

Optimizing Robot Path Planning through Weighted Shortest Path based Dijkstra's Algorithm: Navigating from Source to Goal in 2D Environment on Land

Shubhangi Saha¹, Sharon B. Rodrigues², Vipul Singh³, Siddhant Singh⁴, Geetansh Soni Verdhan⁵,
Dr. Pavithra G⁶ and Dr. T.C. Manjunath⁷

¹⁻⁵Final year BE (ECE) Students, VII Sem, Electronics & Communication Engineering, Dayananda Sagar College of Engineering, Bangalore, Karnataka

⁶Associate Professor, Dept. of Electronics & Communication Engineering, Dayananda Sagar College of Engineering, Bangalore, Karnataka

⁷Professor & HOD, Dept. of Electronics & Communication Engineering, Dayananda Sagar College of Engineering, Bangalore, Karnataka

Email: tcmanju@iitbombay.org

Abstract— The research introduces an advanced approach to robot path planning by incorporating Dijkstra's algorithm along with the inclusion of path weighting. The integration of path weights is designed to improve navigation efficiency from the starting point to the destination across diverse environments. This optimization aims to refine trajectory decisions and enhance overall performance, particularly in intricate and challenging scenarios.

Index Terms— Robot, AI, Path Planning, Shortest Path, 2D.

I. INTRODUCTION

Robot path planning is a pivotal aspect of robotics, focusing on determining the most efficient route for a robot to travel from its starting point to a designated destination while navigating around obstacles. This task is critical for the successful and safe movement of robots in various environments. To address this challenge, researchers often turn to Dijkstra's algorithm, a widely employed graph search algorithm known for finding the shortest path between two points in a graph. In the realm of robotics, Dijkstra's algorithm plays a crucial role in optimizing the path-finding process.

Path weighting, another key concept in this study, introduces a method of assigning different weights to distinct segments of a path. In the context of robot path planning, this means attributing varying costs to different areas or types of terrain that the robot may encounter during its journey. The integration of path weighting with Dijkstra's algorithm is specifically designed to enhance navigation efficiency. By considering the assigned weights for each segment, the robot can make more informed decisions, ultimately leading to an optimized trajectory.

Furthermore, the proposed method demonstrates versatility across diverse environments. This adaptability is particularly essential for robots operating in varied settings, each presenting unique challenges and obstacles. The optimization of trajectory decisions becomes fundamental for achieving superior overall performance, especially in intricate scenarios. The study asserts that this enhanced approach contributes to improved robot performance,

particularly in complex scenarios characterized by dynamic obstacles, changing conditions, or intricate terrain. In summary, the study introduces a method that combines Dijkstra's algorithm with path weighting to improve the efficiency of robot path planning. This enhancement is expected to result in better performance, especially in complex and varied environments. The incorporation of weights allows for a more nuanced consideration of the terrain or conditions, influencing the robot's trajectory decisions for optimal outcomes [1].

II. LITERATURE SURVEY

A number of researchers have worked on the proposed topic, here follows a brief review of the works carried out by other researchers in the form of 6 papers as follows.

PAPER 1: Route Planning for an Autonomous Robotic Vehicle Employing a Weight- Controlled Particle Swarm-Optimized Dijkstra Algorithm - A reliable approach to the route planning method is weight-controlled particle swarm-optimized Dijkstra algorithm (WCPSODA). The Dijkstra algorithm seeks the route with the minimum distance between two points by going in the opposite direction of the length of the paths. In mapping with a single source, it addresses the minimum path issue by determining the shortest route between the origin and destination nodes. In this study, the conventional Dijkstra algorithm is enhanced to address the route planning issue in the square warehouse framework setting. In the square surroundings area, there are several smallest paths of equal length. The Dijkstra method only keeps one intermediary node, making it possible to search for just one fastest route. By setting aside any nodes that are equally far from the origin node as the intermediary nodes, we enhance the Dijkstra method [2].

PAPER 2: Path Planning for Mobile Robot Navigation in Unknown Indoor Environments Using Hybrid PSOFS Algorithm - This paper discusses A* algorithm which is one of the variants of Dijkstra's algorithm and commonly used to find the shortest path in a weighted directed graph. Unlike Dijkstra's algorithm which only can explore one possible path at a time, the algorithm can explore multiple possible paths at a time from the origin vertex, until one of the paths ends at the destination vertex. The efficiency of the algorithm in pathfinding is higher than Dijkstra's algorithm. The algorithm is mainly used for global search in static environments. This paper is aimed to solve the problem of planning the optimum path for a mobile robot to navigate in unknown indoor environments [3].

PAPER 3: Dynamic Path Planning Algorithms with Load Balancing Based on Data Prediction for Smart Transportation Systems - In this paper, a traffic path planning algorithm based on data prediction named TPPDP is proposed. A prediction model is designed to predict the traffic state of the road segment in the future based on historical traffic data and current traffic information, and based on the predicted result, which is the average travel speed of road segments in the next time slot, TPPDP algorithm is proposed to plan a path with the shortest travel time. It discusses the Dijkstra Algorithm and its improved versions in detail [4].

PAPER 4: Improved Dijkstra Algorithm for Mobile Robot Path Planning and Obstacle Avoidance - Optimal path planning avoiding obstacles is among the most attractive applications of mobile robots (MRs) in both research and education. In this paper, optimal collision-free algorithm is designed and implemented practically based on an improved Dijkstra algorithm. To achieve this research objectives, first, the MR obstacle-free environment is modeled as a diagraph including nodes, edges and weights. Second, Dijkstra algorithm is used offline to generate the shortest path driving the MR from a starting point to a target point. During its movement, the robot should follow the previously obtained path and stop at each node to test if there is an obstacle between the current node and the immediately following node [5].

PAPER 5: A Survey of Path Planning Algorithms for Mobile Robot - Path planning algorithms are used by mobile robots, unmanned aerial vehicles, and autonomous cars in order to identify safe, efficient, collision-free, and least-cost travel paths from an origin to a destination. Choosing an appropriate path planning algorithm helps to ensure safe and effective point-to-point navigation, and the optimal algorithm depends on the robot geometry as well as the computing constraints, including static/holonomic and dynamic/non-holonomically, constrained systems, and requires a comprehensive understanding of contemporary solutions. The goal of this paper is to help novice practitioners gain an awareness of the classes of path planning algorithms used today and to understand their potential use cases—particularly within automated or unmanned systems. To that end, we provide broad, rather than deep, coverage of key and foundational algorithms, with popular algorithms and variants considered in the context of different robotic systems. The definitions, summaries, and comparisons are relevant to novice robotics engineers and embedded system developers seeking a primer of available algorithms [6].

PAPER 6: Dijkstra Algorithm based intelligent path planning with topological map and wireless communication - In this paper, an idea of intelligent path planning was introduced. Using information received from a server that

is transmitted through a wireless communication, the data is processed to edit a provided topological map, with which a shortest path calculation will be executed. Information about the destination and blocked road parts is sent to a mobile robot using a wireless ad hoc network communication. The mobile robot will process this information in a topological 2D array map and ignore the blocked parts of the road for the shortest path calculation based on Dijkstra's algorithm. Aside from the intelligent path planning, an automated driving algorithm was also implemented using infra-red sensors installed on the mobile robot to navigate the robot to its destination [7].

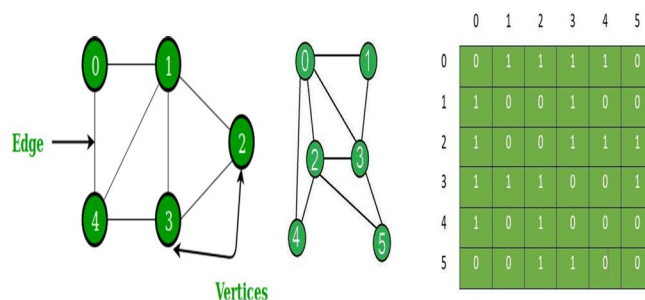


Figure 1. Graph of the network

III. PROPOSED METHODOLOGY

A graph is a nonlinear data structure comprising vertices and edges. Vertices, also known as nodes, are connected by lines or arcs referred to as edges. More formally, a graph is defined by a set of vertices (V) and a set of edges (E), denoted as $G(V, E)$. Graphical data structures serve as potent tools for representing and analyzing intricate relationships between objects or entities. They find practical application in diverse fields such as social network analysis, recommender systems, and computer networks. In sports informatics, graph data structures play a pivotal role in analyzing and comprehending the dynamics of team performance and player interactions on the field [8]. Envisioning a soccer game as a web of connections, where players are the nodes and their interactions form the edges, illustrates how the graph data structure encapsulates and elucidates team dynamics and player performance in sports. The components of a graph include vertices, which are the fundamental units, sometimes referred to as nodes. Each node or vertex can be labeled or unlabeled. Edges, the connections between nodes, can be represented as ordered pairs in a directed graph. There are no strict rules governing how edges can connect nodes, allowing for flexibility in their arrangement. Edges may also be termed arcs, and like vertices, each edge can be labeled or unlabeled. Understanding the distinct roles of vertices and edges is essential for comprehending the structure and representation of data in a graph [9].

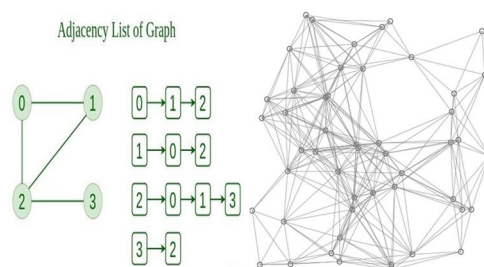


Figure 2. Adjacency list of graph with the shortest path

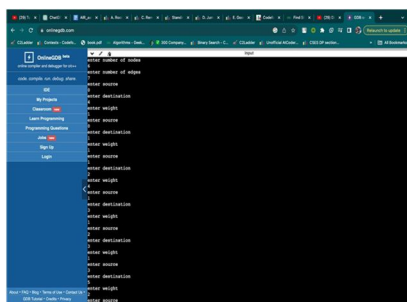


Figure 3. Simulation result

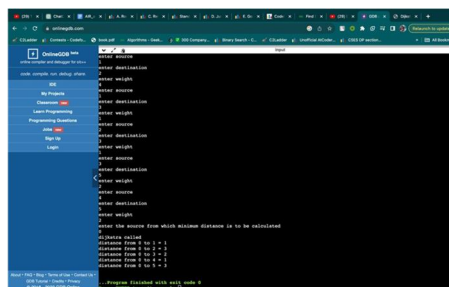


Figure 4. Graph visualisation for the above input

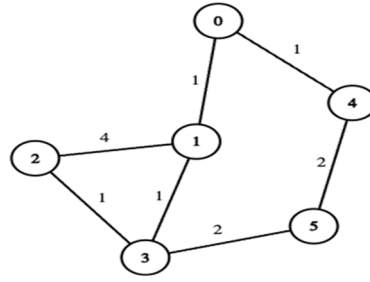


Figure 5. Graph of the network

IV. RDJ ALGO DEVELOPMENT

The RDJ algorithm, a variant of the classic Dijkstra's algorithm for finding the shortest path in graph theory, focuses on optimizing its implementation. Traditional single-source shortest path algorithms for connected weighted graphs with n vertices often have a time complexity of $O(n^2)$. To enhance the efficiency of Dijkstra's algorithm, Barbehen et al. introduced the use of a heap sorting algorithm, reducing the time complexity to $(m + m \log n)$. Despite this improvement, real-time path planning requirements remained challenging. In the RDJ algorithm's node expansion mechanism, which seeks the shortest path from a single source point to each node, two crucial operations are identified: relaxing each node and finding the node with the smallest distance from the source point in each iteration. To mitigate the time-consuming process of extracting the node with the smallest distance, a new node expansion mechanism is proposed. The algorithm leverages a grid-based environmental model and adopts a four-neighborhood search. Instead of selecting a single node with the smallest distance, it employs a breadth-first search approach, considering four nodes in a specific order as the next expansion nodes. The algorithm begins with a binary matrix representing the working environment and the starting point 's.' The Manhattan distance ($L(x)$) from each node 'x' to 's' is calculated based on the breadth-first search. Notably, unlike Dijkstra's algorithm, this approach doesn't consider marking the parent node during expansion. Consequently, upon completing the traversal of all expandable nodes, the algorithm provides a Manhattan distance potential field from each vertex 'x' to the starting point 's,' as depicted in Fig. 1. It's essential to highlight that this algorithm doesn't yield the shortest path from each point to the starting point after traversing all expandable nodes [10].

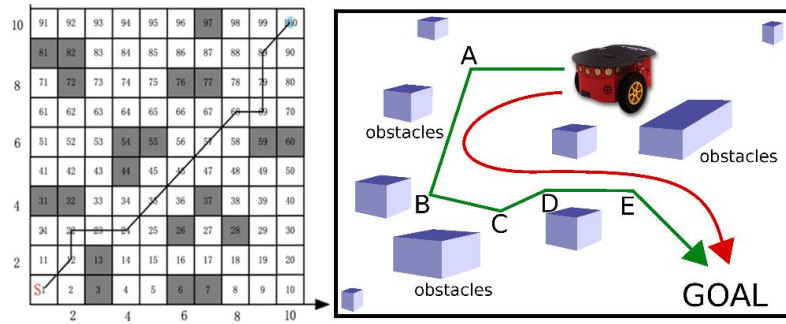


Figure 6. Finding the shortest path using Dijkstra's Algo

V. DESIGN OF THE REVERSE SEARCH ALGORITHM

Designing a reverse search algorithm involves optimizing the path-finding process from a specific point 'e' to a designated point 's,' as illustrated in the provided . Notably, the current path is determined solely by searching horizontally or vertically from point 'e,' and the rate of change along this path direction is denoted as 1. To enhance the algorithm's efficiency, the consideration of diagonal direction search is introduced. In this context, when the change rate along the diagonal direction is 2, indicating greater proximity to the starting point, the actual moving cost for the robot following the diagonal direction is $2-\sqrt{2}$ times lower than that along the horizontal or vertical direction [11].

Leveraging the Manhattan distance potential field, the reverse eight-neighborhood search algorithm is formulated with the following input and output specifications: Input - Manhattan distance potential field from the source point to each point; Output - approximate shortest path from point 'e' to 's.' The algorithm proceeds by initializing 'v' as

'e' and iteratively finding the eight adjacent nodes 'x' of 'v.' It then compares the Manhattan distances ($L(x)$) of these eight nodes, selects the smallest node 'x,' and stores the result. The process continues until 'v' becomes 's,' constructing an approximate shortest path [12].

Essentially employing the greedy algorithm's principle to construct a global optimal solution through the selection of local optimal solutions, the algorithm yields a significantly shortened path from point 'e' to 's' compared to the original path. The length reduction is evident in 5, where the obtained shortest path is approximately $10 + 10\sqrt{2}$, a decrease of $20 - 10\sqrt{2}$ compared to the path in 4. The algorithm's returned path length, given a Manhattan distance 'p' from the source point to the target point, falls within the range of 10 to 100 [13].

VI. CONCLUSIONS

In conclusion, the findings from our case study underscore the efficacy of the proposed Dijkstra algorithm in the realms of robot path planning and obstacle avoidance. The algorithm's ability to dynamically recalculate optimal paths in real time plays a pivotal role in facilitating effective obstacle avoidance for mobile robots. Notably, the success of the algorithm in navigating through dynamically introduced obstacles highlights its adaptability and robustness in handling complex scenarios [14].

Despite a convergence time for the solution algorithm in real-world applications that is proportionate to the square of the number of nodes in a digraph modeling environment, the algorithm demonstrates promising performance. This is particularly noteworthy considering the relatively small number of nodes in the indoor environment studied. The algorithm's practical functionality is further reinforced by the successful implementation on the Arduino board, affirming both computational efficiency and memory capacity. The case study collectively emphasizes the algorithm's potential as a valuable tool in real-world robotic applications, showcasing its ability to address dynamic challenges and navigate through obstacles with adaptability and efficiency [15].

VII. SIMULATION RESULTS & DISCUSSIONS

Analyzing the error in path length from point 'e' to point 's,' it is observed that the shortest path obtained through the DJ algorithm for an eight-neighborhood search aligns with the path length depicted in Fig. 5. However, this alignment does not validate the RDJ algorithm as the shortest path solution. In practice, the RDJ algorithm exhibits a flaw during the eight-neighborhood reverse search, where the change rate becomes equal in two directions, as illustrated in Fig. 5. Notably, the algorithm mistakenly considers the distances from point 'e' to the left and down as equal, leading to a significant error [16].

To mitigate this maximum error, algorithmic improvements are imperative. Currently, when the change rates in two directions are equal, the algorithm accepts either direction with equal probability, offering a partial solution to the issue. Nevertheless, situations like the one depicted in Fig. 5 may still result in unavoidable errors, especially when the target point distance is 'p = 10.' Some equations provide specific error analyses for these scenarios, but they are limited to these special cases and lack general applicability. Experimental simulations in more typical mobile robot working environments, as depicted in Fig. 5, reveal that the errors resulting from these situations remain within acceptable bounds [17].

In terms of algorithm time complexity, the RDJ algorithm differs from the DJ algorithm in two key aspects. Firstly, in the search node expansion mode, the RDJ algorithm abandons relaxation operations and optimizations, opting for expansion based on the order of inserting into the OPEN table, implemented as a circular queue. The time complexity of basic operations, insert and delete, in a circular queue is $O(1)$. Consequently, RDJ algorithm traverses the entire search space in $O(n)$ time. Secondly, while DJ returns a shortest path based on marked parent-child nodes, RDJ employs a greedy algorithm to locate the node with the shortest distance from the starting point in the range potential field, resulting in a time complexity less than $O(n)$. In summary, the time complexity of the RDJ algorithm is determined to be $O(n)$ [18].

To assess the efficacy of the RGJ algorithm, it is subjected to a comparative analysis with the well-established DJ algorithm and A* algorithm, considering both running time and path length. The evaluation employs simulation tools, specifically Visual Studio 2010 and Matlab2010b, with the algorithm running on a Windows 7 system equipped with an Intel Core I3 processor (2.53 GHz) and 2 GB of memory. The simulated map environments encompass random obstacle distributions (referred to as random, with 30% obstacle coverage), mazes, and warehouses, as depicted in Fig. 6. The maps are characterized by grid scales of 200×200 [19].

To optimize program execution speed, the data structure incorporates a binary heap priority queue for the classic DJ algorithm and A* algorithm, implemented through a hybrid programming approach involving both C++ and MATLAB. Throughout the experimental process, each algorithm is executed 100 times on every map variant, with

the starting point consistently positioned at the bottom left and the endpoint at the top right. This systematic evaluation aims to provide a comprehensive understanding of the RGJ algorithm's performance relative to the DJ and A* algorithms across diverse map scenarios [20].

REFERENCES

- [1] Syed Abdullah Fadzli, Sani Iyal Abdulkadir, Mokhairi Makhtar, Azrul Amri Jamal “Robotic Indoor Path Planning using Dijkstra’s Algorithm with Multi-Layer Dictionaries”, Journal Paper, 2015.
- [2] Huijuan Wang, Quanbo Yuan, Yuan Yu, “Application of Dijkstra algorithm in robot path- planning”, Journal Paper, 2011.
- [3] Masato Noto, Hi roa.ki IS a,to ,“ A Method for Shortest Path Search by Extended Dijkstra Algorithm”, Journal Paper, 2002.
- [4] Mr. S. Julius Fusic, Mr.P.Ramkumar, Dr.K.Hariharan, “Path planning of robot using modified dijkstra Algorithm”, Journal Paper, 2018.
- [5] Saif Allah M. Abgenah, Azrul Amri Jamal, Syed Abdullah Fadzli, “Multi-Robot Path Planning using Dijkstra’s Algorithm with Multi-layer Dictionaries”, Journal Paper, 2021.
- [6] Subaselvi Sundarraj, R. Vijaya Kumar Reddy, Mahesh Babu Basam, Gururaj Harinahalli Lokesh, Francesco Flammini, Rajesh Natarajan, “Route Planning for an Autonomous Robotic Vehicle Employing a Weight-Controlled Particle Swarm-Optimized Dijkstra Algorithm”, Journal Paper, 2023.
- [7] Shaher Alshammrei, Sahbi Boubaker and Lioua Kolsi, “Improved Dijkstra Algorithm for Mobile Robot Path Planning and Obstacle Avoidance”, 2022
- [8] Shashank S., Kushal K., Abhay Surya Shankar, Madan Kumar G., Dr. Pavithra G., Dr. Sindhu Sree M., Padmavathy M., Dr. T.C. Manjunath, “RFID based attendance system with SMS Alert”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 774 – 782, Jul. - Sept. 2023.
- [9] Abhishek, Sujith M.S., Jeevan D., P. Kamalesh, Dr. Sindhu Sree M., Dr. Pavithra G., Dr. T.C. Manjunath, “Design & Development of a Table Assisted Robotic Arm”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 818 – 822, Jul. - Sept. 2023.
- [10] Apeksha U., Chithrashree G.S., Divya N.M., Shalmali S. Mankikar, Dr. Sindhu Sree M., Dr. Pavithra G., Dr. T.C. Manjunath, “Wireless LoRa Communication Between Two Arduino Uno for Military Application in Soldier Tracking”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 768 – 773, Jul. - Sept. 2023.
- [11] Akarsh Kesharwani, Ayush P. Chaudhary, Bhanu Pratap Singh, Ved Kumar, Padmavathi M., Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “A Study on Hand Motion Controlled Robotic Arm”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 812 – 817, Jul. - Sept. 2023.
- [12] Leena Jeyakumar, Prerana Aithal, Vismitha R., Pradhan Aithal, Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “Development of Smart Bridge – Automatic Height Increase When Floodings Take Place”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 763 – 767, Jul. - Sept. 2023.
- [13] Sushanthi Raj, Manohar R., Bhuvan G.S., Deepthi. S.R., Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “Design and Development of Obstruction Detection and Removal Bot”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 807 – 811, Jul. - Sept. 2023.
- [14] Anagha, Jhanavi M., Khushi M.S., Nithin Kumar S., Dr. Sindhu Sree M., Dr. Pavithra G., Dr. T.C. Manjunath, “Paralysed Patient Healthcare Monitoring Device”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 758 – 762, Jul. - Sept. 2023.
- [15] Ashmeet Singh, Harshhit Goenka, Prakhar Sahu, Venkatesh L., Pamavathi M., Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “Development of an Automatic Fire Extinguisher”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 802 – 806, Jul. - Sept. 2023.
- [16] Vaishnavi Patil, Dr. Pavithra G., Dr. T.C. Manjunath, “Design, Development of a Diversified Implementation of a Supervisory Control And Data Acquisition based VLSI System (SCADA) framework Utilizing Microcontroller based Programmable Logic Controllers”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 879 – 890, Jul. - Sept. 2023.
- [17] Kavya P., Sanjana S., Harika, Teju R., Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “Design & Development of Drones Using Radio Frequency Controllers”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 707 – 710, Jul. - Sept. 2023.
- [18] S.G. Swathi, Aliya Bhandari, Srushti M. B. Kavya A., Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “Voice Control Robot – Design & Development for Various Domestic Applications”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 791 – 801, Jul. - Sept. 2023.

- [19] Vaishnavi Patil, Dr. Pavithra G., Dr. T.C. Manjunath, “Design of Smart Wheelchair For Disabled (Handicapped) Persons Using Real Time Embedded Systems & Internet Of Things Approach”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 871 – 878, Jul. - Sept. 2023.
- [20] Biswendu Biswas, Madan V.L., Rakesh B.S., Prathik Chandrapal, Dr. Pavithra G., Dr. Sindhu Sree M., Dr. T.C. Manjunath, “Development of remotely operated military purpose aerial vehicles”, Scopus Journal Q3, Schimago Ranking SJR 2022 0.32, H-Index 24, Tuijin Jishu/Journal of Propulsion Technology, ISSN : 1001-4055, Vol. 44, No. 3, pp. 638 – 641, Jul. - Sept. 2023.