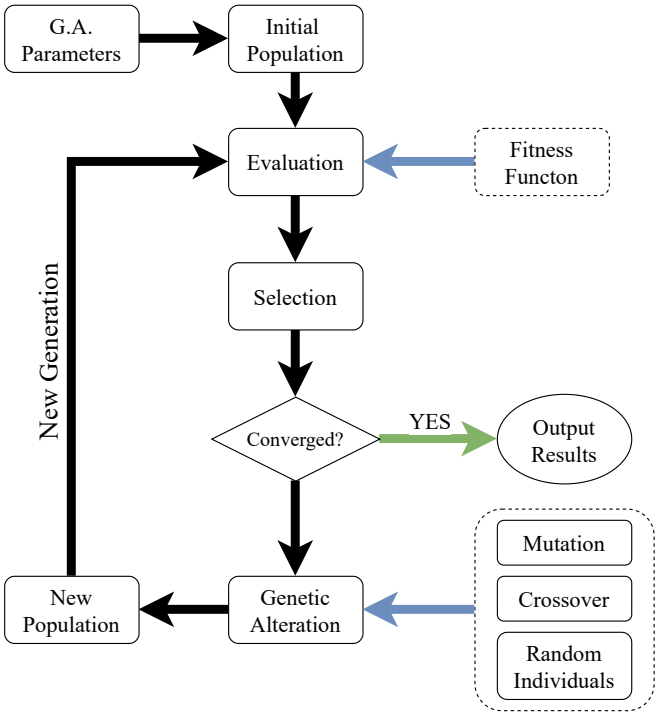

Fig. 3: Structure followed to design the implemented genetic algorithm described in Sect. IV.

airship. The aircraft is filled with hydrogen, which gives it a lightweight design and provides buoyancy. At $55\,\mathrm{km}$ above the surface the vehicle is 100% buoyant, while at $70\,\mathrm{km}$ it is expected to be 10% buoyant. The aerobot has, in the top part of its wings, solar panels which supply energy to the batteries. Lastly, the aircraft airspeed ranges from 0 to $30\,\mathrm{m/s}$. A more detailed explanation of the main design parameters can be seen in the previous paper [10]. In this work, we consider that the aerobot can turn rapidly compared to the large distances it travels, therefore, we consider that it is sufficient to represent the trajectory into a piece-wise linear path.

III. Background

A. Genetic Algorithms

In this paper, Genetic Algorithms (GA) are applied to solve the problem defined in the previous section. Genetic algorithms are often used for parameter selection and performance optimization of a system, but in this paper, they will be used to find the best path between two coordinates in a 3D windy environment. GAs are inspired by the biological world and are modeled after Darwin's theory of evolution [16], [17]. They use a survival of the fittest procedure to solve computational problems and iteratively search for the global optimal solution. An important feature of GAs is that it provides tools to balance exploration and exploitation, making it possible to obtain a globally optimal solution instead of falling into local optima.

As shown in Fig. 3, GAs begin with the selection of an initial population of i individuals which are possible solutions to the problem. Each individual is made up of different genes, which represent the design parameters [18]. In each iteration, also referred to as generation, the GA simultaneously evaluates all individuals against a fitness function. The fitness function is used to determine how well an individual performs in the environment represented by the design requirements and constraints. Individuals with higher fitness functions usually get higher chances to survive in the next generation.

At the end of each generation, a selection or reproduction mechanism is applied to choose the individuals that survive into the next generation. When the individuals reproduce, they create a new possible solution, their offspring, by undergoing genetic alterations [19]. Possible genetic operators are mutation and crossover. The selection of the GA parameters and the fitness function has a high impact on the efficiency of the GA and the balance between exploration and exploitation. For example, if there are only a few individuals in a population or the individuals undergo only minor, low-impact changes (e.g. in mutation), exploitation is favored, which might lead to quick convergence towards a local extremum [20]. If, however, the population is very large and genetic operations are very aggressive (e.g. in crossover), exploration is favored, which can lead to loss of obtained information and increased computational effort [21].

B. Flying with strong winds

In our problem formulation, we assume winds up to $90\,\mathrm{m/s}$ and a maximum aerobot speed of $30\,\mathrm{m/s}$ relative to the wind. Evidently, the vehicle cannot compensate for the wind and is mostly carried away, even if it is heading directly against it. This situation, which occurs when the current velocity is greater than the vehicle's relative velocity to the current, leads to unreachable regions [22]. To guarantee the feasibility of the paths generated by the GAs, we included a restriction on the generation of waypoints. A path between two waypoints, (x_i, y_i, z_i) and $(x_{i+1}, y_{i+1}, z_{i+1})$, is valid if it is inside the cone, shown in Fig. 4 and defined by the angle β_{max}. The cone angle, as formulated by [22], can be obtained from

$$\cos\left(\beta_{max}\right) = \frac{\sqrt{\|\vec{c}\|^2 - v_{max}^2}}{\|\vec{c}\|}, \tag{1}$$

where c is the velocity of the wind at the parent waypoint, (x_i, y_i, z_i), and v is the vehicle's maximum speed relative to the wind. Here we consider that the wind field is constant in the proximity of the parent waypoint.

IV. Methodology

We propose the use of GAs to solve the problem stated in Section II. For this, the next subsections define each block of Fig. 3. A pseudo-code of the proposed method can be seen in Algorithm 1.

A. Initial Population

In order to create an efficient-optimal genetic algorithm, the creation of the initial population (P) is an important

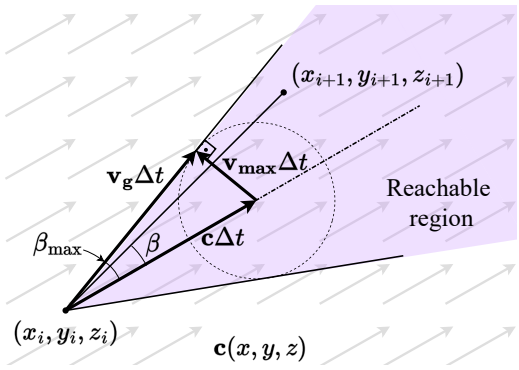

Fig. 4: Reachability cone computed in the function of the wind and vehicle speeds.

Algorithm 1 Energy and Wind Aware Genetic Algorithm for Path Planning

```
1: function [σ*, cost*] = MAIN(G)
2:     P ← InitialRandomPopulation(G)
3:     for g = 1 to G.N_g do
4:         for i = 1 to G.N_i do
5:             C(i) ← 0
6:             for s = 0 to G.N_s + 1 do
7:                 p_s ← GetGenes(P(i), s)
8:                 p_{s+1} ← GetGenes(P(i), s + 1)
9:                 cost(s) ← Cost(p_s, p_{s+1})
10:                if cost(s) = ∞ then
11:                    C(i) ← ∞
12:                    break
13:                end if
14:                C(i) ← C(i) + cost(s)
15:            end for
16:        end for
17:        P ← Selection(G, P)
18:        P ← Mutation(G, P)
19:        P ← Crossover(G, P)
20:        P ← NewRandomIndividuals(G, P)
21:    end for
22:    (σ*, cost*) ← BestIndividual(P)
23: end function
```

step [23]. In our case, all the paths created in the initial population (parametrized by $G.N_i$ Algorithm 1) are feasible and with the number of waypoints, being the same and defined by $G.N_s$. Individuals with $G.N_s$ waypoints are generated from start to goal until all waypoints are feasible, as seen in Algorithm 2. Each individual ($P(i)$) is formed by start position, waypoints (s), and goal position, giving each of those a gene per each coordinate, such that coordinates (x, y, z) would form a total of 3 genes, as shown in Fig. 5. The $InitialRandomPopulation()$ function in Algorithm 2 creates the initial population (P), where all the generated paths are considered unfeasible (line 4 of Algorithm 2), until proven otherwise. Each waypoint of the path (N_s), except for the last one (goal), is created randomly inside the feasible region, generated with the function $Cone()$.

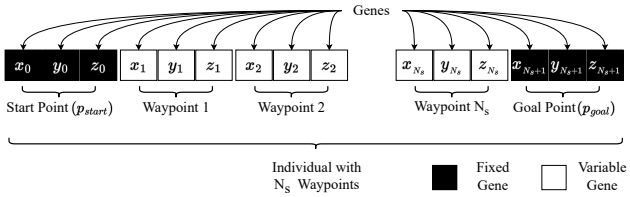

Fig. 5: Each path individual is formed by a start position, N_s waypoints, and a goal position, each formed by (x, y, z) coordinates.

Algorithm 2 Initial Random Population Generation with Accessibility Cone Restriction

```
1: function P = INITIALRANDOMPOPULATION(G)
2:     P ← ∅
3:     for i = 1 to G.N_i do
4:         feasible ← False
5:         while feasible = False do
6:             P(i) ← CreateIndividual(G, i)
7:             (x_o, y_o, z_o) ← G.p_start
8:             for s = 1 to N_s do
9:                 c⃗ ← WindField(x_o, y_o, z_o)
10:                (x, y, z) ← Cone(G, c⃗, (x_o, y_o, z_o))
11:                (x, y, z) ← Constrain(G, (x, y, z))
12:                P(i) ← WriteGenes(P(i), s, (x, y, z))
13:                (x_o, y_o, z_o) ← (x, y, z)
14:            end for
15:            if InCone(G.p_goal, c⃗, (x_o, y_o, z_o)) then
16:                feasible ← True
17:            end if
18:        end while
19:        P ← P ∪ P(i)
20:    end for
21: end function
```

After that, $Constrain()$ function places the waypoint inside the environmental dimensions. Afterward, the waypoint (s) is stored in the corresponding individual ($P(i)$). Then, the $InCone()$ function, checks for goal position, $G.p_{goal}$, feasibility (x and z are always feasible due to the limits established at the beginning) and if y is inside the cone region, then, the path, including the start and goal position, is feasible (line 16 of Algorithm 2). If in the end, the path is still not feasible the function dismisses the path ($P(i)$) and searches for another one until the path is feasible (line 5 of Algorithm 2).

B. Fitness Function

Our proposed fitness function, $Cost()$, is based on the energy exchanges of the aerobot when it travels from the start to the goal position so that the optimal path would be the one with less energy consumed. This energy is calculated by accounting for different sources of energy that the aerobot consumes or generates: energy consumed by the propellers, accumulated potential energy, and energy generated by solar panels. The cost of the path $C(i)$, as shown in line 14 of

Algorithm 1, is the addition of the costs, $cost(s)$, to go from one waypoint (p_s) to another (p_{s+1}) until the goal is reached ($s = G.N_s + 1$). The energy-based cost function is proposed in our previous work as [10]:

$$C = \begin{cases} E_{prop} + E_{pot}^{opp} + E_{solar}^{opp} & \text{if } b + \Delta b \geq 0 \\ \infty & \text{if } b + \Delta b < 0 \end{cases} \quad (2)$$

where E_{prop} is the energy consumed by the propellers, E_{pot}^{opp} is the potential energy, E_{solar}^{opp} is the energy produced by the solar panels, and Δb is battery change, computed as:

$$\Delta b = -E_{prop} + E_{solar}. \quad (3)$$

If during the calculation of the energy cost of the waypoint ($cost(s)$), the battery percentage, b, is lower than zero, $b < 0$, (which means that the aerobot runs out of battery), then we consider the path cost ($C(i)$) to be infinity.

The energy consumed by the propellers, E_{prop}, is based on the integration of the instant power, $P_{prop} = T \cdot v_a$, required by the propellers considering the efficiency of the propellers η_{prop} and shaft η_{shaft}:

$$E_{prop} = \int \frac{T \cdot v_a}{\eta_{prop} \cdot \eta_{shaft}} \, dt \quad (4)$$

where T is thrust and v_a is the aerobot velocity.

The complement of the potential energy is computed as:

$$E_{pot}^{opp} = E_{pot}^{max} - E_{pot} = m \cdot g \cdot (h_{max} - h), \quad (5)$$

where m is the mass of the aerobot, g is the gravity in Venus (approximately $8.87\,\mathrm{m/s^2}$) at height h of the aerobot. Notice that, to avoid negative energy, the potential energy is computed as the difference between maximum potential energy (computed at the highest possible altitude $h_{max} = 70\,\mathrm{km}$) and the current potential energy.

The complement of the solar energy generated by solar panels is calculated as:

$$E_{solar}^{opp} = E_{solar}^{max} - E_{solar} = \int (P_{solar}^{max} - P_{solar}) \, dt. \quad (6)$$

where the solar power is found by $P_{solar} = I_{solar} \cdot \eta_{panels} \cdot A_{panels}$ that accounts for available solar intensity, I_{solar}, solar panel efficiency, η_{panels}, and solar panel area, A_{panels}. In the Venusian atmosphere, the maximum power absorption of the solar panels, P_{solar}^{max}, is $560\,\mathrm{W/m^2}$ which happens when the aerobot is at its highest altitude.

C. Selection

Once, the evaluation of each individual ($C(i)$) is done (using the fitness function), it is time for the selection of the population (line 17 of Algorithm 1). In this work, we are considering two different selections. The first one is the Elitist Selection, which selects the number of best individuals, $G.nb$, to survive unchanged to the next generation [24]. The second one is the Rank Selection which ranks each individual based on their score in the fitness function, in this case, based on the lower energy from start to goal $C(i)$. After being ranked, each individual is assigned a probability of survival given by

$$prob_i = q - (N_i - 1)r, \quad (7)$$

with $q = 2/N_i$ and $r = 2/(N_i(N_i - 1))$, where q is the maximum selection pressure, meaning that the one with the largest energy path consumption has 0 chances of surviving to the next generation ($G.N_g$). N_i is the number of individuals in the population, and r is the amount by which the score of each individual in the rank decreases.

D. Mutation

Once the individuals are selected, then, they go through the first genetic alteration [24], called a mutation, line 18 of Algorithm 1. Mutation consists in changing a random number of genes in each individual. These genes will be mutated individually by adding or subtracting a value between the maximum ("Max. Mutation" parameter) and minimum ("Min. Mutation" parameter) range established as input parameters of the algorithm. Then, the selected population will return to the main one (P) to proceed with the crossover.

E. Crossover

A number of individuals in the population are selected based on the input parameter "Crossover" (line 19 of Algorithm 1). In our implementation double crossover is used instead of single crossover to improve exploration, since more possibilities for different individuals can happen. The double crossover consists in selecting two individuals and taking two random genes from one of them, then taking the same genes from the second one and swapping the gene values of all of the genes compressed between the chosen two [25]. This alteration helps in exploiting part of the individual but also exploring a different part of the individual path to see if an improvement in the energy cost spent is possible in the next generation.

F. New Random Individuals

Finally, the last genetic alteration (line 20 of Algorithm 1) before we evaluate the new population (P) consists in adding $G.N_r$ new random feasible individuals using a procedure similar to the generation of the initial population in Algorithm 2. These individuals are added to the population by substituting the paths with the worst fitness function, in our case, the individuals with the largest energy consumed to travel from start to goal positions.

G. New Population

Lastly, the function $NewRandomIndividuals()$ takes all the individuals (P), after the selections and genetic alterations made in the previous subsections, and creates a new population that will go through the same process as the one described above but starting from the fitness function which will evaluate the energy cost ($C(i)$) of the new individuals (line 14 of Algorithm 1). This process is done multiple times based on the parameter "N. of Generations" ($G.N_g$) or until a certain given time is reached.

TABLE I: Genetic Algorithm Parameters

Parameters	Values
Population Size	80 individuals
New Rand. Individuals	6 individuals
Crossover	40 individuals
Mutation	40%
N. of Generations	100 generations
N. of Best Individuals	2 individuals
Max. Mutation	1000 m
Min. Mutation	-1000 m
Min. Range	$(0, -10, 5.5) \times 10^4$m
Max. Range	$(30, 10, 7) \times 10^4$m
Start Position	$(0, 0, 5.5) \times 10^4$m
Goal Position	$(30, 5, 6.8) \times 10^4$m
N. of Waypoints	5 waypoints

V. NUMERICAL RESULTS

This section shows the results obtained with the genetic algorithm described in the previous section. Our results illustrate the capability and efficiency of the algorithm when finding optimal paths for an aerobot in the atmosphere of Venus. The proposed algorithm was implemented in MATLAB® and run on a MacOS computer (MacBook Pro 2023, Processor M2 Pro, 10 cores, 36 GB RAM).

A. Algorithm Evaluation

Several trials were performed with the parameters in Table I to demonstrate the consistency of the algorithm designed. We can observe that in all of the results, such as the ones in Fig. 6 and 8, the planned path tends to move in high altitudes to allow the aerobot to recharge its battery, and then descends to reach the goal. We notice, by running the code more than 50 times, that the running time and energy cost were similar to each other with an error smaller than 3%, which only is due to the randomness of the initial population. Furthermore, different start and goal positions were tried. Fig. 6 shows the paths with different start and goal positions that explored low and high altitudes, positive and negative longitudes, and different positive latitudes. The start and goal for the blue path are, respectively, $(0, 0, 5.5) \times 10^4$m and $(30, 5, 6.8) \times 10^4$m. For the yellow path, they are $(0, 0, 5.5) \times 10^4$m and $(30, -7.5, 6.8) \times 10^4$m, for the green path, $(2, 0, 6.8) \times 10^4$m and $(30, 0, 5.8) \times 10^4$m, for the purple path, $(1.5, 6, 6.5) \times 10^4$m and $(27, 0, 6.5) \times 10^4$m, and, finally, for the red path they are $(0, 2.5, 5.7) \times 10^4$m and $(30, 6, 7) \times 10^4$m. Other trials were done for a larger number of waypoints ($G.N_s$), resulting in paths with similar costs as the ones with fewer waypoints. The only difference was the time performance of the algorithm which increases as the number of waypoints increases.

After consistency was proven, trials with different parameters were performed in order to find the right balance between exploration and exploitation. The parameters were chosen to be modified where "Population Size" ($G.N_i$), "New Rand. Individuals" ($G.N_r$), and "N. of Generations" ($G.N_g$), that took values from 60 to 90 individuals in steps of 10, from 0 to 30 individuals in steps of 5, and from 60 to 200 generations in steps of 20, respectively. A summary of the best and worst performances based on the energy cost of the path can be seen in Table II. In this table, aside from the four columns already mentioned, there are two more columns. The first one, "Time Cost", accounts for the aerobot's time, in seconds, to travel from the start to the goal position. The second one, "Computational Time", represents the time, in seconds, that the algorithm spent computing the best solution. Results show that the designed genetic algorithm performs better when the "N. of Generations" is large and, also, when "New Rand. Individuals" is not too large (≤ 15). Also, notice the short amount of time the algorithm takes to run, ranging from 1 s in short "N. of Generations" and "Population Size" to 5 s for larger ones. Fig. 7 show the improvement of the best path (found with the parameters in the first line of Table II) through the generations.

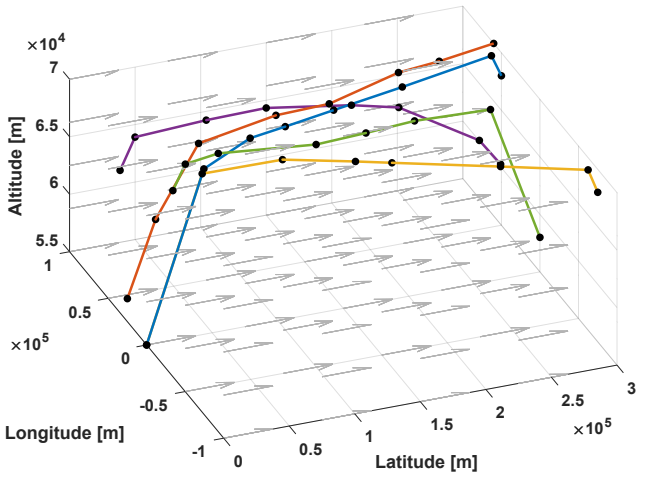

Fig. 6: Best individual (optimal path σ^*) throughout the generations of the GA for different start and goal positions.

TABLE II: Genetic Algorithm Performance

Energy Cost [J]	Time Cost [s]	Population Size	New Random Individuals	Number of Generations	Computational Time [s]
1.304×10^8	2645.1536	80	0	140	3.079
1.306×10^8	2620.1813	90	15	180	4.703
1.306×10^8	2626.1648	70	10	180	3.460
1.307×10^8	2631.6413	60	5	180	3.065
1.307×10^8	2633.5233	90	5	180	4.819
1.647×10^8	2693.3236	70	20	80	1.673
1.662×10^8	2652.5313	80	20	80	1.892
1.707×10^8	2626.6479	60	20	100	1.875
1.712×10^8	2608.6214	70	30	100	2.249
1.725×10^8	2820.6328	70	25	60	1.296

B. Comparison with an RRT-based Algorithm

In this subsection, we compare the results from the proposed Genetic Algorithm with the results obtained from the previous work [10]. In our previous work, a sampling-based planner, the EWRRT, was developed to solve the same navigation problem solved in this paper. The algorithm is based on RRT with a local planner that relies on Dubin's Airplane [26] paths. The algorithm is not optimal but searches for low-cost paths by always choosing connections that minimize the cost function in (2). The wind is considered

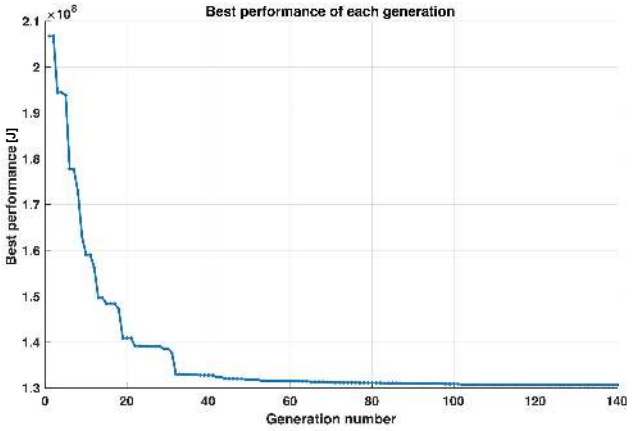

Fig. 7: Evolution of the best individual (σ^*) based on the path cost ($cost^*$) in function of the generations (best of Table II).

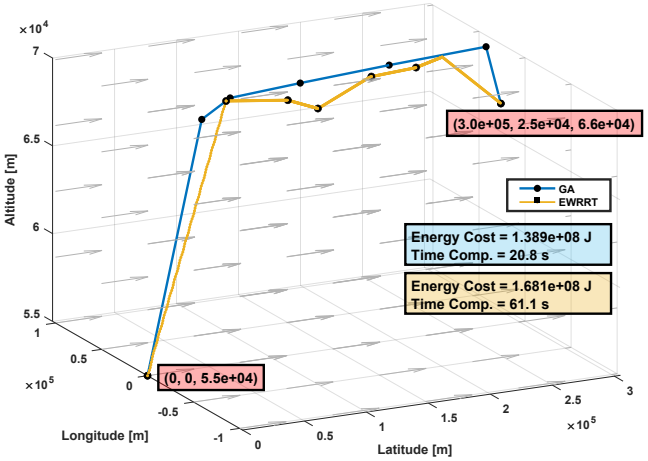

Fig. 8: The feasible path found by the GA is compared to the feasible path found by EWRRT [10]. The GA path is piece-wise linear, while EWRRT connects the waypoints with Dubin's Airplane paths drifted with the wind.

indirectly, i.e., the Dubins Airplane paths are created with a virtual goal and the resulting trajectory is subjected to wind drift before including in the EWRRT graph. To compare the performance of the methods, both algorithms were given the same start and goal conditions: $p_{start} = (0, 0, 5.5) \times 10^4\text{m}$ and $p_{goal} = (30, 2.5, 6.6) \times 10^4\text{m}$. The paths generated by the two algorithms are shown in Fig. 8. The EWRRT's path (yellow) was obtained using 2000 samples and a step size of $1.5 \times 10^4\text{km}$. The energy cost of the path found was $1.6805 \times 10^8\text{J}$, while the path's time cost was $3.0562 \times 10^3\text{s}$. The path was computed by EWRRT in $61.12\,\text{s}$. While EWRRT's parameters are not the same as the ones in the GA, we tried to make them similar in order to make a fair comparison. The GA's path used parameters: 80 for "Population Size", 6 for "New Rand. Individuals", and 1000 for "N. of Generations". The path found had an energy cost of $1.3893 \times 10^8\text{J}$ and a path's time cost of $2.6945 \times 10^3\text{s}$. This path was computed in $20.81\,\text{s}$. The paths in Fig. 8, along with costs and times mentioned, show that GA, with $1/3$ of EWRRT's computational time, is able to reach a solution that has a lower path energy cost ($cost^*$) and a shorter travel time. Also, different trials show that EWRRT needs a larger number of samples and a smaller step size to increase the probability of finding a lower path energy cost, which would require an even larger computational time.

VI. CONCLUSIONS AND FUTURE WORK

This paper presented a Genetic Algorithm-based path planner designed for aerobot navigation in the Venusian atmosphere, where strong winds are present. This algorithm tries to find the optimal path (σ^*) based on a fitness function that accounts for the energy cost of the aerobot when traveling from the start to the goal position.

Results show good performance of the algorithm with trials that do not defer more than 3% from one another, which reflects that the initial population was created according to the needs of the problem. Tests accounting for different parameters were performed to see which would keep the best balance between exploration and exploitation of the GA. Furthermore, algorithm time performance was satisfactory with a time ranging from $1\,\text{s}$ for a small number of generations and population sizes to $5\,\text{s}$ for larger ones. Also, we show how GA outperforms EWRRT, a sampling-based algorithm previously proposed by our group, obtaining smaller energy and travel costs with a smaller computational time.

Our future work includes changing the fitness function to account for the localization of the aerobot. Assuming that the localization uses cameras pointing to the planet, the aerobot will have a lower localization error if it is closer to the surface of Venus ($55\,\text{km}$) and larger if it is far from it ($70\,\text{km}$), due to the thick layer of clouds present in the flying region. In this way, differently from the current cost function, which favors paths in high altitudes, the new one will represent a trade-off between flying above the clouds to charge but below them to localize. Another possible future work is adding non-flying zones in the environment, which would behave like cylindric obstacles in 3D. For our application, this could represent, for example, non-flyable regions due to current climate conditions.

REFERENCES

[1] M. J. Way, A. D. Del Genio, N. Y. Kiang, L. E. Sohl, D. H. Grinspoon, I. Aleinov, M. Kelley, and T. Clune, "Was venus the first habitable world of our solar system?" *Geophysical research letters*, vol. 43, no. 16, pp. 8376–8383, 2016.

[2] A. T. Basilevsky and J. W. Head, "The surface of Venus," *Reports on Progress in Physics*, vol. 66, no. 10, pp. 1699–1734, Sept. 2003, publisher: IOP Publishing. [Online]. Available: https://doi.org/10.1088/0034-4885/66/10/r04

[3] M. Marov, "Results of Venus Missions," *Annual Review of astronomy and astrophysics*, vol. 16, no. 1, pp. 141–169, 1978.

[4] R. Polidan, G. Lee, D. Sokol, K. Griffin, and L. Bolisay, "Venus Atmospheric Maneuverable Platform (VAMP)," in *Workshop on Venus Exploration Targets*, vol. 1781, May 2014, p. 6011, conference Name: ADS Bibcode: 2014LPICo1781.6011P. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2014LPICo1781.6011P

[5] G. Landis, "Low-altitude exploration of the venus atmosphere by balloon," in *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2010, p. 628.

[6] J. S. Greaves, A. Richards, W. Bains, P. B. Rimmer, H. Sagawa, D. L. Clements, S. Seager, J. J. Petkowski, C. Sousa-Silva, S. Ranjan, *et al.*, "Phosphine gas in the cloud decks of venus," *Nature Astronomy*, vol. 5, no. 7, pp. 655–664, 2021.

[7] G. A. Landis, C. LaMarre, and A. Colozza, "Venus atmospheric exploration by solar aircraft," *Acta Astronautica*, vol. 56, no. 8, pp. 750–755, 2005.

[8] J. W. Langelaan, "Tree-based trajectory planning to exploit atmospheric energy," in *2008 American Control Conference*, June 2008, pp. 2328–2333, iSSN: 2378-5861.

[9] A. Chakrabarty and J. Langelaan, "UAV flight path planning in time varying complex wind-fields," in *2013 American Control Conference*, June 2013, pp. 2568–2574, iSSN: 2378-5861.

[10] B. Martinez Rocamora Jr., A. P. I. Juan, and G. A. S. Pereira, "Towards finding energy efficient paths for hybrid airships in the atmosphere of venus," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2022, pp. 386–393.

[11] S. Karaman and E. Frazzoli, "Sampling-based Algorithms for Optimal Motion Planning," *arXiv:1105.1186 [cs]*, May 2011, arXiv: 1105.1186. [Online]. Available: http://arxiv.org/abs/1105.1186

[12] J. M. Buchanan, "Opportunity Cost," in *The World of Economics*, ser. The New Palgrave, J. Eatwell, M. Milgate, and P. Newman, Eds. London: Palgrave Macmillan UK, 1991, pp. 520–525. [Online]. Available: https://doi.org/10.1007/978-1-349-21315-3_69

[13] G. A. Landis and E. Haag, "Analysis of solar cell efficiency for venus atmosphere and surface missions," in *11th International Energy Conversion Engineering Conference*, 2013, p. 4028.

[14] G. Landis, A. Colozza, and C. LaMarre, "Atmospheric flight on Venus," in *40th AIAA Aerospace Sciences Meeting & Exhibit*. Reno,NV,U.S.A.: American Institute of Aeronautics and Astronautics, Jan. 2002. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2002-819

[15] A. Sánchez-Lavega, S. Lebonnois, T. Imamura, P. Read, and D. Luz, "The Atmospheric Dynamics of Venus," *Space Science Reviews*, vol. 212, pp. 1541–1616, Nov. 2017, aDS Bibcode: 2017SSRv..212.1541S. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2017SSRv..212.1541S

[16] C. Machinery, "Computing machinery and intelligence-am turing," *Mind*, vol. 59, no. 236, p. 433, 1950.

[17] I. Rechenberg, "Evolutionsstrategien," in *Simulationsmethoden in der Medizin und Biologie: Workshop, Hannover, 29. Sept.–1. Okt. 1977*. Springer, 1978, pp. 83–114.

[18] G. Erinc and S. Carpin, "A genetic algorithm for nonholonomic motion planning," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 1843–1849.

[19] V. Sathiya and M. Chinnadurai, "Evolutionary algorithms-based multi-objective optimal mobile robot trajectory planning," *Robotica*, vol. 37, no. 8, pp. 1363–1382, 2019.

[20] D. Lin, B. Shen, Y. Liu, F. E. Alsaadi, and A. Alsaedi, "Genetic algorithm-based compliant robot path planning: an improved bi-rrt-based initialization method," *Assembly Automation*, 2017.

[21] Q. Song, S. Li, J. Yang, Q. Bai, J. Hu, X. Zhang, and A. Zhang, "Intelligent optimization algorithm-based path planning for a mobile robot," *Computational Intelligence and Neuroscience*, vol. 2021, 2021.

[22] M. Soulignac, "Feasible and Optimal Path Planning in Strong Current Fields," *IEEE Transactions on Robotics*, vol. 27, no. 1, pp. 89–98, Feb. 2011.

[23] D. Gallardo, O. Colomina, F. Flórez, and R. Rizo, "A genetic algorithm for robust motion planning," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 1998, pp. 115–121.

[24] C. Lamini, S. Benhlima, and A. Elbekri, "Genetic algorithm based approach for autonomous mobile robot path planning," *Procedia Computer Science*, vol. 127, pp. 180–189, 2018.

[25] Z. Qiongbing and D. Lixin, "A new crossover mechanism for genetic algorithms with variable-length chromosomes for path optimization problems," *Expert Systems With Applications*, vol. 60, pp. 183–189, 2016.

[26] H. Chitsaz and S. M. LaValle, "Time-optimal paths for a Dubins airplane," in *2007 46th IEEE Conference on Decision and Control*, Dec. 2007, pp. 2379–2384, iSSN: 0191-2216.