E - ISSN : 2686-6269

Bezier Curve Collision-Free Route Planning Using Meta-Heuristic Optimization

Affiani Machmudah [1]*, Setyamartana Parman[2]
[1]Faculty of Advance Technology and Multidiscipline, Universitas Airlangga, Surabaya 60115, Indonesia
[2]Fakulti Teknologi Kejuruteraan Mekanikal dan Pembuatan, Universiti Teknikal Malaysia Melaka, Melaka 76100, Malaysia
[1]affiani.machmudah@gmail.com*; [2]Setyamartana@utem.edu.my
*Corresponding author

ABSTRACT

A collision-free route is very important for achieving sustainability in a manufacturing process and vehicle robot trajectories that commonly operate in a hazardous environment surrounded by obstacles. This paper presents a collision avoidance algorithm using a Bezier curve as a route path. The route planning is modeled as an optimization problem with the objective optimization is to minimize the route length considering an avoiding collision constraint. The collision-avoidance algorithm based on curve point analysis is developed incorporating metaheuristic optimizations, namely a Genetic Algorithm (GA) and a Grey Wolf Optimizer (GWO). In the collision avoidance algorithm, checking of curve point's position is important to evaluate the individual fitness value. The curve points are analyzed in such a way so that only the paths which are outside the obstacle area are selected. In this case, besides the minimum length as a fitness function, the constraint is the position of curve points from an obstacle. With the help of meta-heuristic optimization, the developed collision avoidance algorithm has been applied successfully to different types of obstacle geometries. The optimization problem is converted to the maximization problem so that the highest fitness value is used to measure the performance of the GA and GWO. In general, results show that the GWO outperforms the GA, where it exhibits the highest fitness value. However, the GA has shown better performance for the narrow passage problem than that of the GWO. Thus, for future research, implementing the hybrid technique combining the GA and the GWO to solve the advanced path planning is essential.

Keywords: Bezier Curve; Collision Avoidance; GA; GWO; Sustainable Manufacturing

This is an open-access article under the CC–BY-SA license.

Article History

Received : May, 07th 2021	Revised : May, 31th 2021	Accepted : May, 31th 2021

I. INTRODUCTION

Nowadays, sustainability becomes a general requirement in engineering and sciences, including in path planning technologies. Path planning is very important for an object, which needs to generate motion, such as a vehicle robot, an arm robot manipulator, a missile, a ship, a submarine, and an aircraft. It is a process to design a feasible path that becomes a trace for vehicles or robots.

One of the advantages of using a Bezier curve for path planning is its adaptability to change the control points. It has been widely used as a manufacturing path in additive manufacturing [1,2,3], vehicle route path of an aircraft [4], and vessel path motion [5]. The goal of sustainability in the manufacturing system is still in the way of the process. In the future, additive manufacturing will be developed into sustainable manufacturing via path planning technologies [1]. Thus, research on the path planning methodologies that address a sustainability goal is highly important.

Sustainable manufacturing refers to applying an approach for manufacturing that has minimal harm to environments and mitigated bad impacts on human health and nature [6]. For the robots and vehicle systems, which commonly operate in obstacle environments, it is necessary to achieve the safety goal by generating a collision-free path. Collision avoidance strategies become an important issue in path planning. The concept of collision avoidance dynamic critical area for vessel path planning was introduced in [5]. Collision avoidance for human-robot interactions in adaptive manufacturing was presented in [3]. Saeed et al. [7] proposed a boundary node method to solve mobile robot path planning in static obstacle environments. Kala et al. [8] applied multi-neuron heuristic search to solve the path planning in the static obstacle environment.

The GA as one of the evolutionary methods has been applied to solve the point-to-point path planning. Linquan et al. [9] proposed to use GA for path planning of robot soccer systems in a dynamic environment. Choi et al. [10] presented two-path planning algorithms based on Bezier curves for autonomous vehicles with waypoints and corridor constraints. Lil et al. 11] assumed the existence of an obstacle-avoiding polyline path and replaced the polyline path with a G^2 cubic spline curve to avoid the obstacles.

DOI: 10.25139/ijair.v3i1.3821

This paper presents the collision avoidance algorithm of the Bezier curve path using meta-heuristic optimizations. A procedure of collision detection based on matrix analysis is proposed to solve this problem. The collision checking is performed via curve points analysis. It needs to check whether any positions of curve points inside the obstacle area. The GA and WOA are applied to solve the optimization of the collision avoidance Bezier curve. The performance of both methods in solving the problem of narrow-passage collision avoidance path optimization is also investigated.

II. PROBLEM STATEMENTS

The trajectory is from point ***R*** to point ***S*** with an obstacle between these two points. The objective is to generate a path as a trajectory reference that is free from a collision. A cubic Bezier curve with four control points B_0, B_1, B_2, B_3, is used. A start point is the first control point B_0, and an endpoint is the fourth control point B_3. The other two control points, i.e., B_1 and B_2 are free to be determined. The collision-free requirement needs to place those control points in a location such that the cubic Bezier curve is outside the obstacle area, as seen in Fig. 1.

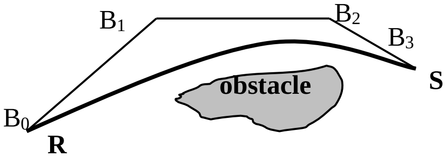


Fig. 1. Trajectory With Obstacle

Mathematically a parametric Bezier curve is defined using Equation (1).

$$P(t) = \sum_{i=0}^{n} B_i J_{n,i}(t) 0 \leq t \leq 1$$
$$J_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}, and \binom{n}{i} = \frac{n!}{i!(n-1)!} \quad (1)$$

where $J_{n,i}(t)$, t and B_i are the i^{th} n^{th} order Bernstein basis function, a curve parameter, and i^{th} control point, respectively.

To construct an n^{th} degree Bezier curve, n+1 control point must be defined. Thus, n is three for the cubic Bezier curve. Equation (2) of the cubic Bezier.

$$\begin{aligned} P(t) &= B_0 J_{3,0} + B_1 J_{3,1} + B_2 J_{3,2} + B_3 J_{3,3} \\ &= (1-t)^3 B_0 + 3t(1-t)^2 B_1 + 3t^2(1-t) B_2 + t^3 B_3 \end{aligned} \quad (2)$$

To avoid collisions, proper selection of control points position is essential to generate the collision-free Bezier curve.

III. METHODS

In two-dimensional cases, the obstacles can be modeled as a composition of one or more kinds of the curve. For example, a triangle is composed of three straight curves. A rectangular is composed of four straight curves, or it can be a composition of other curves such as a polynomial or a trigonometric depending on the geometry of the obstacle object. The collision avoidance algorithm needs analysis of curve points position so that the generated path is outside the obstacle area.

A. Fitness Function

The fitness function for avoiding collision route planning is the path length that needs to be minimized. The constraint function is the position of curve points from the obstacle area. The points in the curve are described by the function *P(t)* based on Equation (3).

$$P(t) = [x(t) \quad y(t)] \quad (3)$$

where *x(t)* and *y(t)* are the *x*-axis and *y*-axis of the cubic Bezier curve, respectively.

The minimum length equation for the parametric curve is described in Equation (4).

$$L = min \int_0^1 \left(\sqrt{\left(\frac{dx(t)}{dt}\right)^2 + \left(\frac{dy(t)}{dt}\right)^2} \right) dt \quad (4)$$

For the cubic Bezier curve, the objective function is the root integral of the fourth-degree polynomial, which has no closed-form solutions. Thus, numerical integration is necessary to solve (4).

The optimum path in a maximize form is the highest fitness value in Equation (5).

$$F_1 = \frac{1}{L} \tag{5}$$

The requirement for obstacle avoidance is the path curve in an obstacle-free area. It needs analysis about the curve point's position to ensure that they are outside the obstacle area. This paper defines a function which is called F_{obs}. It is a function that decides whether the curve is a success or a failure. Only the paths outside the obstacle area for all points are the successful candidates of route planning. It has only two possible values, which are 1 and 0. Mathematically, F_{obs} can be described as an on-off logic operation using Equation (6).

If All curve points are in the free area: F_{obs} =1 (6)
Otherwise: F_{obs} =0

Since it depends on the position of the curve points from the obstacle area, the number of F_{obs} is the same as the number of obstacle areas in the environment. For n obstacles in the environment, there exist *n* F_{obs}s. Thus, the total objective functions can be generated using Equation (7).

$$F_{total} = F_1 \; x \; F_{obs(1)} \; x \; F_{obs(2)} \; x .... x \; F_{obs(n)} \tag{7}$$

Variable F_{obs} describes whether the path is a failure or a success. If any of the paths is a failure, the fitness value becomes zero. For example, if three obstacles are composed of the triangle, the rectangular, and the circular obstacles, there are three F_{obs} values associated with the obstacles. Each of them must be evaluated by checking the position of curve points from each obstacle area.

B. Avoiding Collision method: Curve Points Analysis

To evaluate the function of $F_{obs,}$ the avoiding collision algorithm is proposed as follows:

1. Model the obstacle area. It can be circular, rectangular, a polygon, a curve, or a combination of them. It depends on the obstacle geometry
2. Divide the path, i.e., the cubic Bezier curve, in *n* points. Those *n* points are the points in the curve which its distance from the obstacle must be analyzed.
3. Analysis of each point:
   a. Define a free area and a checking area.
   There are always free areas in the environment that depend on the geometry of the obstacle system.
   b. Evaluate F_{value} for each point in the curve.
   There are two curve point position possibilities. It can be in the free area or the checking area. All points in the free area for which the value is one are considered a passing candidate. There are also the checking areas where the point value could pass or fail. It depends on the position of curve points from the obstacle.
   The value of each point is described as an on-off logic operation as follows:
   - For the curve point in the free area:
     $F_{value}=1$
   - For the curve point in the checking area:
     If the curve point analysis indicates that the point is outside the obstacle: F_{value} = 1
     Otherwise: F_{value} =0
   c. Generate an evaluation matrix**.**
   After the curve points analysis, an evaluation matrix can be generated. The evaluation matrix is the matrix that contains the curve points value. If the cubic Bezier curve is divided into 200 points, the evaluation matrix has a matrix size of 1x200 for one individual. $F_{evaluation} = [F_{v(1)} \quad F_{v(1)} \quad \cdots . \quad F_{v(n)}]$, where $F_{v\,(n)}$ is the F_{value} of point *n.*
   d. Evaluate F_{obs}
   If the Evaluation matrix =1 : F_{obs}=1,
   Otherwise: F_{obs} =0

C. Sample Cases: Five Kinds of Obstacles

The avoiding collision problems for five kinds of obstacles, the rectangular, the circular, the triangular, the area subscribed by two lines and the curve, and the area subscribed by two curves, are presented.

- *Rectangular Obstacle*

  Fig. 2 refers to the rectangular obstacle case. The analysis is For $x_p < a$ or $x_p > b$ or $y_p < c$ or $y_p > d$: F_{value}=1. Otherwise, it is the obstacle area, i.e. F_{value}=1

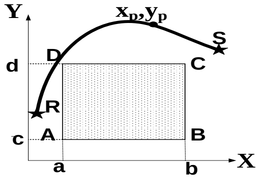


Fig. 2. Rectangular Obstacle

- *Circular Obstacle*

  Fig. 3 shows that for the circular obstacle, the checking area is inside rectangular ABCD. In this checking area, the distance of point *P* to the circle center becomes a tool for curve point analysis using Equation (8).

$$\Delta = \sqrt{\left(x_p - x*\right)^2 + \left(y_p - y*\right)^2} \qquad (8)$$

  Where, (*x**, *y**) is the circle center. If and only if Δ>*R**, then the F_{value}=1. *R** is an obstacle radius.

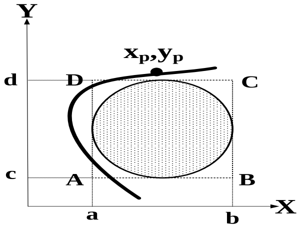


Fig. 3. Circular Obstacle

- *Triangular Obstacle*

  Fig. 4 illustrates the triangular obstacle and its geometrical analysis. It shows that θ_{p1} and θ_{p2} can be used as a tool to analyze whether the point *P* is in the free area or the obstacle area. θ_{p1} and θ_{p2} are point angles from x^*- axis and x^{**}-axis. *(* x^*, y^**)* and *(*x^{**}, y^{**}*)* are obstacle coordinates that coincide with points *A* and *C*. These coordinates are used to simplify the analysis. $\Delta\theta_1$ is the angle between a line AB and a line AC, i.e., $|\theta_2 - \theta_1|$ and $\Delta\theta_2$ is the angle between a line AC and a line BC, i.e., $|\theta_4 - \theta_3|$.

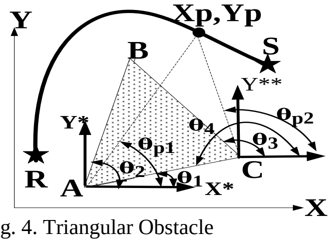


Fig. 4. Triangular Obstacle

The formula for checking procedure can be written as follow:

- If $|\theta_{p1} - \theta_1| > \Delta\theta_1$ or $|\theta_{p2} - \theta_4| > \Delta\theta_2$:
  F_{value}=1
- Otherwise F_{value}=0

where θ_1 and θ_2 are the angle of line AC from x^*-axis and the angle of line BC from x^{**}-axis, respectively. For the case shows in Fig. 4, point *P* is in the free area since the *P* position satisfies the condition of $|\theta_{p2} - \theta_4| \geq \Delta\theta_2$.

- *Two-Straight Lines and One Curve Obstacle*

  Fig. 5 shows the case where the obstacle geometry is composed of two straight lines *AB* and *AC* with coordinate *A* is *(*x_1, y_1*)* and coordinate *B* is (*x_2, y_2*). The curve has an equation $y_1=0.007x^3\sin x + x$. It shows that $\Delta\theta$, i.e., $|\theta_2 – \theta_1|$, can be used as a tool to analyze whether the point *P* is in the free area or the checking area.

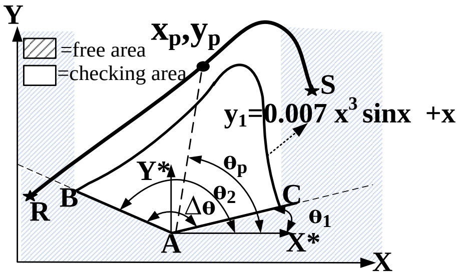


Fig. 5. Two-Straight Lines and One Curve Obstacle

The analysis is as follows

- For the points in the free area : $|\theta_p - \theta_1| > \Delta\theta$ or $x_P < x_1$ or $x_P > x_2$, thus F_{value} = 1
- For the checking area : $|\theta_p - \theta_1| < \Delta\theta$ and $x_1 < x_P < x_2$:
  If $y_p > y1(x_P)$: F_{value} =1
  Otherwise : F_{value} = 0

- *Two-Curve Obstacle*

Fig. 6 shows an example of an obstacle that is composed of two curves with equations y_1=-$x$sin^4x +10 sin x +5 and y_2= 1.5$x$sin^4x – 0.008 x^2. $\Delta\theta$ can be used as a tool to locate the area of checking points. $\Delta\theta$, i.e., $|\theta_p - \theta_1|$, is measured from a line AB. $A(x_1, y_1)$ and $B(x_2, y_2)$ are the intersection points between those curves. The analysis is as follows:

- For the points in the free area: $x_P < x_1$ or $x_P > x_2$
  F_{value} = 1
- For the points in the checking area :
  - For $0 \leq |\theta_P - \theta_1| \leq \pi$ and $x_1 \leq x_P \leq x_2$:
    If $y_p > y_1(x_p)$: F_{value}=1
    Otherwise : F_{value} = 0
  - For $\pi < |\theta_p - \theta_1| < 2\pi$ and $x_1 \leq x_p \leq x_2$:
    If $y_p < y_2(x_P)$: F_{value}=1
    Otherwise : F_{value} = 0

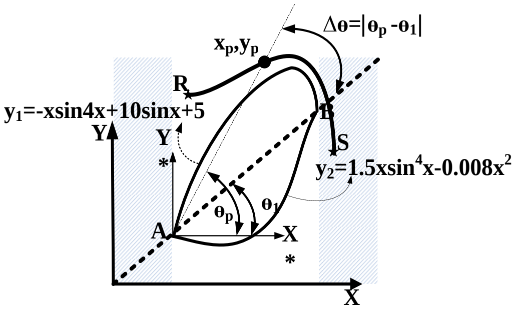


Fig. 6. Two- Curve Obstacle

IV. META-HEURISTIC OPTIMIZATIONS

This section presents meta-heuristic optimizations, namely the GA and GWO, that are used to solve the route planning.

A. Genetic Algorithm

The GA is a searching method in the computation to obtain solutions to optimization based on Darwin evolution theory [12]. The Binary GA needs binary encodings. The chromosome is decoded before an evaluation of the objective function is computed. For the cubic Bezier route path problem, the optimization parameters are the second and third control point positions, i.e. B_1 (x_1,y_1) and B_2 (x_2,y_2) which must be decoded as chromosomes. Fig. 10 illustrates the GA procedure to solve the Bezier curve collision-free route planning.

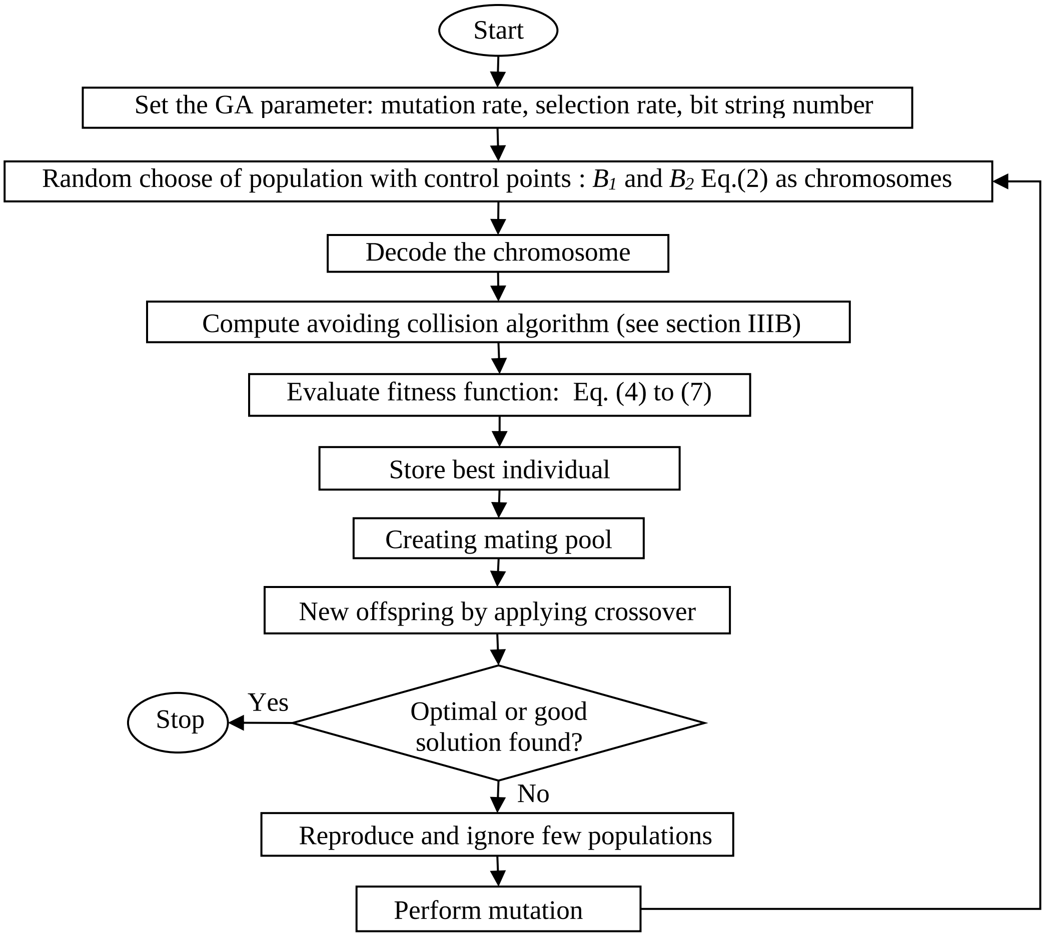

Start
Set the GA parameter: mutation rate, selection rate, bit string number
Random choose of population with control points : B1 and B2 Eq.(2) as chromosomes
Decode the chromosome
Compute avoiding collision algorithm (see section IIIB)
Evaluate fitness function: Eq. (4) to (7)
Store best individual
Creating mating pool
New offspring by applying crossover
Optimal or good solution found?
Yes
Stop
No
Reproduce and ignore few populations
Perform mutation

Fig. 10. Genetic Algorithm for Solving Collision-Avoidance Path

- *Chromosomes*

  Chromosome *C* is expressed by the string using Equation (9).

$$C = [b_1 \; b_2 \; b_3 \ldots\ldots\ldots\ldots b_{Ngene}] \tag{9}$$

  The decimal value of "*b*" elements is determined as Equation (10).

$$X_{decimal} = \sum_{i=0}^{Ngene-1} bix2^i \tag{10}$$

where *Ngene* is the number of bits.

Fig. 7 illustrates an example of a binary encoded chromosome with *Ngene* is 6 bits.

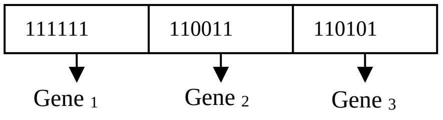


Fig. 7. Gene Representation

- *Decoding*

  The real value of the chromosome is obtained using Equation (11).

$$X = X_{\min} + X_{decimal}\, x \frac{(X_{\max} - X_{\min})}{2^{\lambda} - 1} \tag{11}$$

  Where, $X_{\min}$, $X_{\max}$, and λ are the lower and upper bounds of the side constraints, and the length of the binary string, respectively.

- *Fitness Value*

  The fitness of the individual is the fitness function at its chromosome, which has been decoded. This fitness function represents the objective of the optimization.

- *Mutation and Crossover*

  A crossover is a process of randomly choosing one or more individuals as parents and randomly swapping elements of the parents. Fig. 8(a) and Fig. 8(b) illustrate single-point crossover and two points crossovers.

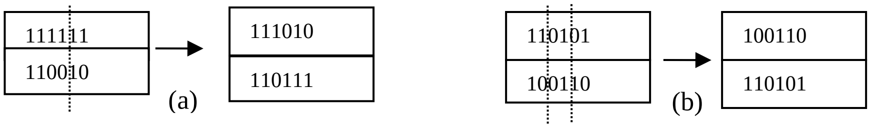


Fig. 8. Crossover

A mutation is a mechanism of randomly change digits in the string, i.e. 0 by 1 and vice versa. Fig. 9 shows an example of this process.

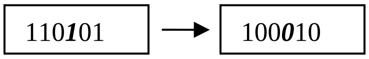


Fig. 9. Mutation

B. Grey Wolf Optimizer

The GWO is the meta-heuristic technique inspired by the grey wolves' leadership hierarchy and hunting mechanism [13]. Four types of grey wolves are considered, namely alpha, beta, delta, and omega. Fig. 11 presents the flow chart of the GWO procedure to solve the Bezier curve collision-free route planning.

The alpha (α) is the fittest solution. The second and third best solutions are beta (β) and delta (δ), respectively. Other candidates' solutions are omega (ω), where they follow the α, β, δ wolves during the hunting procedure. The encircling prey by the grey wolves is modeled using Equation (12).

$$D = \left|\vec{C}.\,\vec{X}_p(t) - \vec{X}(t)\right| \qquad (12)$$
$$\vec{X}_p(t+1) = \left|\,\vec{X}_p(t) - \vec{A}.\vec{D}\,\right|$$

where t, $\vec{A}$,$\vec{C}$, $\vec{X}_p$, $\vec{X}$ are the current iteration, coefficient vector, coefficient vectors the position, vector of the prey, and the position vector of a grey wolf, respectively. Vectors $\vec{A}$, and $\vec{C}$ are defined using Equation (13a), and (13b)

$$\vec{A} = 2\vec{A}.\,\vec{r}_1 - \vec{a} \qquad (13a)$$
$$\vec{C} = 2\,\vec{r}_2 \qquad (13b)$$

$\vec{a}$ are linearly decreased from 2 to 0 for iterations using Equation (13c).

$$a = 2 - k\left(\frac{2}{k_{max}}\right) \qquad (13c)$$

Where, k and k_{max} are the current iteration and the maximum iteration number, respectively, the variables r_1 and r_2 are random vectors in [0, 1]. Grey wolves recognize the location of prey. Alpha is the leader during the hunting mechanism. Following Equations (14) to (16) simulate the hunting behavior of grey wolves

$$\overrightarrow{D}_\alpha = \vec{C}_1.\,\vec{X}_\alpha - \vec{X};\ \overrightarrow{D}_\beta = \vec{C}_2.\,\vec{X}_\beta - \vec{X};\ \overrightarrow{D}_\delta = \vec{C}_3.\,\vec{X}_\delta - \vec{X}; \qquad (14)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1\,.\left(\overrightarrow{D}_\alpha\right);\ \vec{X}_2 = \vec{X}_\beta - \vec{A}_2\,.\left(\overrightarrow{D}_\beta\right);\ \vec{X}_3 = \vec{X}_\delta - \vec{A}_3\,.\left(\overrightarrow{D}_\delta\right) \qquad (15)$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \qquad (16)$$

where $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$ are the positions of alpha wolf, beta wolf, and delta wolf, respectively.

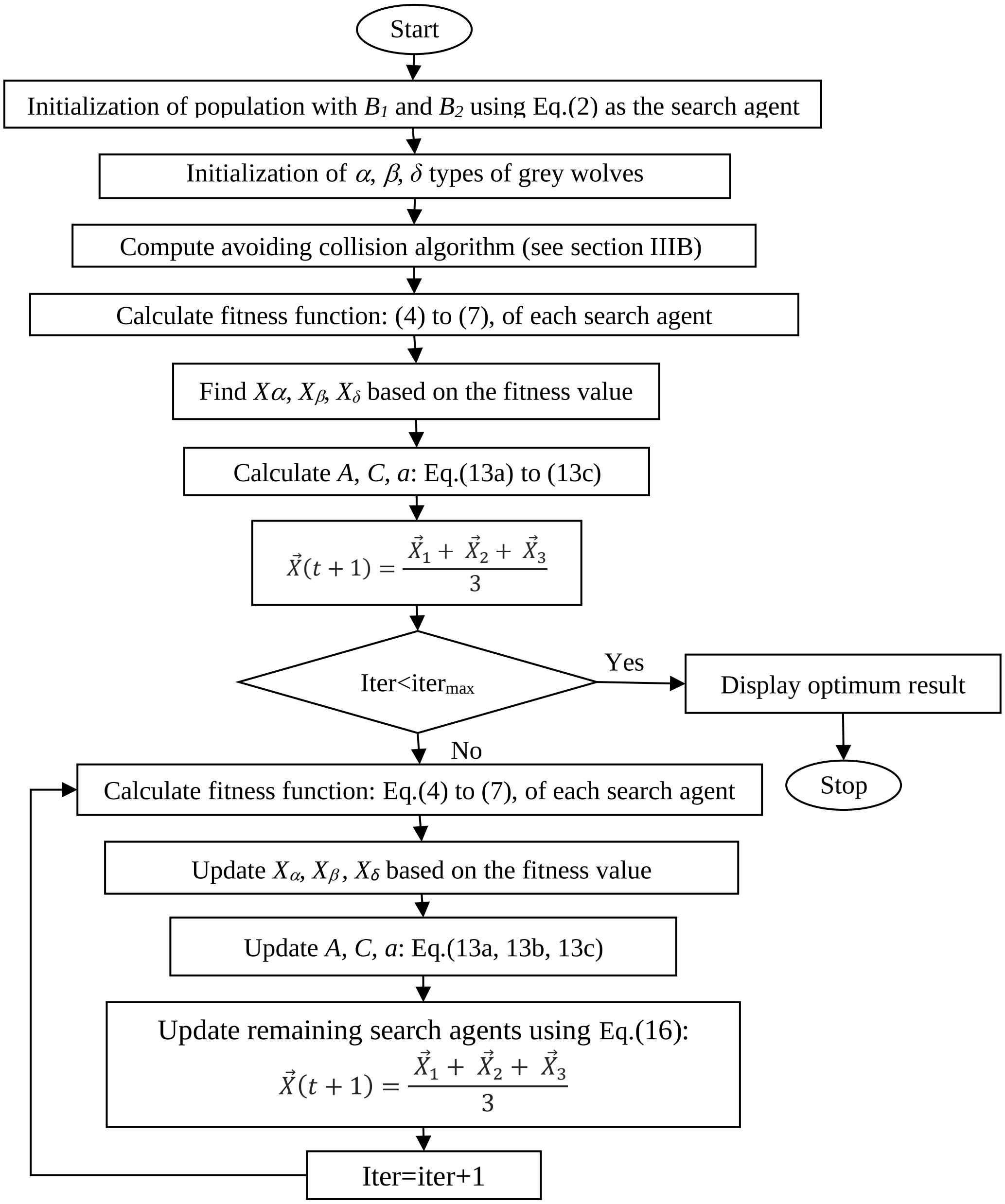

Start
Initialization of population with B_1 and B_2 using Eq.(2) as the search agent
Initialization of α, β, δ types of grey wolves
Compute avoiding collision algorithm (see section IIIB)
Calculate fitness function: (4) to (7), of each search agent
Find $X\alpha$, X_β, X_δ based on the fitness value
Calculate A, C, a: Eq.(13a) to (13c)
$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3}$
Iter<iter$_{max}$
Yes
Display optimum result
Stop
No
Calculate fitness function: Eq.(4) to (7), of each search agent
Update X_α, X_β, X_δ based on the fitness value
Update A, C, a: Eq.(13a, 13b, 13c)
Update remaining search agents using Eq.(16):
$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3}$
Iter=iter+1

Fig. 11. Grey Wolf Optimizer Procedure

$\vec{A}$ is generated randomly within [-2a, 2a]. a is decreased from 2 to 0. $\vec{C}$ contains random values within [0, 2]. The positions of the alpha wolf, the beta wolf, and the delta wolf are updated based on the fitness value following the definition that the alpha, the beta, and the delta are the fittest search agent, the second fittest search agent, and the third fittest search agent, respectively. It needs to check the current fitness value. In the case that the current fitness value is higher than the previous alpha, then the alpha wolf is replaced by the new one. The beta and delta wolves are updated in a similar procedure. Updating the variables: a, A, and c are obtained by using Equation (13a, 13b, 13c). In this mechanism, r_1 and r_2 need to be generated three times to obtained the values of ($\vec{A}_1$, $\vec{C}_1$), ($\vec{A}_2$, $\vec{C}_2$), and ($\vec{A}_3$, $\vec{C}_3$) and to calculate the values of $\vec{X}_1$, $\vec{X}_2$, and $\vec{X}_2$ as Equation (15). The remaining search agents are updated following Equation (16).

V. RESULT AND DISCUSSION

Numerical experiments in MATLAB environment are performed using 20 individuals in the population using 700 curve points. A Simpson's rule is applied as numerical integration to solve F_1. The computation uses 25 iteration numbers. For the GA, the mutation rate and selection rate are chosen to be 0.1 and 0.5, respectively. The number of bits is selected to be 10.

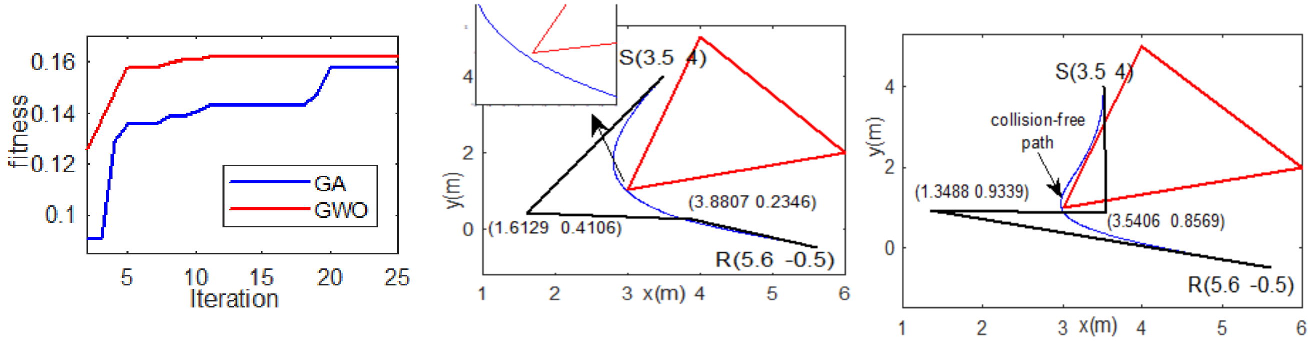
This research uses synthetic data to generate the obstacle environment and the route path using MATLAB software. The programming code is written in the m-file. The synthetic data approach permits an exploration of unusual configurations, unique system behavior, and specific parameter [17]. The synthetic data methodology has been widely applied in the robotics field. Using MATLAB software, the synthetic data has been integrated into the comprehensive systems in solving adaptive trajectory planning [18], Unmanned Aerial Vehicle (UAV) coordination [19], Wide Area Aerial surveillance [17], and route choice problem in transportation science [20]

A. Triangle Obstacle

Table I presents the results of the GA and GWO for route planning from the start point (5.6 -0.5) and the endpoint (3.5 4) in a triangle obstacle environment. It shows that the GWO has better performance than that of the GA since the GWO has the highest fitness value, i.e., 0.01614. As expressed in Equation (5), the minimum length problem has been converted to the maximization problem so that the better performance can be determined by the highest value of the fitness function. Fig. 12a shows the fitness value evolution of the GA and the GWO. It can be observed that within 25 generations, the GWO evolve beyond the GA fitness graph. Fig. 12b and 12c illustrate the collision-free Bezier route obtained from the GA and the GWO, respectively. The results of route planning are in the form of control points (B_{1x}, B_{1y}) and (B_{1x}, B_{1y}) (see Table I), representing the position of lines that determine the Bezier curve geometry. Thus, different values of these control points result in different shapes of the collision-free Bezier curve. Meta-heuristic optimizations search these control points in such a way so that the route path is feasible, i.e., collision-free. These lines are illustrated as the straight line in the black color in Fig. 12b and 12c. The collision-free path, which is illustrated in the blue color in Figs. 12b and 12c are then can be computed by using Equation (2).

TABLE I
RESULTS OF TRIANGLE OBSTACLE

Method	Fitness	Control points			
		B_{1x}	B_{1y}	B_{2x}	B_{2y}
GA	0.15987	3.8807	0.2346	1.6129	0.41056
GWO	***0.1614***	1.3488	0.9339	3.5406	0.8569



(a) (b) (c)

Fig. 12. Result Of Triangle Obstacle (A) Fitness Evolution (B) Path from GA (B) Path From GWO

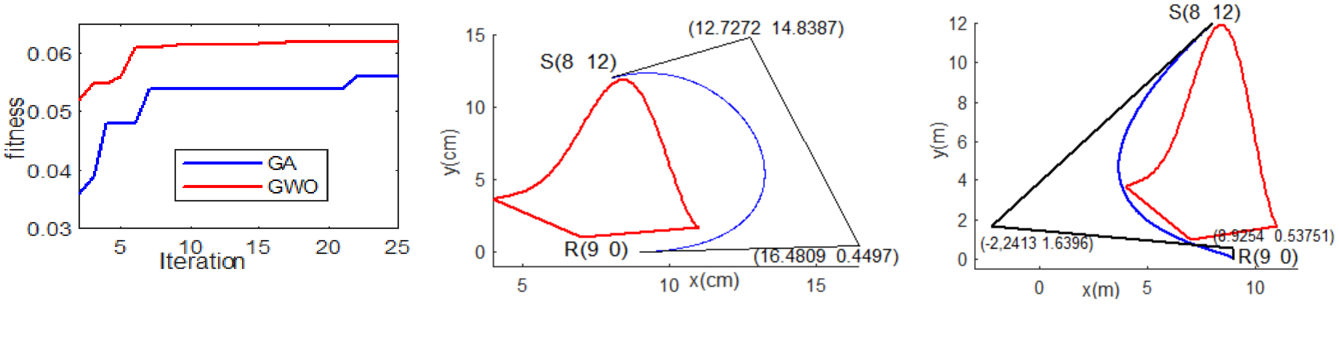
B. One Curve and Two-Line Curve

Table II presents the results of the GA and GWO for route planning from the start point (9, 0) and the endpoint (8, 12) with an obstacle that formed from two straight-line curves. The curve has the equation $Y = 0.007\, x^3 \sin x + 0.001x^2 \cos x + x$. It shows that the GWO has better performance than that of the GA since it has the highest fitness value, i.e., 0.061327. Fig.13a shows the fitness value evolution of the GA and the GWO. The same as in the triangular obstacle, the GWO fitness evolve beyond the GA fitness graph.

TABLE II
RESULTS OF ONE CURVE-TWO-LINE CURVE

Method	Fitness	Control points			
		B_{1x}	B_{1y}	B_{2x}	B_{2y}
GA	0.056481	16.4809	0.449658	12.7273	14.8387
GWO	***0.061327***	8.9254	0.53751	-2.2413	1.6396

Fig. 13a and 13b illustrate the collision-free Bezier route obtained from GA and GWO result for this obstacle environment. It can be observed that in this obstacle geometry, the generated paths obtained from the GA and the GWO are significantly different because the values of Bezier curve control points obtained from the GA and the PSO (see Table II) are in the opposite direction.



(a) (b) (c)

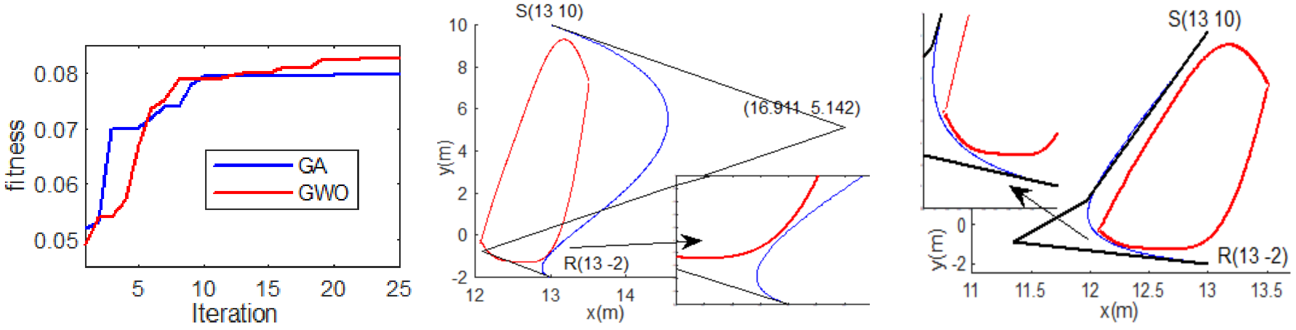
Fig. 13. Result Of Two Straight Line-One Curve Obstacle (A) Fitness Evolution (B) Path From GA (B) Path From GWO

C. Two-Curve Obstacle

Table III presents the results of the GA and GWO for route planning from the start point (13 -2) and the endpoint (13 10) with obstacle formed of two curves $y_1 = -x \sin^4 x + 10 \sin x + 5$ and $y_2 = 1.5x \sin^4 x - 0.008 x^2$. The previous cases show that the GWO has the highest fitness value, i.e., 0.08071. Fig. 14a shows the fitness value evolution of the GA and the GWO. The GWO fitness again evolves beyond the GA fitness graph. Fig. 14b and 14c illustrate the collision-free Bezier route obtained from GA and GWO results for such a two-curve obstacle environment. The same as in the case of one curve two-line obstacle, the generated paths of the GA and GWO are significantly different. Visualization graphs in Figs. 14b and 14c are also clearly indicated that the length of the path from the GWO is lower than that of the GA.

TABLE III
RESULTS OF TWO-CURVE OBSTACLE

Method	Fitness	Control points			
		B_{1x}	B_{1y}	B_{2x}	B_{2y}
GA	0.079292	12.1017	-0.762463	16.911	5.14174
GWO	***0.08071***	11.347	-0.89891	11.96	1.2131



(a) (b) (c)

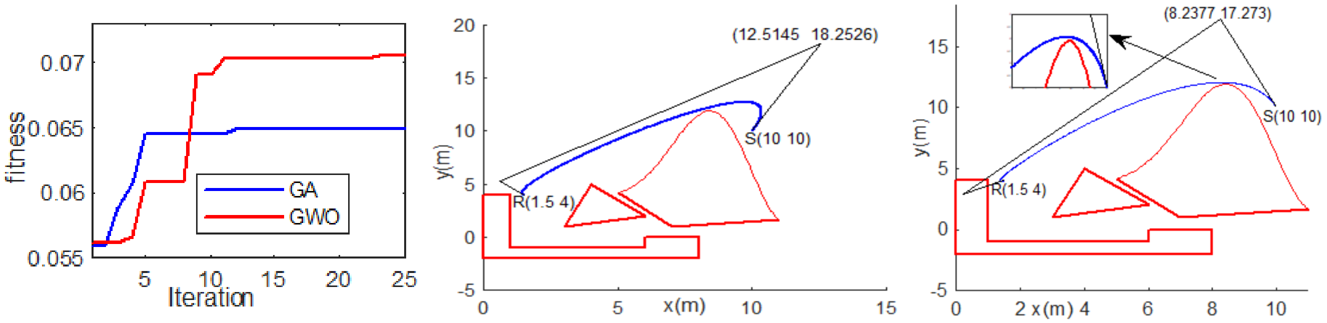
Fig. 14. Result Of Two-Curve Obstacle (A) Fitness Evolution (B) Path from GA (B) Path From GWO

D. Combination Obstacle and Narrow-Passage Collision Avoidances

Table IV presents the results of the GA and GWO for route planning from the start point (1.5 4) and the endpoint (10 10) for combination obstacles (see Fig. 15). As in the previous cases, this table shows that the GWO has better performance than that of the GA, where the GWO best fitness value is higher, i.e., 0.071091, than that of the GA, i.e., 0.064472. The result of the best solution is confirmed by the fitness evolution graph, as shown in Fig. 15a. Applying the route planning result, i.e., Bezier curve control points (see TABLE IV), the collision-free Bezier routes are illustrated in Fig. 15b and 15c for the GA and the GWO, respectively, combination obstacles environment.

The route planning is then modified by changing the starting point to $R(4, 0)$. This new route planning conveys the narrow-passage area. Motion planning involving the narrow-passage problem has been an important topic but a very challenging computational problem [14-16]. Thus, investigating both meta-heuristic optimization methods in solving this type of route planning is essential.

Table V presents the GA and GWO for this route planning which contains the narrow passage area. Unlike previous cases, it shows that the GA has better performance in finding the optimal collision-free route than that of the GWO, where the GA exhibits the highest fitness value, i.e., 0.0561, compared with the best fitness value of the GWO, i.e., 0.053024. Applying the results of the control points in Table IV to Equation (2), Fig. 16a and 16b illustrate the collision-free Bezier route of the GA and the GWO, respectively, for this narrow passage route planning.



(a) (b) (c)

Fig. 15. Result of obstacle combination, case 1 (a) Fitness evolution (b) path from GA (c) path from GWO

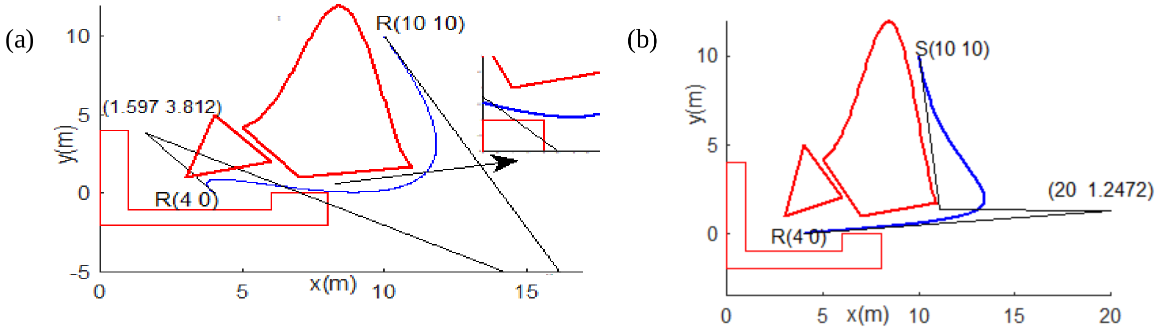


Fig. 16. Result of obstacle combination, case 2

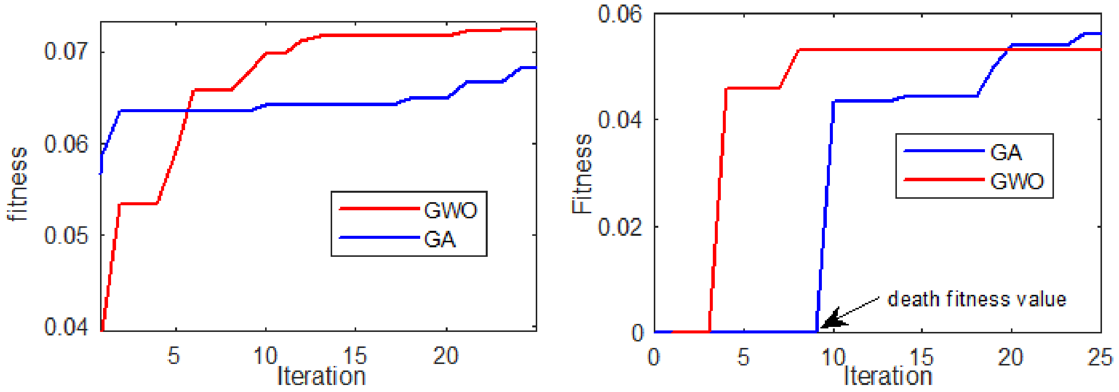


Fig. 17. Fitness value evolution (a) case 1 (b) case 2: involve a narrow passage

TABLE IV
RESULTS OF COMBINATION OBSTACLE, CASE 1

Method	Fitness	Control points			
		B_{1x}	B_{1y}	B_{2x}	B_{2y}
GA	0.064472	0.579014	5.29616	12.5145	18.2526
GWO	***0.071091***	0.22883	2.9223	8.2377	17.273

DOI: 10.25139/ijair.v3i1.3821

TABLE V
RESULTS OF COMBINATION OBSTACLE, CASE 2 (NARROW-PASSAGE)

Method	Fitness	Control points			
		B_{1x}	B_{1y}	B_{2x}	B_{2y}
GA	***0.0561***	1.597	3.812	16.4	-6.92897
GWO	0.053024	20	1.2472	11.153	1.3631

For the narrow passage route planning, both the GA and the GWO encounter the problem of escaping from the death fitness value. As presented in the methodology section, the penalty method is applied. The fitness is set to be a null value when there is a route point inside the obstacles area. The GA and GWO possibly have null fitness values at the beginning of the computation, and they need to escape from this null value. Fig. 17b shows the fitness evolution of both GA and GWO during the narrow passage route planning computation. As a comparison, Fig. 17a shows the fitness evolution of both methods for case 1, where it does not involve the narrow passage area. Without a narrow passage area, both methods do not face the null fitness value problem.

E. Comparison with polynomial-based path

Using the cubic Bezier curve, there are only four unknown parameters, which are the first Bezier curve control point: $B_1(x_1, y_1)$ and the second Bezier curve control point: $B_1\ (x_2, y_2)$, to be searched by the meta-heuristic algorithm. The results have shown that the cubic Bezier curve incorporating the meta-heuristic optimization can be used to solve the narrow passage problem. This section investigates the behavior of a polynomial-based path in solving the presented collision avoidance problem. The polynomial function has been widely used in the path planning of arm robot manipulators when the trajectories are generated in the joint space [21].

The cases presented in this paper are different with arm robot manipulators since the generated trajectories of the present study are in the cartesian space/coordinate. However, the behavior of the polynomial function as the collision-free path is very interesting to be studied because the same with the Bezier curve, the polynomial function has the smoothness property, which is very important in path planning [22]. A quadrinomial [21], which is polynomial degree fourth, is applied as the path in cartesian coordinate in the following:

$$y(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 \tag{17}$$

The boundary conditions of initial coordinates and final coordinates yield:

$$\begin{aligned} a_0 &= y_0 + a_4x_0^4 + a_3x_3^3 + a_2x_0^2 + a_1x_0 \\ a_1 &= \frac{y_f - y_0 + a_4(x_f^4 - x_0^4) + a_3(x_f^3 - x_0^3) + a_2(x_f^2 - x_0^2)}{x_f - x_0} \end{aligned} \tag{18}$$

where (x, y), (x_0, y_0), (x_f, y_f), and a_n are the cartesian coordinates of the path, the initial coordinates, the final coordinates, and *n*th polynomial coefficient, respectively. The path length of y=f(x) is in the following:

$$L = min \int_{x_0}^{x_f} \left(\sqrt{1 + \left(\frac{dy(x)}{dx}\right)^2} \right) dx \tag{19}$$

Variable (x_0, y_0) and (x_f, y_f) are known parameters as the departure point and the destination point, respectively. Using the above equations, there are three searching parameters, which are the fourth polynomial coefficient: a_4, third polynomial coefficient: a_3, and second polynomial coefficient: a_2. Using the GA and the GWO, the computation of the polynomial-based path is similar to the Bezier curve with L in Equation (4) is replaced by Equation (19) because the route is now in the form of $y=f(x)$. The only difference is the generated path, which is now using the polynomial-based function as described in the above equations.

The combination obstacle, namely case 1, presented in the previous section is used as the path planning problem. Table VI presents the results of the GA and the GWO. As in the Bezier curve-based path, the GWO exhibits the highest fitness value, i.e 0.074111.

TABLE VI
RESULTS OF CASE 1, COMBINATION OBSTACLES

Method	Fitness	$[a_4 \quad a_3 \quad a_2]$
GA	0.070343	[-0.00394 0.022641 0.36347]
GWO	***0.074111***	[-0.002396 0.013633 0.12585]

Fig. 18a illustrates the detail of the fitness value evolution during the computation. Fig. 18b shows the polynomial paths obtained from the GA and the GWO. The path results do not look too much different with Figs. 15b and 15c; however, the GA and the

DOI: 10.25139/ijair.v3i1.3821

GWO have failed in finding a collision-free polynomial path for the triangular case, as shown in Fig. 19. Although the obstacle is only one piece of triangular geometry; however, using one segment of the polynomial function is not enough to generate the collision-free path. Compared to the cubic Bezier curve, the polynomial function *y=f(x)* is not agile enough to be used as the collision-free route path.

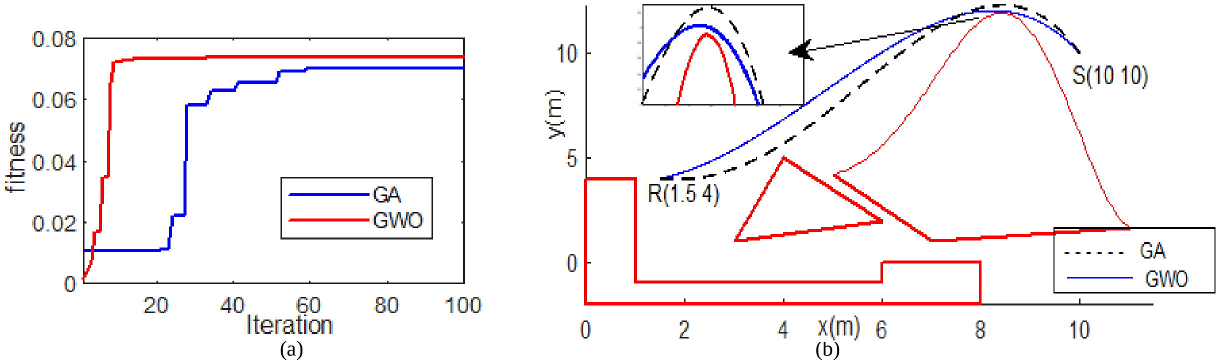


Fig. 18 Result of polynomial-based of combination obstacle, case 1 (a) Fitness evolution (b) path from GA and GWO

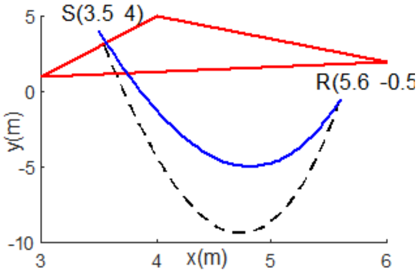


Fig. 19 Result Of Polynomial-Based Of Triangular Obstacle (Failure). The Polynomial-Based Path Is Not Agile Enough.

The results in this paper have shown that, in general, both the GA and GWO have shown good capability in finding the collision-free route applying the developed collision-avoidance algorithm. Without the narrow passage problem, the GWO has shown a good performance than that of the GA, while for the narrow-passage route planning, the GA has outperformed the GWO. The good performance of the GA in escaping from null fitness value is probably due to the mutation procedure. This research uses a quite high mutation rate, i.e., 0.1, where it can further explore the searching capability of the GA. Because of the capability of the GA in solving the narrow passage problem but it less performed than the GWO for the route planning without the narrow passage problem, considering to hybrid between GA and the GWO in solving the advanced planning is also very interesting to be conducted for future research. Investigating other meta-heuristic methods in solving the narrow-passage route planning is also very important to be conducted for future research. Because the results in this paper have shown that one segment of polynomial-based function is less agile for the collision-free route path, it is suggested to study the application of piecewise polynomial functions, for example, the B-spline curve, incorporating the meta-heuristic optimization for the future study. For future research, the developed collision avoidance algorithm can be further implemented to solve the motion planning of an unmanned aerial vehicle and manufacturing path of the industrial manipulator; however, the advanced path planning will require additional constraints to be considered. For example, the other constraint is the maximum curvature for the UAV path planning as described in [23, 24].

VI. CONCLUSION

The collision avoidance Bezier curve optimization has been presented. The avoiding collision algorithm based on curve point analysis was developed. The Bezier control points needed to search in such a way so that the generated route has the minimum length while it was also collision-free. Meta-heuristic optimizations, which were the GA and the GWO, have been applied to solve this route optimization problem. Since the problem has been converted to the maximization problem, the measurement of the best performance was based on the highest fitness value. Results showed that the GWO, in general, has shown better performance than that of the GA, where the GWO exhibits the highest fitness value. However, the GA has shown better performance for the narrow passage problem than that of the GWO. They are considering that the GA has good performance in solving the narrow-passage path planning. At the same time, the GWO outperformed the GA for the case without escaping from the death fitness value. Implementing hybrid GA-GWO to solve the advanced path planning is recommended for future research.

REFERENCES

[1] J. Jiang, Y. Ma Y, “Path planning strategies to optimize accuracy, quality, build time and material use in additive manufacturing: a review,” *Micromachines*, vol. 11, no. 7, pp. 633, 2020

[2] H. Giberti, L. Sbaglia, M. Urgo, “A path planning algorithm for industrial processes under velocity constraints with an application to additive manufacturing,” *J. Manuf. Syst., vol*. 43, pp. 160–167, April 2017

[3] L. Wang, “Collaborative robot monitoring and control for enhanced sustainability,” *Int. J. Adv. Manuf. Technol.*, vol. 81, no. 9, pp.1433–1445, Dec. 2015

[4] M. Zhang, A. Filippone, N. Bojdo, “Multi-objective optimisation of aircraft departure trajectories,” *Aerosp. Sci. Technol.*, vol. *79*, pp. 37–47, Aug. 2018

[5] M. Gil, J. Montewka, P. Krata, T. Hinz, S. Hirdaris, “Determination of the dynamic critical maneuvering area in an encounter between two vessels: Operation with negligible environmental disruption,” *Ocean Eng.*, vol. 213, Oct. 2020

[6] M. Garetti, M. Taisch M, (2012) “Sustainable manufacturing: trends and research challenges,” Prod. Plan Control, vol. 23, no, (2-3), 2019, pp. 83–104

[7] R. A. Saeed, D. R. Recupero, P. Remagnino, (2020) “A boundary node method for path planning of mobile robots,” Robot. Auton. Syst., vol. 123, pp. 03320, Jan. 2020

[8] R. Kala, A. Shukla, R. Tiwari, “Robotic path planning in static environment using hierarchical multi-neuron heuristic search and probability based fitness,” Neurocomputing, vol. no. 74 (14–15), pp. 2314-2335, July 2011

[9] Y. Linquan, L. Zhongwen, T. Zhonghua, Lv Weixian, “Path planning Algorithm for Mobile Robot Obstacle Avoidance Adopting Bezier Curve Based on GA,” in Proc. 2008 Chinese Control and Decision Conference (CCDC 2008), pp. 3286-3289, doi: 10.1109/CCDC.2008.4597937

[10] J. Choi, G. H. Elkaim, “Smoth Path Generation Based on Bezier Curves for Autonomous Vehicle,” in Proc. World Congress on Engineering and Computer Science (WCECS), 2009, ISBN: 978-988-98671-0-2.

[11] Z. Lil, D.S. Meek, D.J. Walton, “A smooth, obstacle-avoiding curve”, *Comput. Graph.*, Vol. 30, pp. 581–587, Aug. 2006

[12] S.N. Sivanandam, S.N. Deepa, *Introduction to Genetic Algorithm*, Spinger-Verlag, 2008

[13] S. Mirjalili, S. M. Mirjalili, A. Lewis, “Grey wolf optimizer,” Adv. Eng. Softw., vol. 69, pp. 46–61, March. 2014

[14] J. M. Lien, J. Keyser, N. M. Amato NM, (2006) “Simultaneous shape decomposition and skeletonization,” in ACM symposium on solid and physical modeling, 2006, pp 219–228

[15] V. Vonásek, R. Pěnička, B. Kozlíková, “Searching Multiple Approximate Solutions in Configuration Space to Guide Sampling-Based Motion Planning,” *J Intell Robot Syst*, vol. 100, pp. 1527–1543, Aug. 2020

[16] A. Varava, J. F. Carvalho, F. T. Pokorny, D. Kragic, “Free space of rigid objects: Caging, path non-existence, and narrow passage detection. In Workshop on algorithmic foundations of robotics, 2018, pp. 19-35

[17] E. J. Griffith, C. Mishra, J. F. Ralph and S. Maskell, "A system for the generation of synthetic Wide Area Aerial surveillance imagery", *Simulation Modelling Practice and Theory*, vol. 84, pp. 286-308, 2018.

[18] C. Yuan, H. Cai, “Spatial reasoning mechanism to enable automated adaptive trajectory planning in ground penetrating radar survey,” *Autom. Constr.*, vol. 114, pp. 103157, 2020

[19] A.L. Alfeo, M.G. Cimino, G. Vaglini, “Enhancing biologically inspired swarm behavior: Metaheuristics to foster the optimization of UAVS coordination in target search,” Computers & Operations Research, vol. 110, pp. 34-47, 2019

[20] Zimmermann M., Frejinger E., “A tutorial on recursive models for analyzing and predicting path choice behavior,” EURO Journal on Transportation and Logistics, vol. 9, pp. 100004, 2020

[21] B. I. Kazem, A. I. Mahdi, A. T. Oudah, "Motion Planning for a Robot Arm by Using Genetic Algorithm", *Jordan Journal of Mechanical and Industrial Engineering*, vol. 2, no. 3, pp. 131-136, 2008.

[22] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms In MATLAB*, Berlin: Springer, 2017

[23] M. Shanmugavel, A. Tsourdos, B. A. White, R. Zbikowski, “Differential Geometric Path Planning of Multiple UAVs,” *ASME Journal*, Vol. 129, No. 11, pp. 620-632, 2007.

[24] M. Shanmugavel, A. Tsourdos, B. A. White, R. Zbikowski, “Co-operative Path Planning of Multiple UAVs using Dubins Paths with Clothoid arcs” *Control Engineering Practice*, Vol. 18, pp. 1084-1092, 2010.