Genetic algorithm for path planning of UAVs as a maze-solving problem

M A Gutierrez-Martinez, L E Cabriales-Ramirez, E U Rojo-Rodriguez, E J Ollervides-Vazquez, Pedro Castillo Garcia, O. Garcia-Salazar

Genetic algorithm for path planning of UAVs as a maze-solving problem

M.A. Gutierrez-Martinez[1], L.E. Cabriales-Ramirez[1], E.U. Rojo-Rodriguez[1], E.J. Ollervides-Vazquez[1,2], P. Castillo[3] and O. Garcia-Salazar[1]

***Abstract*— This research work addresses the problem of path planning for unmanned aerial vehicles (UAVs) in complex environments such as a maze. A genetic algorithm (GA) with variable chromosomes and gene change conditions is designed when the decision point criteria and a collision with a wall are presented; without considering large chromosomes. Our proposed GA obtains the sequence of minimum movements required to solve the maze. Then, the trajectories are generated by high-order polynomials. Finally, numerical simulations and real-time tests are carried out to validate the proposed algorithm.**

I. Introduction

Unmanned aerial vehicles (UAVs) have been used in multiple increasingly complex applications, such as agriculture, parcel delivery, and surveillance [1], [2], and [3], due to their significant development and technological advances in control, intelligence techniques, and algorithms. Due to this, algorithms for optimal navigation have been of interest to researchers in robotics, mechatronics, and aeronautics fields. These algorithms have been widely studied and used in UAVs to achieve optimal navigation to make UAVs autonomous and efficient [4]. In [5], authors worked on the task of exploring unknown environments. They aborded the construction of maps for indoors where the information on the environment was unknown. Then a Lee algorithm was implemented for reduced the memory cost to generate the map, and an A-star algorithm calculated the distance to a particular point. Thus, Lee and A-star algorithms calculated angles and distance to nodes from a point for the orientation of UAVs in a maze.

The tasks such as exploration, mapping, and recognizing unknown environments have been essential for autonomous navigation [6]. Thus, algorithms that solve path planning in complex environments as maze-like have been essential to achieve this navigation. In this sense, path planning algorithms were designed to execute missions considering obstacle penalties or a minimum distance of trajectory criteria [7]. In [8] and [9], a modified potential field algorithm for path planning of UAV in an indoor environment was designed. The improved potential field algorithm was capable of avoiding unknown obstacles types of barriers and reaching the target of the trajectory. Therefore, the simulations in a virtual environment were performed, and experiments were implemented in real-time in a low-cost UAV to validate the proposed method. In [10], an algorithm for path planning based on cells decomposition was presented, in which authors implemented this algorithm to find trajectories in a 3D static environment for UAVs.

Classical methods were used in path planning in maze environments; however, heuristic and intelligent methods, among which the evolutionary and bio-inspired algorithms are used in cases where the classical methods usually presented problems. In [11], a particle distribution algorithm (PSO) was used to find optimal trajectories, while by high degree bezier curves based on polynomial equations, authors smoothed the trajectory and made it continuous for tracking. Hence the proposed method was implemented in environments with barriers as obstacles, avoiding colliding with the walls and generating safe trajectories. In [12] and [13], authors showed that bio-inspired and evolutionary methods such as ant colony and GA solve the path planning problem for UAVs by creating efficient trajectories to avoid collisions. In [14], a GA was designed to solve complex maze-like environments through sequences of movements. However, the authors worked with large chromosomes of 400 genes, which make the function of genetic operators difficult. In [15], the authors used GA to find a maze puzzle solution in which used a novel Edge assembly crossover operator (EAX) is used, and it reduces computational time and finds trajectories that solve a maze picture environment. Finally, in [16], the authors proposed a GA to find the shortest path in a maze-type network. The proposed algorithm solved the classical combinatorial optimization problems and showed that GA is a valuable tool for solving maze-like environments and path planning.

The main contribution of this work is a variable chromosome genetic algorithm to solve mazes in the least number of movements and with the shortest path distance. The motion sequence is converted to waypoints to generate UAV trajectories through high-order polynomials. Finally, these trajectories were simulated and implemented in a UAV to validate the proposed algorithm.

The organization of this work is as follows: section II presents the problem statement to be solved. Section III shows the equations of the mathematical model of UAVs. Section IV describes our genetic algorithms, the genetic operators with their percentages of crossover and mutation, the objective function, and the criteria of gene change. In section V, numerical simulations and real-time validation

[1]Aerospace Engineering Research and Innovation Center, Faculty of Mechanical and Electrical Engineering, Autonomous University of Nuevo Leon, Apodaca NL, Mexico. manuel.gutierrezmrt@uanl.edu.mx; luis.cabrialesrm@uanl.edu.mx; edgar.rojordrg@uanl.edu.mx; edmundo.ollervidesvz@uanl.edu.mx; octavio.garciasl@uanl.edu.mx

[2]La Laguna Institute of Technology-TecNM, Torreon Coahuila, Mexico. ejollervi@ieee.org

[3]Heudiasyc Laboratory CNRS UMR 7253, Universitè de Technologie de Compiègne, Compiègne France, castillo@hds.utc.fr

for the generated high-order polynomials are presented. The conclusions are present in section VI.

II. Problem statement

This research focuses on the problem of creating a path in complex environments such as a maze. The optimization of the path is considered the criteria of minimum distance and the minimum number of movements since the shortest distance is not necessarily the path with the fewest movements. Thus, having many changes or turns in the path can be a problem for the trajectory generators and the control techniques; then, minimizing the number of movements is essential. A genetic algorithm is proposed to minimize the number of movements and the path distance. Solving a path in a maze becomes a combinatorial optimization problem because a specific order of movements is needed to find a path. For this reason, a genetic algorithm of variable chromosomes is proposed with a criterion of changes of gene only when the conditions of the intersection point, decision point, final point are accomplished; this helps to solve the path in the minimum number of movements and with a short variable of chromosomes.

III. Equations of motion

The mathematical model of UAV considers the inertial frame $\mathcal{I}$=$\{x_{\mathcal{I}}, y_{\mathcal{I}}, z_{\mathcal{I}}\}$, the body frame $\mathcal{B}$=$\{x_{\mathcal{B}}, y_{\mathcal{B}}, z_{\mathcal{B}}\}$ that is center of gravity fixed on the vehicle and the wind frame $\mathcal{A}$=$\{x_{\mathcal{A}}, y_{\mathcal{A}}, z_{\mathcal{A}}\}$; this frame is generated during the flight.

The formulation Newton–Euler for rigid bodies is used to obtain the mathematical model of quadrotor (2)–(4).

$$\dot{\xi} = V \tag{1}$$

$$m\dot{V} = Re_3(-T_T) + mge_3 \tag{2}$$

$$\dot{R} = R\hat{\Omega} \tag{3}$$

$$J\dot{\Omega} = -\Omega \times J\Omega + \tau_a \tag{4}$$

where the position and velocity to the inertial frame relatives are $\xi = (x, y, z)^\top \in \mathbb{R}^3$ and $V = (\dot{x}, \dot{y}, \dot{z})^\top \in \mathbb{R}^3$. $\eta = (\phi, \theta, \psi)^\top \in \mathbb{R}^3$ represents the rotation coordinates where ϕ, θ and ψ represent roll, pitch and yaw, respectively.

The rotation matrix $R \in SO(3) : \mathcal{B} \to \mathcal{I}$, it satisfies $SO(3) = \{R | R \in \mathbb{R}^{3\times3}, \det[R] = 1, RR^\top = R^\top R = I\}$ this matrix is Euler angles parameterized ϕ, θ and ψ. Then the rotation matrix is written as

$$R = \begin{pmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{pmatrix}$$

$\Omega = (p, q, r)^\top \in \mathbb{R}^3$ contains the angular velocity with respect to the body frame, and $\hat{\Omega}$ is a skew-symmetric matrix, that is characterized by

$$\hat{\Omega} = \begin{pmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{pmatrix}$$

In addition, vectors e_1, e_2, and e_3 of the canonical basis of $\mathbb{R}^3$ are considered. The inertia moments are considered on $J \in \mathbb{R}^{3\times3}$, and $m \in \mathbb{R}$ denotes the mass of the quadrotor UAV. $T_T \in \mathbb{R}^3$ represents the total thrust, and $\tau_a \in \mathbb{R}^3$ is the actuator moments of the quadrotor. The actuator moments τ_a of four rotors are described as follows

$$\tau_a = \begin{pmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{pmatrix} = \begin{pmatrix} d[(f_2 + f_4) - (f_1 + f_3)] \\ d[(f_3 + f_4) - (f_1 + f_2)] \\ Q_2 + Q_3 - Q_1 - Q_4 \end{pmatrix}$$

where d is the distance from the center of mass to the rotor axis, f_i is the lift force or thrust force of the propeller for $i = 1, 2, 3, 4$ and $Q_i = \rho_a A_{d_i} \omega_i^2 r_i^3 c_{Q_i}$ is the reaction moment of the rotors with c_{Q_i} that denotes the coefficient moments of the rotor shaft, ρ_a is the air density, A_{d_i} denotes the disk area, ω_i denotes the angular velocity of the rotor and r_i is the rotor radius for $i = 1, 2, 3, 4$.

IV. Genetic algorithm maze solution

A. Initialization of genetic algorithm

A genetic algorithm (GA) is proposed to solve a maze in a static environment. Our genetic algorithm works with a random population of real numbers instead of a binary population to reduce the chromosome length so that the genetic operators calculations are less complicated. Each gene of population is defined as $A \in \mathbb{R}$, where $A = \{1, 2, 3, 4, 5\}$. In addition, a population of 100 individuals and 20 genes as maximum in the chromosome is implemented. The phenotype of the genetic algorithm is translated as state variables of motion for the resolution of the maze. Then each algorithm gene can take a value between 1 to 4; these states are translatable into the four possible movements, right, up, left, and down, respectively; also, a fifth state is added that represents no movement. In this sense, when the sequence of movements reaches its final point, it remains static. Also, to save processing and avoid using chromosome length, a variable chromosome criterion is used, where a case can be resolved in 3 movements; however, the same case can be resolved in 2 movements, it can be seen in Fig. 1.

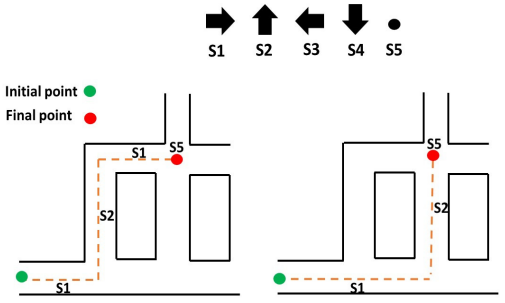

Fig. 1: Scheme of movements.

Our proposed GA aims to optimize the number of movements until reaching the final point; the order of the genes in the chromosome is essential because it translates into the order of movements in the maze. Moreover, in our

proposed GA, the algorithm only evaluates the necessary genes until reaching the final point. The chromosome is variable and depends on the number of movements until reaching the final point. Thus, the algorithm works with movements controlled by the index of chromosome gene; in this proposed algorithm, the gene index only changes when the following conditions are achieved.

The first gene change condition to determine subsequent movement is considered if it hits a wall and the same gene movement is maintained; this is shown in Fig 2. The second condition of gene change is that the sequence of movements is at a decision point, which means the points where different paths can be taken. The third change condition is when the final point is reached; in this case, there is a gene change in the chromosome of movement and a change in individual in the population. Fig. 3 shows the operation of the gene change conditions. It can be seen in Fig. 3 a) when the sequence of movements reaches the first decision point, it can take two accessible paths, which are up, and the right. It can also go down or even backward; however, these are paths penalized by distance and wall collision, which these paths are discarded in future iterations. Then, when the sequence of movements reaches the decision points, there is an increase in the index that controls the movement gene; however, this is the same value. Thus, the algorithm continues moving along the same path to the right. Then, the movements upon reaching the second decision point a change in the chromosome occurs again, this causes increasing the index that controls the movement gene. The next movement in the chromosome is towards up the path is takem up until the final point is reached, as shown in Fig. 3 b). When reaching the final point, it is observed in Fig. 3 c) a number five, is placed in the following gene of the movement chromosome, remaining at that point.

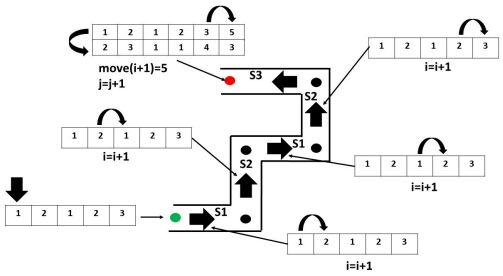

Fig. 2: Change of direction by the intersection point.

B. Objective function

The objective function considers the following criteria to be able to solve the maze, that the distance of the path is as short as possible by (5), that it be solved in the least number of movements, and that the distance between the final point of the path and the final point of the maze is the minimum. In addition, penalty pen conditions are added for these sequences of movements that leave the maze and

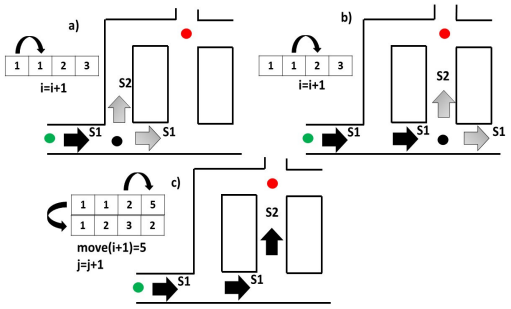

Fig. 3: Change of direction by decision point.

collide with a wall. That means these paths are discarded in future iterations. The genetic operators select the sequences of movements close to the final point, avoiding colliding in a wall, and remaining in the maze.

$$dist_j = \sqrt{(xp_{j,m} - stopx)^2 + (yp_{j,m} - stopy)^2} \quad (5)$$

In Equation (5) $dist$ is the distance from the last movement to the final point, xp and yp are the position of the sequence of movements, and it is updated every cycle. $stopx$ and $stopy$ are the positions of the final point. $crash$ is a counter of times when a sequence of movements collides with a wall, and pen is a high penalty added for leaving the dimensions of the maze. $Countm$ is a counter of movement changes, and $Countp$ is a counter of steps taken to solve the maze until it reaches the desired point. Then, the sequence of movements that reach the final point satisfies the conditions; the sequence of movements that make it in fewer steps satisfies better the established conditions and minimizes the distance of the path. In this sense, the objective function is the following (6), where k_i is a constant, with $i = \{1, 2, 3, 4\}$

$$\begin{aligned} fv_j = dist_j + k_1 * Countm_j + k_2 * Countp_j \\ + k_3 * crash_j + k_4 * pen_j \end{aligned} \quad (6)$$

C. Genetic operators

1) Selection: For the selection of individuals, a deterministic criterion is used. Based on the fitness value of the movement sequences, the individuals are ordered from lowest to highest, and half of the population with the best fitness values are chosen for crossover.

2) Crossover: For the crossover operator, a double crossover point criterion is used. Fig. 4 shows the double crossover point criterion, where a crossover point PC splits the chromosome and sets a boundary between the secondary crossover points $Pcsub1$ and $Pcsub2$ where the crossover points change in each iteration. Then, because the entire selected population experiment crossover, it has 100 % crossover.

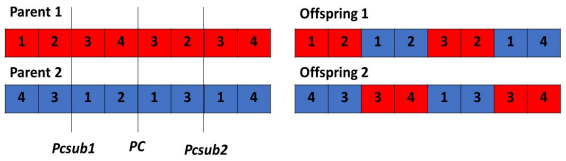

Fig. 4: Double crossover point.

3) Mutation: The high mutation criterion is used with a percentage of 4 % where 80 % of the population experiences the mutation. However, the best individuals are preserved for future iterations to maintain some individuals without mutation and to have convergence. The high mutation provides greater exploration and a combination of movements. The mutation occurs by altering the chain of movements by introducing a new movement, a random number between 1 and 4.

The genetic algorithm, in general, can be observed in *algorithm 1* also is shown the criteria of gene change in the chromosome of movements, where the variable i controls the index. *Algorithm 2* shows the cases of movements and the counters to detect how many accessible paths there are. If there are two free paths, it is known that it is at the decision point, and this is counted by $con1$ while $con2$ is a counter to determine it is within the limits of the maze.

Algorithm 1: Genetic algorithm to solve maze.

Set variable P population size, Nc number of chromosomes.
Set Pc crosrover rate,Pm Mutation rate.
Set counter $i = 1$, $j = 1$, $g = 1$
Initializes population $Move_{P,Nc}$
while $g <= gene$ **do**
  Objective function
  while $j <= P$ **do**
    $xp = startx$; $yp = starty$;
    while $i <= Nc$ **do**
      $n = move_{j,i}$
      Algorithm2(n)
      if $con2 >= 4$ **then**
        if $con1 < 1$ **then**
          $i = i + 1$
        end
      end
      if $xp < 0$ *or* $yp < 0$ *or* $xp > pfx$ *or* $yp > pfy$ **then**
        $fv_j = fv_j + p$
        $i = i + 1$
      end if $xpf = obsx$ *and* $yfp = obsy$ **then**
        $i = i + 1$
      end
      if $xp = obsx$ *and* $yp = obsy$ **then**
        $crash_j = crash_j + 1$
        $i = i + 1$
      end
      $ic = i$
      if $xp = stopx$ *and* $yp = stopy$ **then**
        $move(i + 1) = 5$; $i = Nc$
      end
      $m = m + 1$;
    end
    $countp = m$; $countm = ic$
    Calculate distance by Equation (5)
    $j = j + 1$;
    Evaluate objective function with (6)
  end
  Selection operator
  Crossover operator
  Mutation operator
  Replace population
  $g = g + 1$;
end

V. Validation

A. Validation genetic algorithm

Two maze-type environments are prepared for six different cases to evaluate the genetic algorithm. Also, numerical simulations and experimental validation are executed to verify that the control algorithms can follow the high-order polynomials.

1) Case 1: In case 1, a maze is prepared as shown in Fig. 5. The green point is the initial point at $0.3m$, $0.3m$ at x, y, and the red point is the final point at $1.2m$ and $3.9m$ at x, y. The blue dotted line is the sequence of movements by the genetic algorithm; it can be seen that the algorithm solves this case in 4 movements, this being the minimum number of movements to solve this case. The orange line is the trajectory response of the navigation points generated by the sequence of movements; a high–order polynomial generates this trajectory. This case is solved with seven genes of chromosomes.

2) Case 2: Case 2 is shown in Fig. 6 as in the previous case; it has an initial point at $0.3m$ and $0.3m$ at x, y, the final point is at $4.5m$ and $3.9m$ at x, y. It can be observed that the sequence of movements generated by the genetic algorithm passes through three decision points and an intersection point; also, the algorithm solves this case in the minimum number of movements which for this case is four. The high-order polynomial generates the orange trajectory, and the points generated by the sequence of movements; are obtained by the GA. This case is solved with six genes of chromosomes.

3) Case 3: Case 3 is presented in Fig. 7 in which another maze-like environment is presented with more complicated paths and more decision points. The initial green point is at $0.10m$, $0.10m$, at x, y respectively, and a final redpoint is at $0.3m$ y $2.6m$ in x, y. The path, generated by the sequence of movements, is a blue line dotted. It can be seen that the movements pass through five decision points, and these could

Algorithm 2: Cases algorithm.

switch n **do**
  Case 1 Move right
  $xpf = posx + (dp * 2)$; $ypf = yp$;
  $xp = xp + dp$; $yp = yp$;
  Case 2 Move up
  $xpf = xp$; $ypf = yp + (dp * 2)$;
  $xp = xp$; $yp = yp + dp$;
  Case 3 Move left
  $xpf = xp - (dp * 2)$; $ypf = yp$;
  $xp = xp - dp$; $yp = yp$;
  Case 4 Move down
  $xpf = xp$; $ypf = yp - (dp * 2)$;
  $xp = xp$; $yp = yp - dp$;
  Case 5 No move
$con1 = 0$; $con2 = 0$;
$x_1 = xp + \delta p$; $y_1 = yp + \delta p$; $x_2 = xp$; $y_2 = yp + \delta p$;
$x_3 = xp - \delta p$; $y_3 = yp$; $x_4 = xp$; $y_4 = yp - \delta p$;
$x_5 = xp + \delta p$; $y_5 = yp + \delta p$; $x_6 = xp + \delta p$;
$y_6 = yp - \delta p$; $x_7 = xp - \delta p$; $y_7 = yp + \delta p$;
$x_8 = xp - \delta p$; $y_8 = yp - \delta p$;
while $n <= 8$ **do**
  if $x_n = obsx$ *and* $y_n = obsy$ **then**
    $con1 = con1 + 1$;
    if $i >= 5$ **then**
      $con2 = con2 + 1$;
    end
  end
end

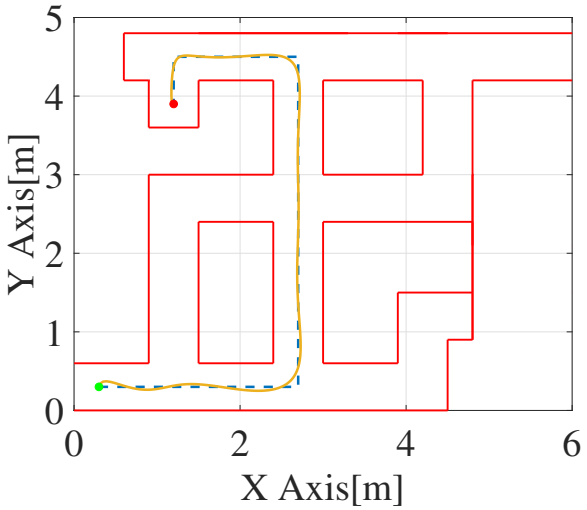

Fig. 5: Case 1.

have deviated or made unnecessary movements. However, the algorithm finds the appropriate sequence of movements to arrive at the final point with the minimum number of movements, which are six movements. After having the sequence of movements, the sequence of movements is discretized to obtain navigation points; these navigation points obtained are used to generate the continuous trajectory through the high–

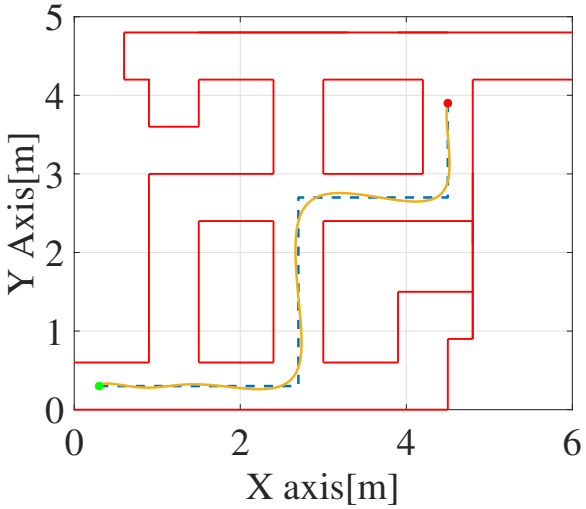

Fig. 6: Case 2.

order polynomial, which can be observed in orange color. This case is solved with nine genes of chromosomes.

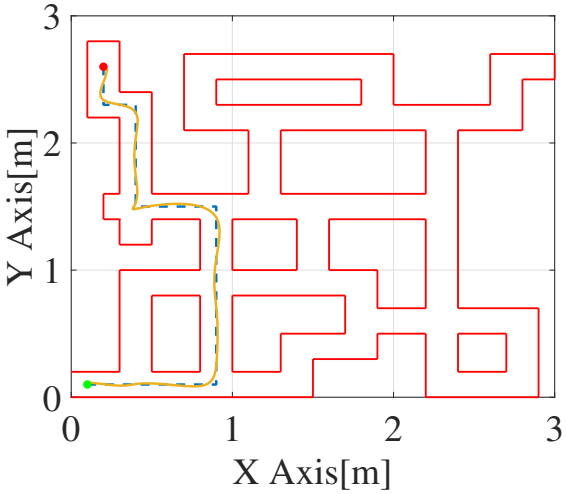

Fig. 7: Case 3.

4) Case 4: In case 4, it can be seen that the initial point is at $0.1m$, $0.1m$ in x, y and the final point is at $2.9m$, $2.6m$, in x, y; the final point is set as far from the inital point to test the algorithm, it can be seen in Fig. 8. The path generated by the sequence of movements is the blue dotted line. It can be seen that the algorithm can go through many decision points, so it is difficult to reach the final point and even more difficult to arrive at the least number of movements. Then, it can be noted that the algorithm finds the trajectory in the least number of movements, which is seven. This case is solved with twelve genes of chromosomes.

5) Case 5: Case 5 is shown in Fig. 9. In this case, the initial point and final point are placed at different extremes, where the initial point is at $2.8m$ and $0.1m$ at x, y while the final point is at $0.3m$ and $2.6m$ the algorithm solves the maze in the minimum number of movements, which in this

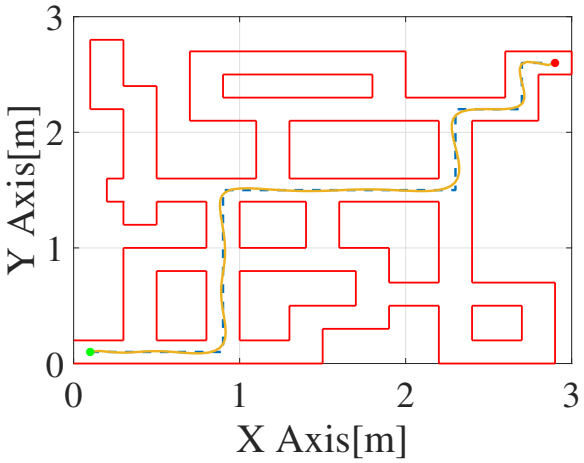

Fig. 8: Case 4.

case are six. As in the previous cases, the points generated by the sequence of movements are used, and the high-order polynomial is obtained to generate the trajectory, the blue dotted line is the trajectory generated by the sequence of movements. On the other hand, the orange line is the continuous trajectory. This case is solved with nine genes of chromosomes.

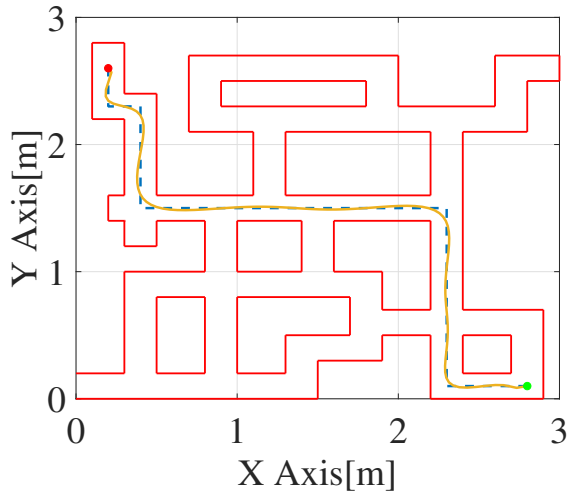

Fig. 9: Case 5.

6) Case 6: Finally, case 6 is shown in Fig. 10. It can be seen that the initial point is at $1.4m$ in x and $0.2m$ in y, the blue dotted line represents the sequence of movements made by the GA. The algorithm could have arrived by the path on the right so that these do not to have to move down. However, the algorithm evaluates and decides to take the path on the left even if it has to move down, this is because the path on the left is solved in fewer steps and fewer movements. The orange path is the one made by the high–order polynomial and the navigation points obtained from the sequence of movements. This case is solved with ten genes of chromosomes.

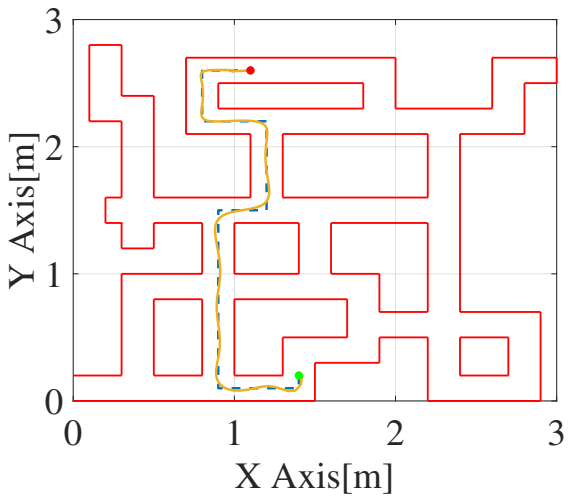

Fig. 10: Case 6.

The fitness value graphs are shown in Fig. 11; these graphs show the evolution of the algorithm until the solution is found. The figure shows the fitness value graphs for the case 1,2,3,4,5 and 6 corresponding to letters a), b), c), d), e) and f). It can be seen that cases one and two corresponding to maze 1 (Fig. 11 a) and b)) are solved in a few less than 20 iterations, because maze 1 has fewer decision points. Also, it can be seen that most of the cases are solved in almost 20 iterations; the only case that took more iterations to solve is case e) corresponding to case 5, where this case is more difficult to solve due to the number of decision points and possible movements to solve the case.

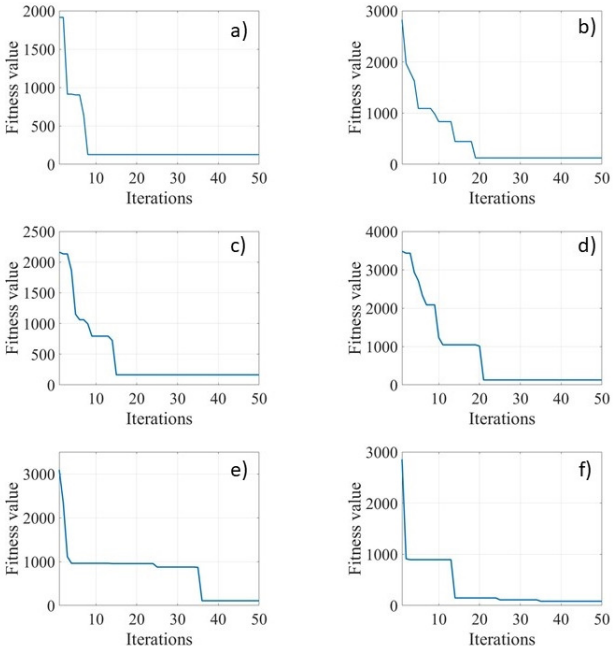

Fig. 11: Fitness value.

B. Simulation

In this subsection the simulations are carried out to validate the proposed algorithm a numerical validation of the trajectories generated with GA and high-order polynomials. The UAV used to obtain the model is a mini-quadrotor that is part of a fleet of aerial vehicles developed in the at CIIIA-FIME-UANL [17]. In the Table I the numerical values of the model parameters are shown.

Parameter	Value	Parameter	Value
c_T	9×10^{-6} $[N-s^2]$	d	0.2 $[m]$
c_Q	3×10^{-7} $[N-m-s^2]$	J_{xx}	15×10^{-3} $[kg-m^2]$
m	0.3 $[kg]$	J_{yy}	15×10^{-3} $[kg-m^2]$
g	9.81 $[m/s^2]$	J_{zz}	30×10^{-3} $[kg-m^2]$

TABLE I: Quadrotor parameters.

The parameters are as follows: c_T and c_Q are aerodynamic coefficients, and d is the distance of the center of gravity of the quadrotor with respect to the axis of rotation of the propellers.

1) Case 1: The Fig. 12 shows the top view of the trajectory UAV performed in the first maze environment. The trajectory executed for the simulation is the same as the reference by the high-order polynomial.

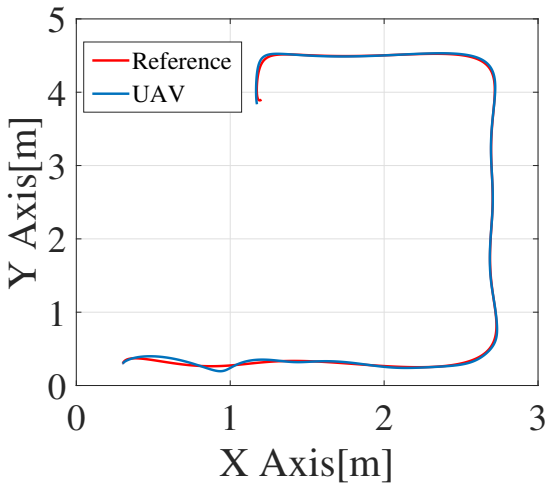

Fig. 12: Trajectory in the XY-axis for case 1.

2) Case 2: The Fig. 13 shows the top view of the trajectory UAV performed and implemented in the first maze-type. Even having four turns in the trajectory, the simulation illustrates that the UAV tracks the reference.

3) Case 3: The Fig. 14 shows the 2D trajectories, the trajectory performed for UAV, and the reference, implemented for the second maze environment. The simulation shows the performance of the trajectory tracked for the UAV.

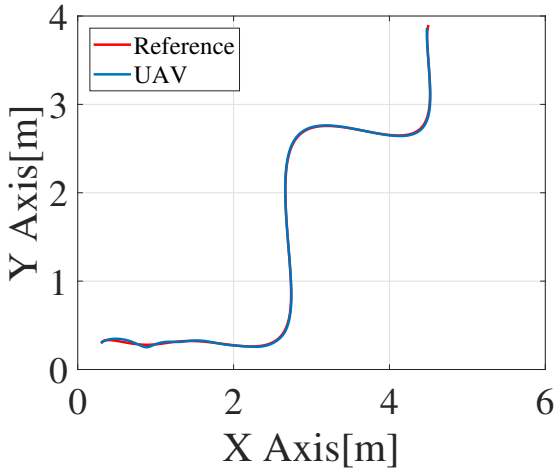

Fig. 13: Trajectory in the XY-axis for case 2.

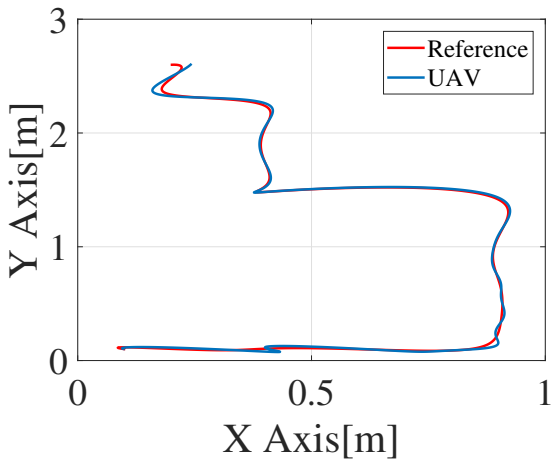

Fig. 14: Trajectory in the XY-axis for case 3.

4) Case 4: The Fig. 15 shows the top view of the trajectory UAV performed. Even having six turns in the trajectory, the simulation shows that the UAV performs the reference signals.

5) Case 5: The Fig. 16 illustrates the 2D of the trajectory UAV performed. As in the previous cases, the simulation demonstrates the tracking of the reference trajectory without any alteration.

6) Case 6: The Fig. 17 shows the 2D trajectories, the trajectory response for UAV and the reference. These 4 cases are considered the second maze-type.

In all above graphics, it can be seen that the trajectories generated from the GA and high–order polynomials can be executed out on a UAV.

C. Implementation

1) Real-time experiments: Real-time experiments are performed for path planing demonstration at known space. Two trajectories are determinated offline and commanded to the quadrotor aircraft by a time-dependent continuous polynomial function.

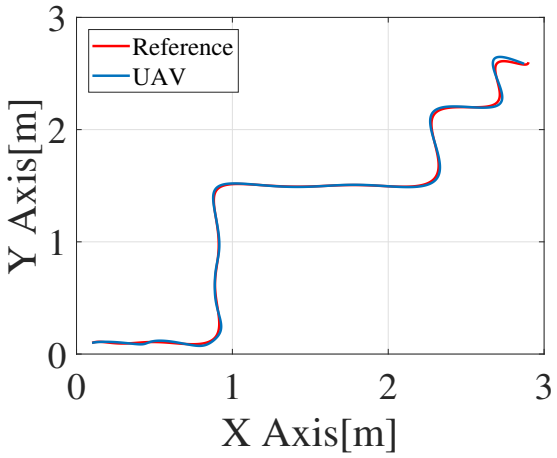

Fig. 15: Trajectory in the XY-axis for case 4.

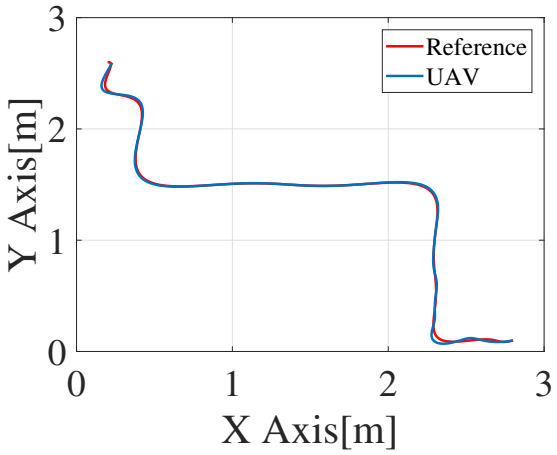

Fig. 16: Trajectory in the XY-axis for case 5.

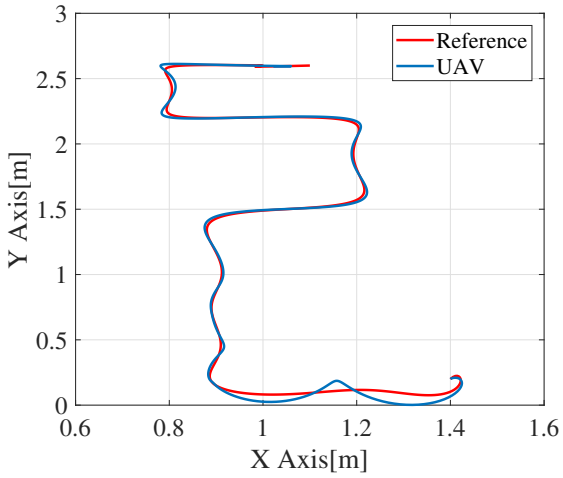

Fig. 17: Trajectory in the XY-axis for case 6.

2) Experimental platform: Small form-factor quadrotor is selected as experimental platform as well as motion capture system. The testing facilities are located at the Navigation Laboratory, of the Aeronautical Engineering Research and Innovation Center (CIIIA), of the Autonomous University of Nuevo Leon (UANL).

The motion capture system consists of 16 VICON T40 cameras, streaming position and orientation data to a centralized computer at $100Hz$, with an accuracy of $0.1mm$ and $0.1°$. The quadrotor aircraft used is designed and built for testing purposes in the same facilities, shown at Fig. 18, as basic characteristics it is based on "X" configuration, with 12.5cm arms and 1.9N thrust for each rotor.

Fig. 18: Quadrotor used for experimental tests.

3) Case 1: The first experimental trajectory is defined by a constant height of $1500mm$, $X = -2000mm$ and $Y = -900mm$ as initial point, $X = 2100mm$ and $Y = 2700mm$ for final one, computing the known maze and the target points is generated a trajectory defined by a high-order polynomial. The Fig. 19 ideal references and actual trajectory performed by the quadrotor are shown. It is possible to note a deviation from ideal and actual trajectory due to response time of the aircraft. It is appreciated ground effect is presented as well as aerodynamic effects. Notwithstanding, an acceptable performance is exhibited for the trajectory generated, completing the mission without collisions.

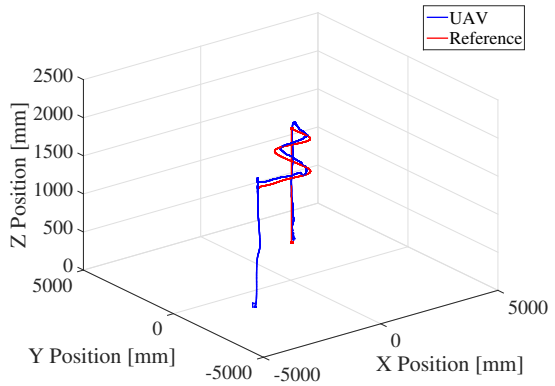

Fig. 19: 3D trajectory for first experimental scenario.

The same scenario is presented in Fig. 20, where the maze walls are shown, the point to stand out is related to trajectory which avoids collisions due to the algorithm based on path planning.

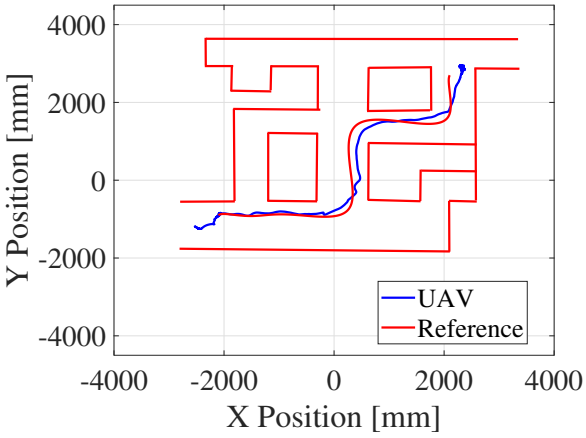

Fig. 20: Trajectory in the XY-axis for first experimental scenario.

Individual axis signals are presented for $X - axis$ at Fig. 21 and Fig. 22 for $Y - axis$. Maze trajectory is initialized approximately at $25s$ and finalized at $45s$, the rest of time is set for take off and landing process. The Fig. 21 and Fig. 22 it is presented the follow-up of the trajectory evolving in time.

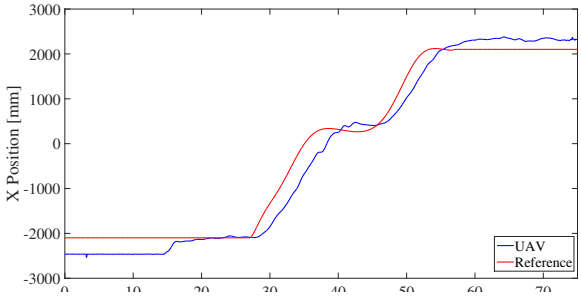

Fig. 21: Trajectory in the X-axis for first experimental scenario.

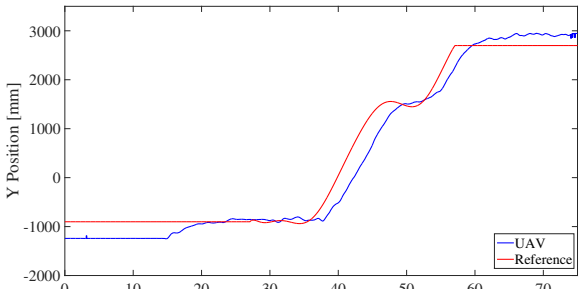

Fig. 22: Trajectory in the Y-axis for first experimental scenario.

4) Case 2: A simple scenario is presented as second experiment, 3 decision points are performed to accomplish the trajectory. A constant height of $1500mm$, $X = -2000mm$ and $Y = -900mm$ as initial point is established, and $X = -1200mm$ and $Y = 2700mm$ for final solution. A trajectory generation is accomplished by the genetic algorithm and implemented using a high-order polynomial. Fig. 23 and Fig. 24 show the ideal and actual trajectory and for 2D perspective in Fig. 24 it is appreciated the maze which is actually the same as experiment 1 but with other final point.

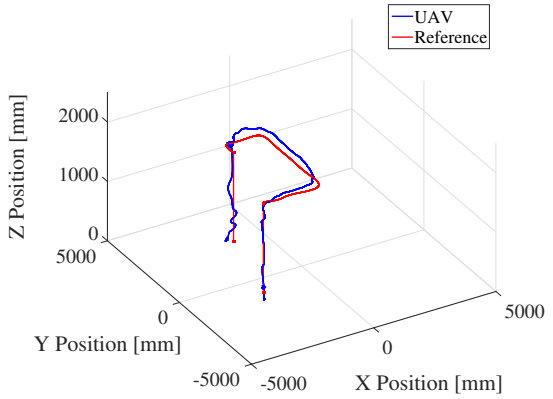

Fig. 23: 3D trajectory for second experimental scenario.

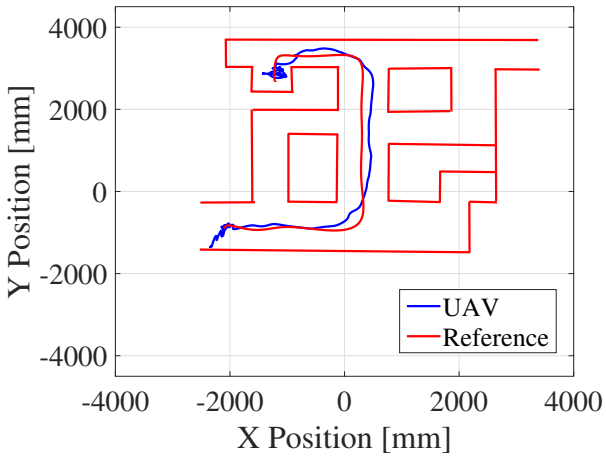

Fig. 24: Trajectory in the XY-axis for second experimental scenario.

The Fig. 25 and Fig. 26 present $X, Y - axes$ respectability for individual signal analysis. Maze trajectory is evolved approximately from $30s$ to $60s$. This is a more aggressive trajectory, showing the effects on $X - Y\ axes$, some disturbance is presented due to aerodynamic effects and navigation control tracking. Nonetheless, the performances exhibited is acceptable for the mission.

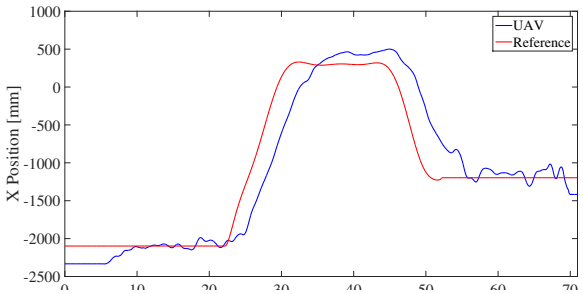

Fig. 25: Trajectory in the X-axis for second experimental scenario.

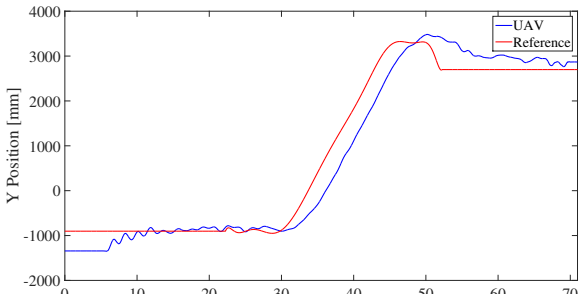

Fig. 26: Trajectory in the Y-axis for second experimental scenario.

This experimental test is captured on video and is available on the following link `https://www.youtube.com/watch?v=uCr2W3Zdnac`

VI. Conclusions

The proposed algorithm has been efficient for path planning in complex maze-like environments. The criteria of change of gen in the chromosome at the decision and the intersection points have solved the six cases proposed with few genes. The fitness value graphs have showed that the minimization of the objective function has been achieved by solving the cases in a few movements. In addition, the method to obtain a trajectory with high order polynomials has generated smooth trajectories so that the UAV quadrotor tracks these paths in real-time.

Acknowledgment

This research work was supported by the Office of Naval Research Global through the grant number N62909-20-1-2030.

References

[1] D. C. Tsouros, S. Bibi, and P. G. Sarigiannidis, "A review on uav-based applications for precision agriculture," *Information*, vol. 10, no. 11, p. 349, 2019.

[2] H. Huang, A. V. Savkin, and C. Huang, "Optimal control of a hybrid uav/train parcel delivery system," in *2019 Chinese Control Conference (CCC)*. IEEE, 2019, pp. 6606–6609.

[3] J. Gu, T. Su, Q. Wang, X. Du, and M. Guizani, "Multiple moving targets surveillance based on a cooperative network for multi-uav," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 82–89, 2018.

[4] E. Rojo-Rodriguez, E. Ollervides, P. Zambrano-Robledo, and O. Garcia, "A fuzzy gain scheduling control algorithm for formation flight of multi-uavs," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 712–720.

[5] I. Mikhaylov and V. Kukhtiaeva, "Algorithm of autonomous uav orientation for applying in complex indoor environment," in *2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*. IEEE, 2017, pp. 943–946.

[6] S. Manfreda, M. F. McCabe, P. E. Miller, R. Lucas, V. Pajuelo Madrigal, G. Mallinis, E. Ben Dor, D. Helman, L. Estes, G. Ciraolo *et al.*, "On the use of unmanned aerial systems for environmental monitoring," *Remote sensing*, vol. 10, no. 4, p. 641, 2018.

[7] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Computer Communications*, vol. 149, pp. 270–299, 2020.

[8] T. T. Mac, C. Copot, A. Hernandez, and R. De Keyser, "Improved potential field method for unknown obstacle avoidance using uav in indoor environment," in *2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2016, pp. 345–350.

[9] T. T. Mac, C. Copot, R. De Keyser, and C. M. Ionescu, "The development of an autonomous navigation system with optimal control of an uav in partly unknown indoor environment," *Mechatronics*, vol. 49, pp. 187–196, 2018.

[10] M. Lupascu, S. Hustiu, A. Burlacu, and M. Kloetzer, "Path planning for autonomous drones using 3d rectangular cuboid decomposition," in *2019 23rd International Conference on System Theory, Control and Computing (ICSTCC)*. IEEE, 2019, pp. 119–124.

[11] B. Song, Z. Wang, and L. Zou, "An improved pso algorithm for smooth path planning of mobile robots using continuous high-degree bezier curve," *Applied Soft Computing*, vol. 100, p. 106960, 2021.

[12] S. Konatowski and P. Pawłowski, "Ant colony optimization algorithm for uav path planning," in *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*. IEEE, 2018, pp. 177–182.

[13] M. Gutierrez-Martinez, E. Rojo-Rodriguez, L. Cabriales-Ramirez, L. Reyes-Osorio, P. Castillo, and O. Garcia-Salazar, "Collision-free path planning based on a genetic algorithm for quadrotor uavs," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 948–957.

[14] N. S. Choubey, "A-mazer with genetic algorithm," *International Journal of Computer Applications*, vol. 58, no. 17, 2012.

[15] Y. Nagata, A. Imamiya, and N. Ono, "A genetic algorithm for the picture maze generation problem," *Computers & Operations Research*, vol. 115, p. 104860, 2020.

[16] J. Kim and S. K. Kim, "Genetic algorithms for solving shortest path problem in maze-type network with precedence constraints," *Wireless Personal Communications*, vol. 105, no. 2, pp. 427–442, 2019.

[17] E. J. Ollervides-Vazquez, E. G. Rojo-Rodriguez, E. U. Rojo-Rodriguez, L. E. Cabriales-Ramirez, and O. Garcia-Salazar, "Two-layer saturated pid controller for the trajectory tracking of a quadrotor uav," in *2020 International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE)*, 2020, pp. 85–91.