Bee Colony Optimization for Simplified Space Flight Trajectories

1st Vincent Broda
EECS Department
University of Tennessee
Knoxville, United States
vincentbroda@gmail.com

2nd Emma Brown
EECS Department
University of Tennessee
Knoxville, United States
emmagrace0429@gmail.com

3rd George Evans
EECS Department
University of Tennessee
Knoxville, United States
gevans16@utk.edu

***Abstract*—This project investigates whether the application of bio-inspired optimization can address interplanetary distance problems. We work on solving key problems in space trajectory and flight, some of which are fuel usage and gravity of the spacecraft. Our initial research started with particle swarm optimization, specifically ant colony optimization. After analyzing the behavior of the ant colony optimization through literature research, we finally decided to use artificial bee colony optimization. This is mainly due to the ability to use onlooker bees, scout bees and employed bees. These bees serve as a distinct process to balance exploration and exploitation of the search space at hand. Scout bees are responsible for exploring new areas and bringing back the information to the hive. The employed bees are responsible for developing known high-quality solutions. The onlooker bees will evaluate the quality of the food source that was gathered by the employed bees and determine whether to exploit the food source or decide to find alternative sources. For this research, we decided to use four planets in our solar system and the sun as our parameters. In order to get more of a dynamic system, we added gravitational pulls to each of our parameters to add more complexity to our simulations.**

***Index Terms*—Bee, optimization, spaceflight**

I. Introduction and Motivation

Spaceflight is the penultimate problem for extreme design constraints, which makes it an attractive target for optimization functions. To limit the necessary resources included in any spaceflight, all fuel and movements should be kept as minimal as possible. Apart from more traditional optimization algorithms, researchers have also used bio-inspired computational techniques to uncover novel solutions to spaceflight problems. Acciarini et. al. used ant colony optimization to solve a series of benchmark spaceflight optimization problems, such as Earth to Venus flights which include wait times around planetary bodies. Their ant colony optimization approach was competitive with contemporary spaceflight optimization algorithms [1].

However, there are other types of swarm optimizations that take inspiration from other biological sources. For example, artificial bee colony (ABC) optimization takes inspiration from the structure of bee colonies to develop a solution to a problem similar to the ants. There is generally a generated set of food sources, each with particular rewards, which the bees attempt to build pathways and optimized routes to harvest. In contrast to the ant swarm optimization approach, the computational agents are separated into bees with three different roles. There are onlooker bees, scout bees, and employed bees. The scout bees randomly search the space, and report new food sources that have been located. The employed bees follow routes to the known food sources, and report the quality of the food source based on a fitness function. The onlooker bees take in the information provided by the employed bees, and decide to either exploit a food source or move on [2]. There are also more advanced versions of bee colony optimization (BCO) which make use queen bee behavior, simulated pheromones, and reproduction [3].

The main motivation of our project is to calculate the interplanetary distances while taking into account the effect of gravity on space travel. We are interested in exploring whether particle swarm optimization can effectively address these challenges. The reason for choosing particle swarm optimization methods is mainly due to the ability of these optimizations to adapt to complex problems. These specific algorithms are able to search space areas that are non-liner and high dimensional. Both of these elements that were mentioned, make particle swarm optimization well-suited for modeling and optimizing complex interplanetary travel. Since space missions require efficient trajectory planning to help minimize fuel usage and travel time, it is also important to consider the significant impact of the gravitational dynamics of the solar system planets.

II. Related Works

In the early stages of our project, we explored bio-inspired optimization methods, specifically particle swarm optimization. The initial focus was on the optimization of ant colonies. An influential paper that showed biological-inspired approaches was the "MHACO: a Multi-Objective Hypervolume-Based Ant Colony Optimizer for Space Trajectory Optimization," which demonstrated how effectively the ant optimization approach can be applied to space trajectory. The ant colony optimization is useful for space trajectory mainly for its flexibility, adaptability, and its ability to solve difficult problems. This adaptability was important for the space trajectory mainly because of the complexity and uncertain environments. These characteristics of this algorithm

Identify applicable funding agency here. If none, delete this.

are critical when it comes to solving problems in astrodynamics. Even though the ant colony optimization was a strong algorithm to use for space trajectory, we shifted our focus to the artificial bee colony optimization also known as ABC optimization. This switch was mainly due to the conceptual similarities between this paper and our current project. [1]

There are many different PSO methods that can be utilized, not just ants. Some papers have made use of schools of fish or flocks of birds, and each has their own benefits. However, as a whole, PSO methods present some clear differences compared to other evolutionary approaches. Bessette et. al. describe their experiment testing three different evolutionary approaches to a Low Earth Orbit optimization problem. They found that PSO had the second best scores on the optimization problem itself, but that the PSO also converged the fastest and had the best reliability metrics compared to the other methods. It was thus deemed the best option for the optimization at hand [4]. For our approach, we felt that PSO techniques were still a viable option for the spaceflight trajectory problem, but that a shift in the type of PSO was the best path for the project to take.

Once we shift our focus to artificial bee colony optimization, we examine related research on applying the artificial bee colony optimization algorithm to shortest paths. One paper that stood out was the paper "Solve Shortest Paths Problem by Using Artificial Bee Colony Algorithm" by Mansouri, Asady, and Gupta. This paper discusses the foraging behavior of the bees. The paper demonstrates how the bees are able to adapt to explore the critical space and exploit the paths in the network. The paper illustrates the Artificial Bee Colony Optimization's ability to balance the exploration and exploitation qualities of the algorithm. The balance that is discussed is an essential characteristic for solving complex problems, which aligns quite closely with the problems related to our project. [5]

As we continued to further our research on the topic of bio-inspired optimization techniques, the paper "Path Planning Algorithms in the Autonomous Driving System: A Comprehensive Review" became a reference for learning and understanding Artificial Bee Swarm Optimization. The paper discusses different advancements that have been made to the Artificial Bee Colony Optimization. Some of these modifications include the Arrhenius ABC, the Adaptive Dimension Limit-ABC, the Coevolution Global best Leading ABC, and lastly the improved hybrid iABC–EP. The Arrhenius ABC modification incorporates a temperature-based probabilistic model that helps enhance the algorithm's exploration. The Adaptive Dimension Limit-AB modifications add a parameter dynamic to enhance the performance of convergence in the algorithm. The Coevolution Global best Leading ABC uses subpopulations to prevent premature convergence. Finally, the improved hybrid iABC-EP combines artificial bee colony optimization with evolutionary programming to find more specific solutions for diversity. Every one of these enhanced optimizations aim to improve the performance, speed, solution quality and adaptability to complex challenging environments. [6]

Building on our knowledge on Artificial Bee Colony Optimization (ABC), we examined another related study that modifies ABC optimization. In the paper " A Modified Artificial Bee Colony Algorithm," this paper demonstrates the modifications of the Artificial Bee Colony Optimization and illustrates the modified behaviors. The researchers changed the mechanisms of the search to be more chaotic, which was accomplished by using logistic mapping function and Tent mapping. This change in search introduced the functionality of having controlled randomness behavior. Additionally, the researchers changed the learning to be opposition based. This technique allows the algorithm to be able to try to evaluate a possible solution while simultaneously checking the opposite of that solution at the same time. The opposition based learning modification effectively allows the algorithm to have double the exploration efficiency. These modifications help the algorithm explore more deeply, and the change in behavior allows the algorithm to improve the diversity of the population and the quality of the solutions. [7]

As we continue to investigate the capabilities and applications of artificial bee colony optimization, through the paper "Artificial Bee Colony Algorithm to Optimize the Safety Distance of Workers in Construction Projects," we gathered a better understanding of how the ABC optimization can be applied to real life applications. In addition to discussing the general outlines of the underlying behavior of ABC optimization, this paper explores how to apply artificial bee colony optimization to determine how far to build a solution. The key concept that the paper discusses is how the researchers were able to achieve the distance. The way that they achieved this idea was having each bee represent as a possible solution and the bee moves by comparing its current position in the space to another solution that is located nearby. The difference that is gathered from the different positions act as a distance in the exploration space. The bees will use this difference to help decide where to go next. The use of finding the distance helps the algorithm to balance exploration and exploitation. Since our approach involves finding the best paths for interplanetary distances, this paper improved our understanding on how we can apply ABC optimization for distances. [8]

During our research on the context of trajectory for space travel, the paper "Benchmarking different global optimization techniques for preliminary space trajectory design" became a paper of crucial information which provided vital insights on this topic. The work discussed in this paper introduces a set of representative benchmark cases based on real-world mission scenarios. The scenarios include the Cassini, Rosetta, and MESSENGER missions, and the authors highlight the challenges of space complexity. The researchers of this paper then evaluated the performance of several optimization techniques. The techniques that were evaluated are Genetic Algorithms, Differential Evolution, Adaptive Simulated Annealing, GLOBAL algorithm, and finally Particle Swarm Optimization. This testing provided a comprehensive analysis on the strengths and weaknesses of the algorithms in the context of space trajectory. The findings that were provided in this paper helped us decide that the approach Particle Swarm

Optimization was the correct path for us to solve the complex challenges that are faced in space trajectory designs. [9]

III. METHODS

For this project, we constructed three main components to form the simulation and agents. The first two are the planetary orbit simulator and a rocket controller that is built on top of the simulator. The last is the BCO section, which again is built upon the planetary simulator, but takes some ideas from the rocket controller. All of the pipeline was made using python 3.13.1.

A. Planetary Orbit Simulator

The planetary orbit simulator provides a 2D representation of planetary orbits with several options. The key equations for this section are given by Eqs. 2-14. Although standard gravitational equations such as the Universal law of gravitation, given by Eq. 1 would use $G = 6.67430 * 10^{-11}$, but for simplicity we have opted to use a modified version given by Eq. 2. This G_{AU} is in terms of Astronomical Units, where s_{day}^2 is the seconds in a day, and km_{AU} is the kilometers in an AU [10].

$$F = G\frac{m_1 m_2}{r^2} \tag{1}$$

$$G_{AU} = 6.67430e^{-11}\frac{s_{day}^2}{(km_{AU}1000)^3 M_E} \tag{2}$$

Because not all orbits are perfectly circular, orbit equations use the measurements of ellipticals, which decompose into circular equations if the eccentricity e is 0. The eccentricity can be found via Eq. 4, a component of which is given by the semi major axis of the elliptical given by Eq. 3. The terms aphelion and perihelion refer to the points where the object is farthest and closest to the star, respectively.

$$a = \frac{(r_{aph} + r_{per})}{2} \tag{3}$$

$$e = \frac{(r_{aph} - r_{per})}{(r_{aph} + r_{per})} \tag{4}$$

$$M = E - e * sin(E) \tag{5}$$

Kepler's Equation is given in Eq. 5, also called the mean anomaly relation. It uses the eccentricity e with the eccentric anomaly E to find the mean anomaly M. This equation is used in often in orbital mechanics of a body [11]. The Newton-Raphson algorithm, given by Eq. 6, is a method of approximating solutions ot complex equations, and thus can be used for the orbital mechanics equations to solve for behavior [12].

$$E_{n+1} = \frac{E_n - (E_n - e * sin(E) - M)}{1 - e * cos(E)} \tag{6}$$

With some of the background equations, we can then solve for actual orbital values. We can find the true anomaly and radius by applying terms such as e and E in Eqs. 7 and 8.

$$\theta = 2\text{atan2}(\sqrt{1+e}\sin(E/2), \sqrt{1-e}\cos(E/2)) \tag{7}$$

$$r = \frac{a(1-e^2)}{1e\cos\theta} \tag{8}$$

We can also use our gravitational constant G_{AU} to find the orbital period via Eq. 9 and the motion via Eq.10. A body's position in a circular orbit is given by Eq. 11 [13].

$$T = 2\pi\sqrt{\frac{a^3}{G_{AU}M_C}} \tag{9}$$

$$n = \frac{2\pi}{T} \tag{10}$$

$$r(t) = [a\cos(nt + M_0), a\sin(nt + M_0)] \tag{11}$$

Since we are applying gravity in our simulation, it is important to note the gravity present in the total system. The gravity applied by the star in the system is given by Eq. 12 and the planets are totaled by a summation equation shown in Eq. 13. These add together to a total gravity g_{tot} shown in Eq. 14.

$$g_{star}(p) = \frac{G_{AU}M_C(-p)}{||p||^3} \tag{12}$$

$$g_{pl}(p) = \sum_i \frac{G_{AU}M_i(r_i - p)}{||r_i - p||^3} \tag{13}$$

$$g_{tot}(p) = g_{star}(p) + g_{pl}(p) \tag{14}$$

The script itself takes in various input from the command line, including orbit shape, the number of earth years to simulate, and the time step in days. It also takes a path command, which the system will then calculate and use to judge the agents. The distant units are AUs, mass is units of Earth's mass M_E, and angles are in degrees, which where chosen to help minimize the possibility of floating point number errors. From our equations, we require the radius, mass, aphelion, perihelion, and the mean anomaly. We could not find the most up to date mean anomaly of Earth, so we chose a random value. We also made use of the mass M_S and radius r_S of the Sun for our center body. The information of the simulation is saved and cached so later steps can just load in the information and not need to calculate it every time.

B. Rockets

For the rocket class, an init scrip, a simulation script, and some test demo scripts were made. Only the init and simulation ones are used for the BCO so they will be the only ones mentioned here. To construct a rocket, we generated an initial position, initial velocity, mass, max thrust, max km/h, and made use of the simulation to access the gravity equations. The units are still using AUs and M_E, but there are some helper functions to do some conversions between units. There are also some key attributes, such as the position and velocity vectors, the throttle (between 0-1 of max throttle), the direction angle in rads, the path for plotting, and a flag if the rocket has landed or collided with a body. As the system iterates each step, the velocity is updated based on the throttle x, velocity angle ϕ, and gravitational acceleration g. This process is displayed by Eq. 15.

$$v_{n+1} = v_n x \cos\phi, \sin\phi + gt \tag{15}$$

The landed flag checks to see if there is a collision with a celestial body. This method sometimes had issues, likely due to gravity calculations becoming exponentially larger as two objects grew closer together. For each step, this process determines how the rocket, also known as the bees or agents for this experiment, move in the environment.

C. Bee Colony Optimization

The bees, each of which is an agent in the swarm optimization, begin at a starting planet with an appropriate velocity based on that planet. The bees require multiple attached attributes to make the simulation function. They use velocity, fitness score, trial to improve logic for scouting, path and path length, their individual role, and whether they have finished. The bees' roles are split between employed, scout, and onlooker based upon input parameter ratios. Using those ratios, the simulation first sets the number of scouts, then the ratio of employed bees from the remainder, and the rest are onlookers. The bees do not change types dynamically, but can change their behavior. For example, after a certain number of steps, the scouts will abandon bad trajectories and search for new ones in random directions. Typically the bees might be able to swap roles in BCO, but we were unable to implement that functionality for this experiment.

For each step the bee takes, their position is calculated based on the methods described in the rocket simulation. Their position is updated based on their velocity, which is made up of several components, and they must pass a check to see if they have collided with another planetary body The fitness score can be broken down into three main parts: the path length, the food path bonus, and the close to orbit bonus. The bee receives a bonus in its fitness score if it reaches the destination planet's orbit, even if it has not reached the planet itself. This bonus increases based on proximity to the planet. The bee also receives a bonus based on Eq. 16 if the path length is under 1.20 times the straight line path. This is applied as a penalty if the path length exceeds the 1.2 times the straight line length threshold but the penalty is simply (Path Length)/(Path Length Threshold). Foot batch bonuses are given around the orbit, but these are pretty small since they can be far away from the planet. The bee receives the biggest bonus if the bee makes it to the destination and the path length is within a small threshold.

$$\text{Path Length Bonus} = 1 + \frac{\text{Path Length}}{\text{Path Length Threshold}} \tag{16}$$

Employed bees compute their own heading based on their velocity, and the target heading, which in our case is the real planet orbit location. Then they blend the headings with some random inertia, staying roughly where they want to go with a minor attraction to the food patch and random variance added. They update their velocity and move according to the velocity changes made each step discussed earlier. They have a random chance to reset based on their current fitness as well. The reset mechanic works via a trial, which increments whenever they fail to improve fitness, and resets to zero with probability equal to their new fitness. So higher-fitness bees get "trial reset" more often, allowing them to continue exploiting high yield sources. This allows bees to continue even if their path is poor, but the worse fitness scores they get, the more likely they are to reset.

The onlooker bees will wait in the hive and watch the employed bees fitness values. Employed bees only share their fitness scores with the onlooker bees. Onlooker bees then use roulette-wheel sampling over those fitnesses to pick one employed bee, and copy that bee's velocity. The onlooker bee will not generate any original velocities, they will exploit paths already found by employed bees.

The scout bees handle the exploration by abandoning stagnated food sources. They do a stagnation check if their fitness has not improved in a number of steps, and will choose random directions to explore based on the reset mechanic. They do not inherit from any other roles, and continue to re-explore throughout the simulation.

D. Parameter Search

Number of Bees	20	30	50	75
Employed Ratio	0.1	0.25	0.5	0.75
Scout Ratio	0.1	0.15	0.25	0.5
Extra Food Sources	0	3	5	8

TABLE I: Parameter values used in permutation search

To test the impact of different parameters on the space trajectory optimization problem, we ran a permutation search of the following parameters: number of bees, employed bee ratio at initialization, scout bee ratio at initialization, and the number of extra food sources. The values of the parameters explored are given in Table I. The parameters are increased exponentially where viable, in order to determine a more broad range of behaviors and possible trends.

IV. Results

The best path achieved by a network throughout the parameter search is shown in Fig. 1. The outermost planet had the highest yield food source, so the network was successful in reaching the desired target. We can also see that the bees successfully navigated though the system's gravity, with the biggest pull seeming coming from the Sun in the center. This result aligns with expectations, since the Sun has the highest mass by far.

To examine the results of the permutation search, we can examine a set of correlation heatmaps, all plotting each

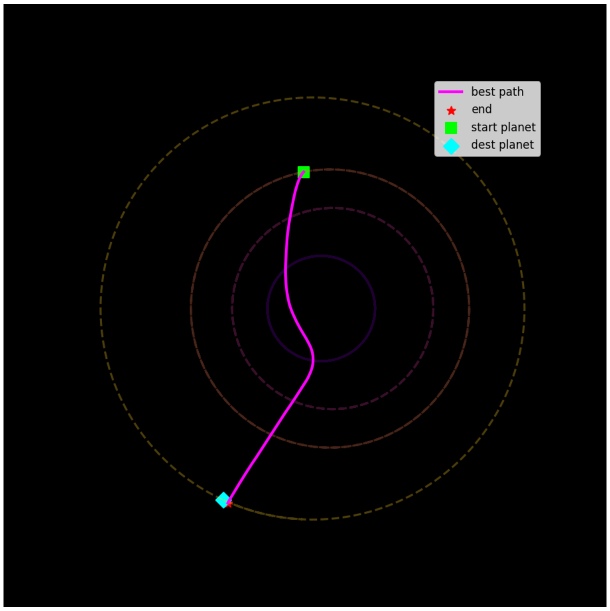

Fig. 1: Best path achieved during the parameter search

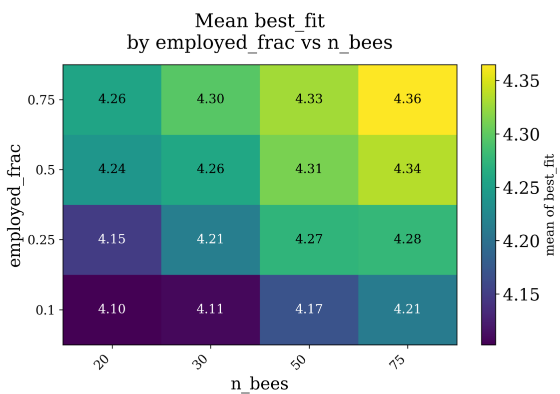

Fig. 2: Correlation heatmap between the number of bees and the fraction of the initial population with the role employed

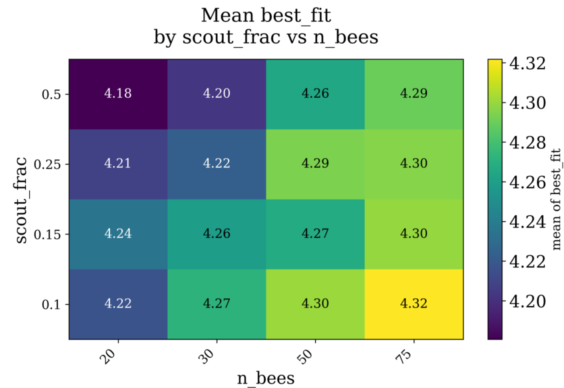

Fig. 3: Correlation heatmap between the number of bees and the fraction of the initial population with the role scout

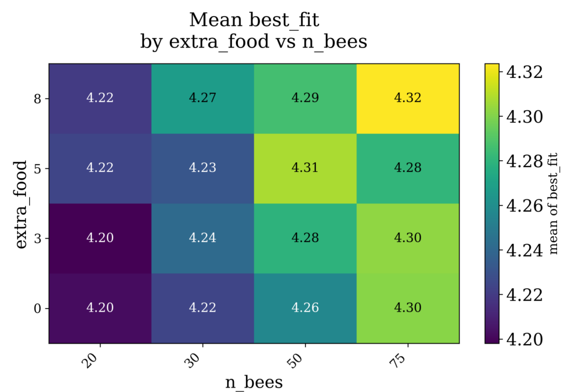

Fig. 4: Correlation heatmap between the number of bees and the extra food sources

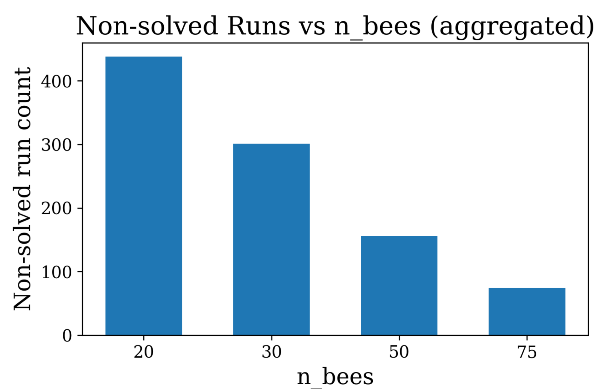

Fig. 5: Number of unsolved runs for each value of the number of bees

parameter of the permutation search against the number of bees parameter. The number of bees displayed the most stable and linear relationship, which made it a good parameter to compare the effect of other parameters against.

The number of bees directly impacts the performance of the network, and Figs 2-4 each show that as the number of bees increases, the performance of the network improves. As the number of bees increase, Fig. 5 shows that the number of unsolved networks also decreases.

Examining Fig. 2, the heatmap shows that as the ratio of employed bees in the initial population increases, the performance also increases. Conversely, Fig. 3 shows that as the ratio of scouts in the initial population increases, the performance decreases.

The number of extra food sources, compared against the number of bees parameter in Fig. 4, has little to no impact on the fitness score of the network.

V. Discussion, Conclusions, and Future Work

The inverse effects of the employed bee ratio and the scout bee ratio parameters can be explained by the nature if an agent network and the size of the search space. Having a higher ratio of employed bees at the start of the run might allow for more effective exploitation of food sources that have already been located. Thus, the agents achieve a higher fitness score because a better path has been achieved by the higher number of agents exploiting the path. On the other hand, higher ratios of scout bees at initialization negatively impacts the fitness score because there are too many agents searching, and not enough exploiting the available food sources. This may also be a result of the size of the search space. For a larger search space, the scout role may present a more positive benefit, particularly if the best food sources are located farther away from the point of origin. If the runs are given more time, the agents may also have more time to find the optimal ratios of each role, since the agents can change between particular roles during the course of the simulation.

For future experiments, to test whether the same results would hold true, we would look at a larger space with more food options, more gravitational obstacles, and planetary movement. A larger search space with more food options would help determine whether the negative impact of the scout bee ratio would hold for a space where more scouts may be necessary to discover the optimized route. More gravitational obstacles and planetary movement would add more complexity to the problem, and could demonstrate the capabilities of the artificial bee colony if it was able to successfully solve for an optimized path. We would also like to implement role change for the bees, so that they can adjust the number of bees in each role based on the current needs. We also believe that the fitness function could be improved by giving increased bonuses based on a straight line proximity, rather than being close to the target planet's orbital path. In the future, it may reveal more about how the agents interact with gravity to add planets or moons in such a way that a gravity assist from a planet would increase the effectiveness of a path. However, this may require the fitness score to also take a temporal aspect into account.

Acknowledgment

Thank you to Dr. Schuman for teaching us the required skills to perform this experiment and for helping guide us through the issues and changes this project underwent.

References

[1] G. Acciarini, D. Izzo, and E. Mooij, "Mhaco: a multi-objective hypervolume-based ant colony optimizer for space trajectory optimization," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, p. 1–8, July 2020.

[2] M. A. Cooper and B. Smeresky, "An overview of evolutionary algorithms toward spacecraft attitude control," *Advances in Spacecraft Attitude Control*, 2020.

[3] A. K. Kar, "Bio inspired computing – a review of algorithms and scope of applications," *Expert Systems with Applications*, vol. 59, pp. 20–32, 2016.

[4] C. R. Bessette and D. B. Spencer, "Optimal space trajectory design: A heuristic-based approach,"

[5] P. Mansouri, B. Asady, and N. Gupta, "Solve shortest paths problem by using artificial bee colony algorithm," *Advances in Intelligent Systems and Computing*, vol. 258, pp. 183–191, 2014. Presented at the Third International Conference on Soft Computing for Problem Solving (SoCProS 2013).

[6] M. Reda, A. Onsy, A. Y. Haikal, and A. Ghanbari, "Path planning algorithms in the autonomous driving system: A comprehensive review," *Robotics and Autonomous Systems*, vol. 174, p. 104630, Apr. 2024.

[7] W.-f. Gao and S.-y. Liu, "A modified artificial bee colony algorithm," *Computers Operations Research*, vol. 39, p. 687–697, Mar. 2012.

[8] E. Forcael, I. Carriel, A. Opazo-Vega, F. Moreno, F. Orozco, R. Romo, and D. Agdas, "Artificial bee colony algorithm to optimize the safety distance of workers in construction projects," *Mathematics*, vol. 12, no. 13, p. 2087, 2024.

[9] T. Vinko, D. Izzo, and C. Bombardelli, "Benchmarking different global optimisation techniques for preliminary space trajectory design,"

[10] A. E. Kossovsky, *Newton's Law of Universal Gravitation*, pp. 171–172. Cham: Springer International Publishing, 2020.

[11] E. W. Weisstein, "Kepler's equation."

[12] C. Van Dine, W. Fimple, and T. Edelbaum, "Application of a finite-difference newton-raphson algorithm to problems of low-thrust trajectory optimization," in *Methods in Astrodynamics and Celestial Mechanics* (R. L. Duncombe and V. G. Szebehely, eds.), vol. 17 of *Progress in Astronautics and Rocketry*, pp. 377–400, Elsevier, 1966.

[13] S. Q. Kidder and T. H. V. Haar, *2 - Orbits and Navigation*, p. 15–46. San Diego: Academic Press, Jan. 1995.