TECHNICAL SCIENCE RESEARCH IN UZBEKISTAN
ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

ROUTE CONSTRUCTING FOR A MOBILE ROBOT BASED ON THE D-STAR ALGORITHM

Vladyslav Yevsieiev[1], Amer Abu-Jassar[2], Svitlana Maksymova[1], Ahmad Alkhalaileh[3]

[1]Department of Computer-Integrated Technologies, Automation and Robotics, Kharkiv National University of Radio Electronics, Ukraine

[2]Faculty of Information Technology, Department of Computer Science, Ajloun National University, Ajloun, Jordan

[3]Senior Developer Electronic Health Solution, Amman, Jordan

ABSTRACT:

This paper discusses the use of the D-star algorithm to construct an optimal route for a mobile robot in a space with obstacles. We present a mathematical description of the D-star algorithm operating principle, which is based on the idea of dynamic programming and step-by-step path cost updating. Based on this description, a Python program was developed that is capable of building a route for the robot, taking into account the situation around it. To test the efficiency and accuracy of the algorithm, a number of experiments were carried out on various test maps with different obstacle configurations. The results showed that the D-star algorithm demonstrates high efficiency and reliability in constructing the optimal route for a mobile robot under various conditions.

Key words: Industry 5.0, Mobile Robots, Work zone, Manufacturing Innovation, Industrial Innovation, Route Planning.

INTRODUCTION

Modern industry is on the verge of moving into the era of Industry 5.0, where digital technologies play an increasingly important role in increasing the efficiency and autonomy of production [1]-[9]. In this context, the use of mobile robots is becoming increasingly relevant for work in various production areas [10]-[23], including work areas with obstacles. And here various methods and approaches can be used [24]-[30].

Planning the optimal path for mobile robots in work areas is key to ensuring their safety and efficiency [31]-[35]. One of the effective algorithms for solving this problem is the D-star algorithm.

The D-star algorithm allows you to build an optimal route between the starting and ending points, taking into account obstacles and changes in the environment. This algorithm is widely used in various areas where route planning is required, including industry, where mobile robots are used to perform various tasks.

The purpose of this article is to implement and study the applicability of the D-star algorithm for constructing an optimal route for a mobile robot in a work area, taking into account obstacles. The results of the study can be useful for industrial enterprises seeking to improve the efficiency of their production in the context of Industry 5.0.

RELATED WORKS

The problem of path planning for a mobile robot is extremely relevant. Many scientists are developing and modifying various algorithms to solve this problem. Among such algorithms, the D-Star algorithm should be highlighted. Let's consider some recent works devoted to this application of this algorithm.

The main task of mobile robots is to follow the route quickly and without damage. The D-star algorithm is often used for obstacles avoidance, but despite its effectiveness, it has some drawbacks [36]. The article [36] presents the optimization of the path for that modification which allows to avoid wasting time on stopping and turning in the right direction.

The paper [37] focuses on the modelling a non holonomic mobile platform and to develop a hybrid algorithm for avoiding dynamic obstacles. star lite and a geometric-based hybrid algorithms are combined to generate the optimized path for avoiding the dynamic obstacles. The weighting function introduced along with the D star lite algorithm decreases computational time by decreasing the number of expanding nodes during path generation.

The researchers [38] propose to use the improved A-Star algorithm, the sparse A-Star search (SAS), and the dynamic A-Star algorithm(D-Star). They respectively adopted to address the problem of penetration route planning for stealth UAV in three different threat scenarios.

Authors in [39] note, that using the A* algorithm is one of the pathfinding methods used in video games to find the shortest path on the track to avoid static or dynamic obstacles, and The D* Lite algorithm is highly successful and capable of eliminating several difficulties and producing more ideal outcomes.

The study [40] proposes a Multi-level Prediction D-star algorithm (MLP D-star) based on threat cost to address the path planning problem of mobile robots in local unknown environments. The algorithm improves the node expansion of the D-

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

star algorithm using a multi-level prediction structure, which avoids excessive turning points in the planned path.

Scientists [41] propose a novel Demonstrative Self-TrAining fRamework (D-STAR) that leverages multi-perspective thought navigation. D-STAR iteratively optimizes a question generator and an entity retriever by navigating thoughts on a dynamic graph reasoning across multiple perspectives for question generation.

Chai, Y., and co-authors [42] propose an autonomous ship obstacle avoidance method that combines the D-star algorithm and the twin-delayed deep deterministic policy gradient algorithm in order to solve the problem of autonomous ship collision avoidance.

Thus, we can see how diverse the areas and ways of applying the D-Star algorithm are. Further in this article we will present our vision of how this algorithm can be used to determine the trajectory of a mobile robot.

Mathematical description of the D-Star algorithm for planning the route for a mobile robot

The D* algorithm (D-star) is an algorithm for finding the shortest path in a graph, which is a modification of the A* algorithm and is designed to work with dynamic environments. Unlike A*, D* is able to adapt to changes in the environment during pathfinding. It uses two main data structures: a graph to represent the mesh, and a priority queue to control the order in which vertices are traversed.

The D* (D-star) algorithm is a powerful tool for constructing optimal routes on a terrain map for moving mobile robots. Here are some key features of its use:

– the D* algorithm allows the mobile robot to adapt to changes in the environment in real time. This is especially useful if obstacles arise or your travel plan changes;

– the D* algorithm is highly efficient due to incremental updating of path information and the ability to recalculate the path without revising the entire map;

– the D* algorithm guarantees finding the optimal path between the starting and ending points, which is important for saving time and resources;

– real-time operation: Due to its efficiency and ability to handle changes in the environment, D* can be used for real-time path planning without noticeable delays;

– ease of implementation - despite its power, the D* algorithm is relatively simple to implement and can be adapted to different types of robots and environments;

– resistance to noise, the D* algorithm is able to effectively manage noise in the data or incomplete knowledge about the environment due to its adaptability;

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

– the D* algorithm allows you to work with grids of arbitrary shape and complex topology, which makes it a universal tool for various path planning problems;

– the D* algorithm can take into account various risk factors when planning a route, such as speed, probability of obstacles and others;

– the D* algorithm is widely used in real conditions, such as autonomous navigation of robots in industrial complexes, shops or warehouses;

– the D* algorithm can be extended and modified to take into account additional factors or improve performance in specific use cases.

As a result, the main steps of the D* algorithm can be represented as follows. Let the set of vertices U contains all the vertices of the graph, then we denote by $g(u)$ - the cost of the path from the initial vertex s to the vertex u. Let us denote that initially $g(s)=0$, for all other vertices $g(s)=\inf$. Let us introduce the notation $rhs(u)$ - the "correct" (current) cost of the path from the vertex u to the final point of the "goal" $goal$.

To update the correct cost rhs for all vertices $u \in U$, we can imagine it like this:

$$rhs(u)=\min(g(v))+c(u,v), \tag{1}$$

v – neighbour u;

$c(u,v)$ – cost of transition from u to v.

The next step is to calculate the key to the vertex u

$$k(u)=[\min(g(u)),rhs(u)+h(start,u)+km], \tag{2}$$

km – some heuristic constant;

$h(start,u)$ – heuristic function for estimating cost from $start$ to u.

Then while $U \neq 0$ we perform the following set of actions:

– select the vertex u with the smallest key $k(u)$;

– if $g(u)>rhs(u)$, then $g(u)$ becomes equal to $rhs(u)$, otherwise we update $g(u)$ and all the neighbours u. After this we update rhs for all the neighbours u, if $g(start)=rhs(start)$, the algorithm terminates.

After completing the algorithm, the path can be restored, moving from the goal to the start at the lowest cost; when the cost of an edge changes, recalculation is carried out for rhs and keys of the vertices through which the changed edge passes.

Software implementation and experiments

To check the correctness of the reasoning, we will develop a program in Python in the PyCharm 2022.2.3 (Professional Edition) development environment.

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

Let us give an example of software implementation of the above described mathematical expressions.

```
# Function for finding a path using the D* method
def d_star(start, goal, obstacles):
    grid = {}
    for i in range(grid_size):
        for j in range(grid_size):
            grid[(i, j)] = float('inf')
    grid[start] = 0
    open_set = {start}
    came_from = {}
```

This function represents the D* algorithm for finding a path on a grid with obstacles. It initializes the grid with infinite values for each cell and sets the value of the initial cell start to 0.

open_set is the set of cells to consider. At the beginning of the algorithm it contains only the starting cell.

came_from is a dictionary that will contain information about which cell the current cell came from.

Next, the algorithm will gradually update the values in the grid, iteratively considering the cells from open_set to find the optimal path to the goal goal.

```
# Path recovery
    path = []
    current = goal
    while current != start:
        path.append(current)
        if current not in came_from:  # Checking if a key is in the dictionary
            break
        current = came_from[current]
    path.append(start)
    path.reverse()
    return path
```

This piece of code is responsible for reconstructing the optimal path from the target cell `goal` to the starting cell `start` using the dictionary `came_from`, which contains information about previous cells on the path:

– an empty list `path` is created to which the path cells will be added;

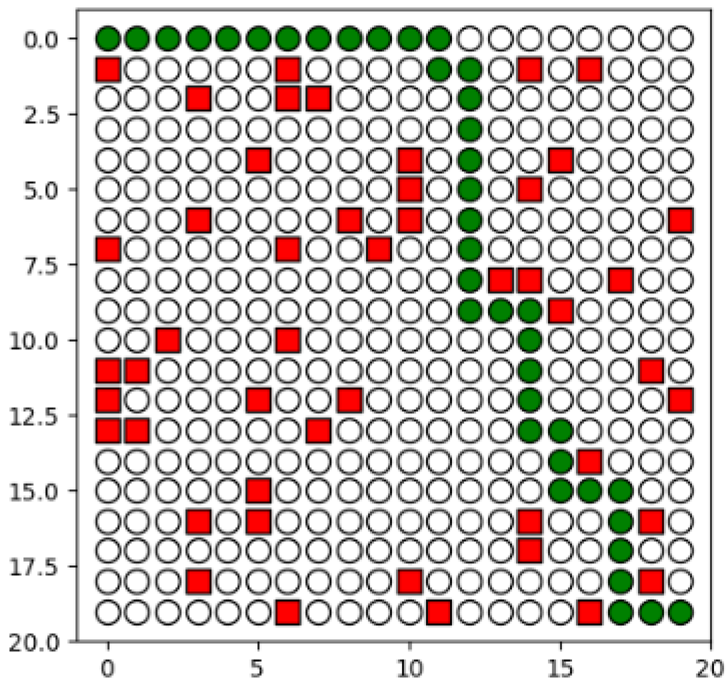
– starts with the `goal` cell and is added to `path`;

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

– further, until the current cell reaches the initial cell `start`, the cycle adds the previous cell to `path`, taken from the dictionary `came_from`. If there is no information about the previous cell for the current cell (for example, if the algorithm could not find a path), the loop is broken using the `break` operator;

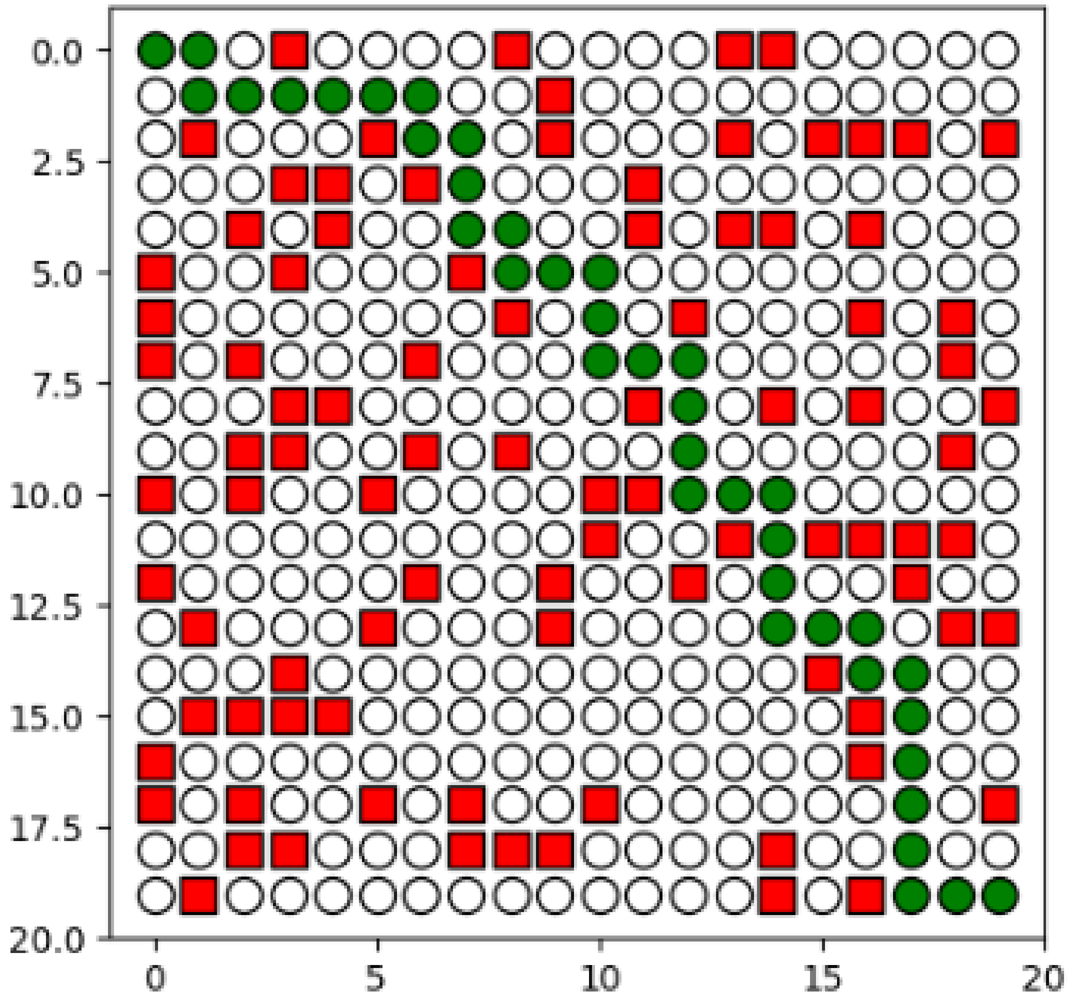
– when all path cells have been added to `path`, starting with target and ending with start, the `path` list is expanded so that the path is ordered from beginning to end, and is returned as the result of the function.

The following hardware was used for research: CPU Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz, RAM 16 Gb, GPU NVideo GeForce GTX 1660Ti (Ram 8Gb), Web-camera HD WebCam, OS Windows 10 Pro (Version 22H2). The program for implementing route construction based on the D* algorithm was developed in the PyCharm 2022.2.3 (Professional Edition) environment in Python. The results of the program are presented in Figure 1.



a) (grid_size = 20); (num_obstacles = 50)

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4



b) (grid_size = 20); (num_obstacles = 100)

Figure 1: Results of the route construction program based on the D* algorithm

To successfully use the D* algorithm, it is necessary to carefully tune its parameters, such as the cost of moving between nodes and the accuracy of the heuristic function. This will help achieve a balance between the accuracy of finding the optimal path and the cost of computing resources. Additionally, when working with dynamic environments, it is important to regularly update information about obstacles and changes in the map so that the algorithm can adapt to new conditions and find optimal paths.

CONCLUSION

The D* (D-star) algorithm is an effective method for finding the optimal path for mobile robots in a changing environment. Its applications are widely used in robotics and autonomous systems to avoid obstacles and achieve goals. The main advantage of D* is that it can recalculate the path as the environment changes, making it an ideal choice for dynamic environments. However, as with any

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

algorithm, D* has its own characteristics and limitations. First of all, D* is highly scalable, making it suitable for pathfinding in large, complex environments. However, as the map size or number of obstacles increases, the algorithm may consume more computing resources, which may affect system performance. It should also be noted that D* does not always guarantee to find the absolute optimal path in all cases due to its heuristic-based nature.

It is important to understand that the D* algorithm is not a universal solution for all scenarios. In some cases, other algorithms such as A* or RRT (Rapidly-exploring Random Tree) may be more efficient. Therefore, before choosing an algorithm for a specific task, it is necessary to analyze the requirements and characteristics of the environment in order to select the most suitable method.

REFERENCES:

1. Attar, H., Abu-Jassar, A. T., Amer, A., Lyashenko, V., Yevsieiev, V., & Khosravi, M. R. (2022). Control System Development and Implementation of a CNC Laser Engraver for Environmental Use with Remote Imaging. Computational intelligence and neuroscience, 2022, 9140156.
2. Nevliudov, I., Yevsieiev, V., Baker, J. H., Ahmad, M. A., & Lyashenko, V. (2020). Development of a cyber design modeling declarative Language for cyber physical production systems. J. Math. Comput. Sci., 11(1), 520-542.
3. Abu-Jassar, A. T., Al-Sharo, Y. M., Lyashenko, V., & Sotnik, S. (2021). Some Features of Classifiers Implementation for Object Recognition in Specialized Computer systems. TEM Journal, 10(4), 1645.
4. Matarneh, R., Maksymova, S., Deineko, Z., & Lyashenko, V. (2017). Building robot voice control training methodology using artificial neural net. International Journal of Civil Engineering and Technology, 8(10), 523-532.
5. Matarneh, R., Maksymova, S., Lyashenko, V. V., & Belova, N. V. (2017). Speech Recognition Systems: A Comparative Review. IOSR Journal of Computer Engineering (IOSR-JCE), 19(5), 71-79.
6. Nevliudov, I., & et al.. (2020). Development of a cyber design modeling declarative Language for cyber physical production systems. J. Math. Comput. Sci., 11(1), 520-542.
7. Lyashenko, V. V., Matarneh, R., Baranova, V., & Deineko, Z. V. (2016). Hurst Exponent as a part of wavelet decomposition coefficients to measure long-term memory time series based on multiresolution analysis. American Journal of Systems and Software, 4(2), 51-56.
8. Abu-Jassar, A. T., Attar, H., Yevsieiev, V., Amer, A., Demska, N., Luhach, A. K., & Lyashenko, V. (2022). Electronic User Authentication Key for

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4
Access to HMI/SCADA via Unsecured Internet Networks. Computational intelligence and neuroscience, 2022, 5866922.

9. Al-Sharo, Y. M., Abu-Jassar, A. T., Sotnik, S., & Lyashenko, V. (2021). Neural Networks As A Tool For Pattern Recognition of Fasteners. International Journal of Engineering Trends and Technology, 69(10), 151-160.

10. Stetsenko, K., & et al. (2023). Exploring BEAM Robotics for Adaptive and Energy-Efficient Solutions. Multidisciplinary Journal of Science and Technology, 3(4), 193-199.

11. Al-Sharo Y., & et al. (2023). A Robo-hand prototype design gripping device within the framework of sustainable development. Indian Journal of Engineering, 2023, 20, e37ije1673.

12. Borysov, H., & et al. (2023). Parameters for Mobile Robot Kinematic Model Development Determination. Multidisciplinary Journal of Science and Technology, 3(4), 85-91.

13. Maksymova, S., & et al. (2024). The Bipedal Robot a Kinematic Diagram Development. Journal of Universal Science Research, 2(1), 6-17.

14. Nevliudov, I., & et al. (2023). A Small-Sized Robot Prototype Development Using 3D Printing. In XXXI International Conference CAD In Machinery Design Implementation and Educational Issues, 12.

15. Attar, H., Abu-Jassar, A. T., Yevsieiev, V., Lyashenko, V., Nevliudov, I., & Luhach, A. K. (2022). Zoomorphic Mobile Robot Development for Vertical Movement Based on the Geometrical Family Caterpillar. Computational intelligence and neuroscience, 2022, 3046116.

16. Baker, J. H., Laariedh, F., Ahmad, M. A., Lyashenko, V., Sotnik, S., & Mustafa, S. K. (2021). Some interesting features of semantic model in Robotic Science. SSRG International Journal of Engineering Trends and Technology, 69(7), 38-44.

17. Sotnik, S., Mustafa, S. K., Ahmad, M. A., Lyashenko, V., & Zeleniy, O. (2020). Some features of route planning as the basis in a mobile robot. International Journal of Emerging Trends in Engineering Research, 8(5), 2074-2079.

18. Maksymova, S., Matarneh, R., Lyashenko, V. V., & Belova, N. V. (2017). Voice Control for an Industrial Robot as a Combination of Various Robotic Assembly Process Models. Journal of Computer and Communications, 5, 1-15.

19. Sotnik, S., & Lyashenko, V. (2022). Prospects for Introduction of Robotics in Service. Prospects, 6(5), 4-9.

20. Ahmad, M. A., Sinelnikova, T., Lyashenko, V., & Mustafa, S. K. (2020). Features of the construction and control of the navigation system of a mobile

robot. International Journal of Emerging Trends in Engineering Research, 8(4), 1445-1449.

21. Al-Sharo, Y. M., Abu-Jassar, A. T., Sotnik, S., & Lyashenko, V. (2023). Generalized Procedure for Determining the Collision-Free Trajectory for a Robotic Arm. Tikrit Journal of Engineering Sciences, 30(2), 142-151.

22. Lyashenko, V., & Sotnik, S. (2022). Overview of Innovative Walking Robots. International Journal of Academic Engineering Research (IJAER), 6(4), 3-7.

23. Abu-Jassar, A. T., Attar, H., Lyashenko, V., Amer, A., Sotnik, S., & Solyman, A. (2023). Access control to robotic systems based on biometric: the generalized model and its practical implementation. International Journal of Intelligent Engineering and Systems, 16(5), 313-328.

24. Lyubchenko, V., Matarneh, R., Kobylin, O., & Lyashenko, V. (2016). Digital image processing techniques for detection and diagnosis of fish diseases. International Journal of Advanced Research in Computer Science and Software Engineering, 6(7), 79-83.

25. Vasiurenko, O., Lyashenko, V., Baranova, V., & Deineko, Z. (2020). Spatial-Temporal Analysis the Dynamics of Changes on the Foreign Exchange Market: an Empirical Estimates from Ukraine. Journal of Asian Multicultural Research for Economy and Management Study, 1(2), 1-6.

26. Nevliudov, I., Yevsieiev, V., Lyashenko, V., & Ahmad, M. A. (2021). GUI Elements and Windows Form Formalization Parameters and Events Method to Automate the Process of Additive Cyber-Design CPPS Development. Advances in Dynamical Systems and Applications, 16(2), 441-455.

27. Mustafa, S. K., Yevsieiev, V., Nevliudov, I., & Lyashenko, V. (2022). HMI Development Automation with GUI Elements for Object-Oriented Programming Languages Implementation. SSRG International Journal of Engineering Trends and Technology, 70(1), 139-145.

28. Lyashenko, V. V., Matarneh, R., Kobylin, O., & Putyatin, Y. P. (2016). Contour detection and allocation for cytological images using Wavelet analysis methodology. International Journal, 4(1), 85-94.

29. Ahmad, M. A., Kuzemin, O., Lyashenko, V., & Ahmad, N. A. (2015). Microsituations as part of the formalization of avalanche climate to avalanche-riskiness and avalanche-safety classes in the emergency situations separation. International Journal, 3(4), 684-691.

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4
30. Babker, A., & Lyashenko, V. (2018). Identification of megaloblastic anemia cells through the use of image processing techniques. Int J Clin Biomed Res, 4, 1-5.

31. Basiuk, V., Maksymova, S., Chala, O., & Miliutina, O. (2023). Mobile Robot Position Determining Using Odometry Method. Multidisciplinary Journal of Science and Technology, 3(3), 227-234.

32. Yevsieiev, V., & et al. (2024). Object Recognition and Tracking Method in the Mobile Robot's Workspace in Real Time. Technical Science Research In Uzbekistan, 2(2), 115-124.

33. Nevliudov, I., & et al. (2023). Mobile Robot Navigation System Based on Ultrasonic Sensors. In 2023 IEEE XXVIII International Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory (DIPED), IEEE, 1, 247-251.

34. Yevsieiev, V., & et al. (2024). Active Contours Method Implementation for Objects Selection in the Mobile Robot's Workspace. Journal of Universal Science Research, 2(2), 135-145.

35. Yevsieiev, V., & et al. (2024). Using Contouring Algorithms to Select Objects in the Robots' Workspace. Technical Science Research In Uzbekistan, 2(2), 32-42.

36. Kadry, S., & et al. (2022). Path optimization for D-star algorithm modification. In AIP Conference Proceedings, AIP Publishing, 2425(1).

37. Sulaiman, S., & Sudheer, A. P. (2022). Modeling of a wheeled humanoid robot and hybrid algorithm-based path planning of wheel base for the dynamic obstacles avoidance. Industrial Robot: the international journal of robotics research and application, 49(6), 1058-1076.

38. Zhang, Z., & et al. (2020). A novel real-time penetration path planning algorithm for stealth UAV in 3D complex dynamic environment. Ieee Access, 8, 122757-122771.

39. Sarbini, R. N., & et al. (2024). Development Of Pathfinding Using A-Star And D-Star Lite Algorithms In Video Game. Journal of Theoretical and Applied Information Technology, 102(3).

40. Suo, H., & et al. (2023). Threat Cost Based Multi-level Prediction D-star Algorithm. International Journal of Social Sciences and Economic Management 4(2), 96-107.

41. Peng, B., & et al. (2024). Multi-perspective thought navigation for source-free entity linking. Pattern Recognition Letters, 178, 84-90.

ISSN (E): 2992-9148 SJIF 2024 = 5.333
ResearchBib Impact Factor: 9.576 / 2024
VOLUME-2, ISSUE-4

42. Chai, Y., & et al. (2023). Intelligent ship navigation method based on deep reinforcement learning algorithm. In 2023 6th International Conference on Intelligent Autonomous Systems (ICoIAS), IEEE, 135-140.