IMPLEMENTATION OF BRUTE-FORCE ALGORITHM AND BACKTRACKING ALGORITHM FOR FIREFIGHTING ROBOT SIMULATION

Tegar Arifin Prasetyo[1], Rudy Chandra[2], Wesly Mailander Siagian[3], Tahan HJ Sihombing[4], Sarbaini[5]
[1,2]Teknologi Informasi, Fakultas Vokasi, Institut Teknologi Del
[3]Manajemen Rekayasa, Fakultas Teknologi Industri, Institut Teknologi Del
[4]Informatika, Fakultas Teknik Informatika dan Elektro, Institut Teknologi Del
[5]Jurusan Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sultan Syarif Kasim

email: [1]tegar.prasetyo@del.ac.id, [2]rudy.chandra@del.ac.id, [3]wesly.siagian@del.ac.id, [4]tahan.sihombing@del.ac.id, [5]sarbaini@uin-suska.ac.id

Abstract

In general, a robot is defined as a mechanical device used by humans to ease human work. Robots are usually used for difficult and dangerous tasks. One example of its use is a firefighting robot that replaces human tasks in extinguishing fires. The firefighting robot is on duty to find fire spots in a city then extinguishing them. To be able to put out a fire, the robot must implement an efficient program in finding and determining the shortest path to the location of the fire and then put it out. For this reason, the robot is equipped with proximity and fire sensors to detect the presence of fire. The design is made with a three-step program that is designing needs of robot control, robot control mechanism scheme preparation and implementing an algorithm for making program syntax. The Brute-Force Algorithm can be implemented to indicate the presence of a hotspot signal and the backtracking Algorithm is implemented to find the shortest path to the hotspot location. This paper discusses the use of a brute-force algorithm and a backtracking algorithm in a firefighting robot program to make the fire search process more efficient. The results show that from 8 input fire points, the firefighting robot is able to find all the points within 3.12 seconds with 13 times trial. In its application, the writer used Visual Basic 6.0 in the firefighting robot program.

Keywords: *Firefighting Robot, Brute-Force Algorithm, and Backtracking Algorithm.*

Abstrak

Secara umum robot didefinisikan sebagai suatu alat mekanik yang digunakan oleh manusia untuk mempermudah pekerjaan manusia. Robot biasanya digunakan untuk tugas-tugas yang sulit dan berbahaya. Salah satu contoh penggunaannya adalah robot pemadam kebakaran yang menggantikan tugas manusia dalam memadamkan api. Robot pemadam kebakaran bertugas untuk menemukan titik api di suatu kota kemudian memadamkannya. Untuk dapat memadamkan api, robot harus menerapkan program yang efisien dalam mencari dan menentukan jalur terpendek menuju lokasi kebakaran kemudian memadamkannya. Untuk itu, robot dilengkapi dengan proximity dan fire sensor untuk mendeteksi adanya api. Perancangan dibuat dengan tiga langkah program yaitu perancangan kebutuhan pengendalian robot, penyusunan skema mekanisme kendali robot *dan implementasi algoritma untuk pembuatan sintaks program. Algoritma Brute-Force dapat diimplementasikan untuk menunjukkan adanya sinyal hotspot dan Algoritma backtracking diimplementasikan untuk mencari jalur terpendek ke lokasi hotspot. Penelitian ini membahas tentang penggunaan algoritma brute force dan algoritma backtracking pada simulasi program robot pemadam kebakaran agar proses pencarian kebakaran menjadi lebih efisien. Hasil penelitian menunjukkan bahwa dari 8 input titik api, robot pemadam kebakaran mampu menemukan semua titik dalam waktu 3,12 detik dengan 13 percobaan. Dalam penerapannya penulis menggunakan Visual Basic 6.0 pada program robot pemadam kebakaran.*

Kata kunci*: Robot Pemadam Kebakaran, Algoritma Brute-Force, dan Backtracking.*

1. INTRODUCTION

These days, technological development is very fast, especially in robotic technology [1,2]. Robots as one of the latest technological products are often used by humans to support their daily activities or in industrial activities such as the production processes in a factory [3]. Robots have many things in common with internal automation, they use similar power devices (such as electric, hydraulic, or pneumatic) and they are controlled by programmable sequences, which allow the machine to the desired position. This environment is defined as an environment, which is under development of this automated machine, which will have various specifications depending on the needs of human activities for industrial automation and robotics [4]. Robots are typically used for heavy, dangerous, repetitive, and dirty work. Other robotic uses include cleaning up toxic waste, exploring underwater and outer space, mining, search and rescue work, and mine searches. Recently, robots have begun to enter the consumer market in the entertainment sector, and household appliances, such as vacuum cleaners and lawnmowers.

One of the tough and high-risk human jobs is firefighting work. It requires a fast reaction because fireman must extinguish the fire before it spreads, if the fire has spread then firefighting work will be difficult. Fire problems can be reduced if the fire source is quickly identified. Firemen certainly want the most efficient path to get to the fire area for saving time and money so that they can carry out their duties properly and more quickly. As a result of dangerous firefighting work risk, eventually, humans design many robots to do firefighting work that helps them [5,6]. Most robotics research uses a finite state machine algorithm that describes robot navigation [7,8,9]. There are other studies related to the use of the depth first search algorithm to reduce time of exploration in multi-robots [10,11,12].

Several previous studies have proposed related algorithms to determine the shortest path. Chen et al. [13] proposed an optimization algorithm for sortest paths in autonomous vehicle navigation systems. Other research shows that the genetic algorithm gives optimal results for the shortest path problem, but for the value of n cities, the processing time of the algorithm will be longer [14]. Shen et al. [15] show that Backtracking Algorithm is able to handle the shortest path problem very well.

In this research a firefighting robot program will be created by implementing the brute-force and backtracking algorithms to detect fire points and search for hotspots that determine the shortest path to the fire point location for the more efficient time used so that the firefighting robot can carry out its duties appropriately. In a simple application, the author describes the implementation of the brute-force and the backtracking algorithms using Visual Basic 6.0 in the firefighting robot program simulation.

2. RESEARCH METHODOLOGY

2.1. Research Design

At the designing stage of a fire extinguishing robot program making, we must understand how the robot works. The firefighter robot works if it detects a fire spot which indicates a fire in a city. Afterward, it looks for the fire point signal and determines the shortest path to the fire point location so that the time required for the fire fighting robot to extinguish the fire will be efficient. For this reason, the implementation of the brute-force algorithm and the backtracking algorithm is needed for the process of making a fire fighting robot program using the Visual Basic 6.0 application. For software, there are several processes that will be carried out: (a) Simulation process is carried out to test the mechanism of the firefighting robot program using the Visual Basic 6.0 application, (b) The process of detecting a fire signal in which the fire occurred, (c) In hotspot search, the robot will detect and determine which path is faster to reach the hotspot signal location, (d) Robot starts the path and goes to where the fire signal is located and (e) Robot managed to find the fire signal and extinguish the fire.

After designing the software, the author applies it to the making of a firefighting robot program by implementing a backtracking algorithm and a brute-force algorithm using the Visual Basic 6.0 application. Then the test is carried out to make sure the program is successful or failed.

2.1. Algorithm

Algorithm is a collection of commands or a way to solve a problem that is used to obtain or explain a certain situation from the beginning to the end so that it can be easier to understand. For the fire fighting robot program, the authors use a brute-force algorithm and a backtracking algorithm.

In its making, the authors use a brute force algorithm to detect hotspot signals. In this place, the fire signal found is illustrated by x, while for each fire signal from several existing fire signals, is illustrated by $x_1, x_2, x_3, \dots \dots \dots, x_n$. The detection is complete when x is found. The following is the design of the brute-force algorithm for the fire fighting robot program:

Hotspot signal searching procedure (input $\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \dots \dots \dots, \mathrm{x}_\mathrm{n}$: integer, x : integer, output idx : integer).
{find x from any existing $\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \dots \dots \dots, \mathrm{x}_\mathrm{n}$. The location in which x is found is filled into idx. If x is not found, then idx is filled with 0.

```
 Input: $x_1, x_2, x_3, \dots\dots\dots, x_n$
Output: idx}
 Declaration:
 k: integer
```

To describe the backtracking algorithm in detail, we divide the path into a series of steps. A move consists of one unit cell movement in a specific direction. There are only four possible directions, they are up, down, to the left, and the right. So, the outline of the algorithm is:

```
while hasn't arrived at the do's destination
    if there is a correct direction such that we have never moved to a cell in that direction
        then
            move one step in that direction
        else
            backtrack the step until there is a direction as mentioned above
    end if
end while
```

In its making, the authors use a recursive solution (which implicitly stores all steps). Recursive solutions are more appropriate for backtracking algorithm because they are the easiest solution and it often use recursive schemes [16]. The advantage of the backtracking algorithm is that the robot does not need to take a long time because the robot can determine the path with a recursive solution[17]. Below is the design of a backtracking algorithm with a recursive solution in the fire fighting robot program:

```
if the solution has been found then
 return true
    else
    for in each movement directions (up, down, left, right) do
    move (robot, direction) {move one step (one cell) according to that
    direction and the direction that has been passed will be saved
    automatically}
 if Solve robot then
 return true
     else
     unmove (robot, direction) {backtrack }
end if
    end for
     return false {all directions have been tried, but still stuck, so the conclusion:
no solution}
end if
```

The algorithm above describes how the robot searches for hotspots and the path which is automatically saved because it uses a recursive solution. Algorithm design for fire fighting robots to find and extinguish the fire:

```
The Procedure to Find and Extinguish the Fire (Robot input: path to be followed, input / output found: boolean)
{initial found value is false because fire has not been found}
Direction Declaration {up, down, left, right}
Algorithm:
examine the path
if find fire point then
        found ← true
        extinguish the fire
        else
        while enter each direction and not found do
         move forward (Robot, direction) {move one step in that direction}
         Search for Fire (Robot, found)
         end while
end if
```

3. RESULT AND DISCUSSION

A firefighting robot is a robot that will be tasked to find hotspots after a fire point detection is found and determine the shortest path to a fire point in a city and extinguish it [18]. To detect a fire point signal, the authors implement the brute-force algorithm in the firefighting robot program. The author implements it by providing several hotspot signals which are illustrated to $x_1, x_2, x_3, \dots\dots\dots, x_n$, and then from some of these fire signals will be searched for a hotspot illustrated to *x* which indicates a fire in a city. The brute-force algorithm is implemented in this which solution is by comparing several hotspot signals and then looking for an active fire signal and indicating the occurrence of a fire. If the hotspot signal illustrated to *x* is found, then the output is the fire signal index and if the hotspot signal is not found then the output is 0. Afterward, the robot determines the shortest path to the hotspot location by implementing a backtracking algorithm that is dividing the path into a series of steps. A move consists of a robot movement in a specific direction. The possible directions are: above (up), below (down), to the left, to the right. The firefighting robot solution in finding and determining the shortest path to the fire point location is a recursive solution. That is best applied in backtracking algorithm because it stores the path data that has been traversed so that it can determine the next path. In the manufacturing process, the author uses the Visual Basic 6.0 application to create a firefighting robot program that implements the brute-force algorithm and the backtracking algorithm.

The firefighting robot program uses the brute-force and backtracking algorithms with the Visual Basic 6.0 application as follows:

When we are opening the firefighting robot program in Visual Basic 6.0, the computer screen will appear as shown in Figure 1:

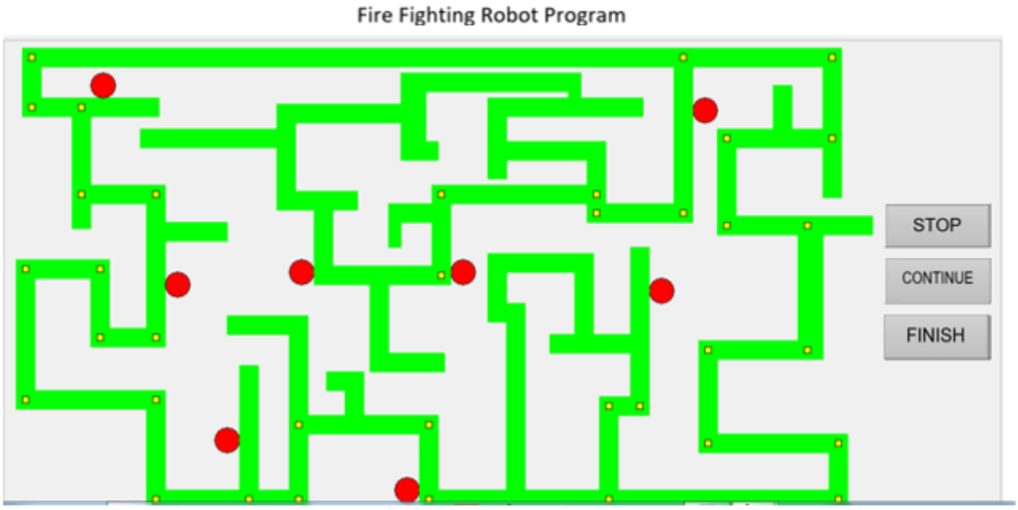


Figure 1. Fire fighting robot program display when run

In Figure 1 display, the red shape has likened to a signal that the fire is located and the yellow shape is likened to a firefighting robot that is ready to extinguish a fire if there is a fire in a city. When clicking the command button STOP will be shown in Figure 2:

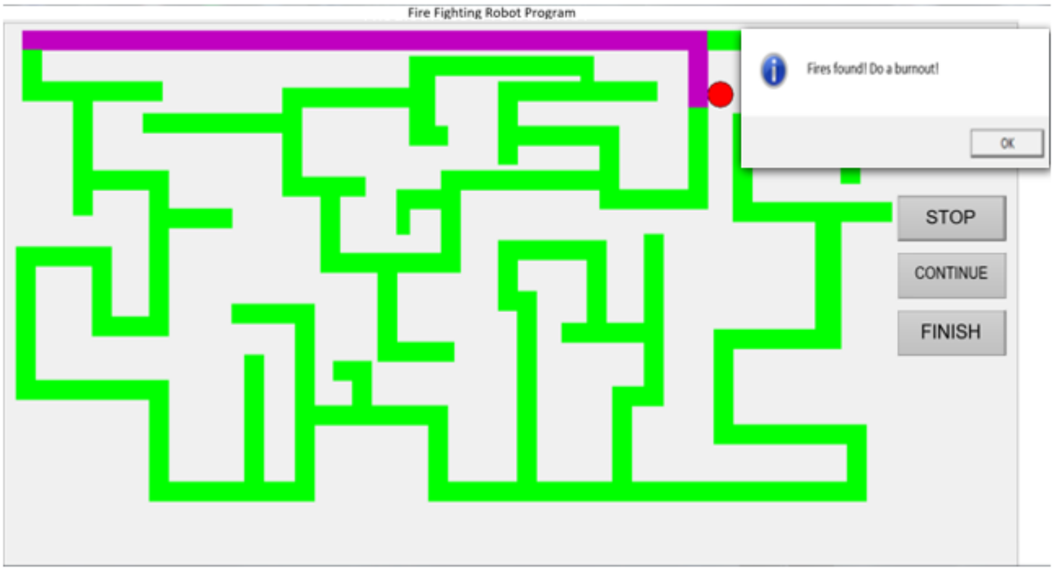

Fires found! Do a burnout!

Figure 2. Fire fighting robot program display when stop

After the fire fighting robot program is started, and when clicking the command button STOP, the red shape will stop and show the location of the fire point. Then the firefighting robot will determine the shortest path to reach that location. The shortest path is indicated by the purple line. When you click *OK*, the yellow shape, representing the firefighting robot, will go to the fire point location and will extinguish it. The appearance is as shown in Figure 3:

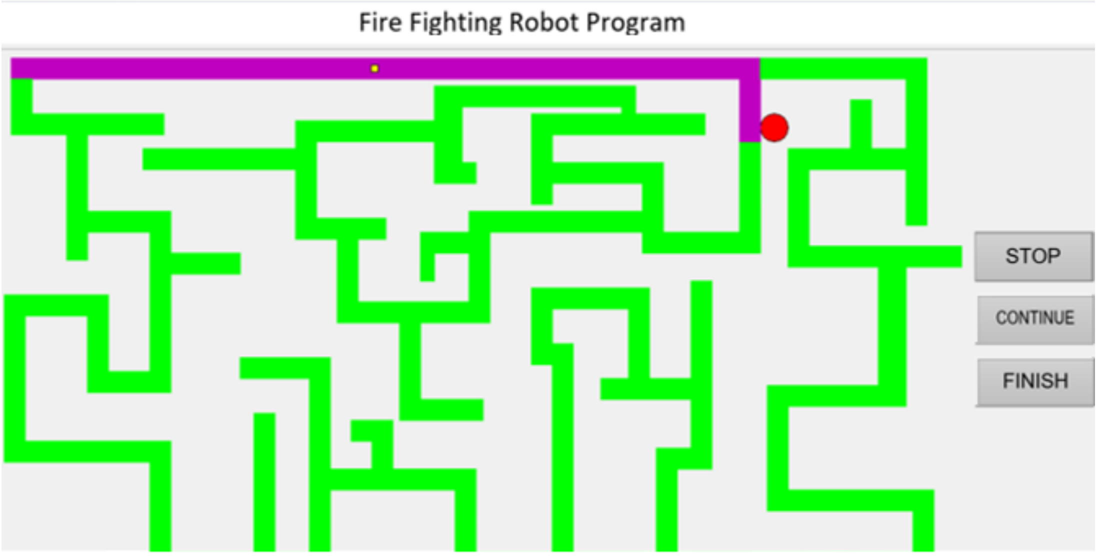
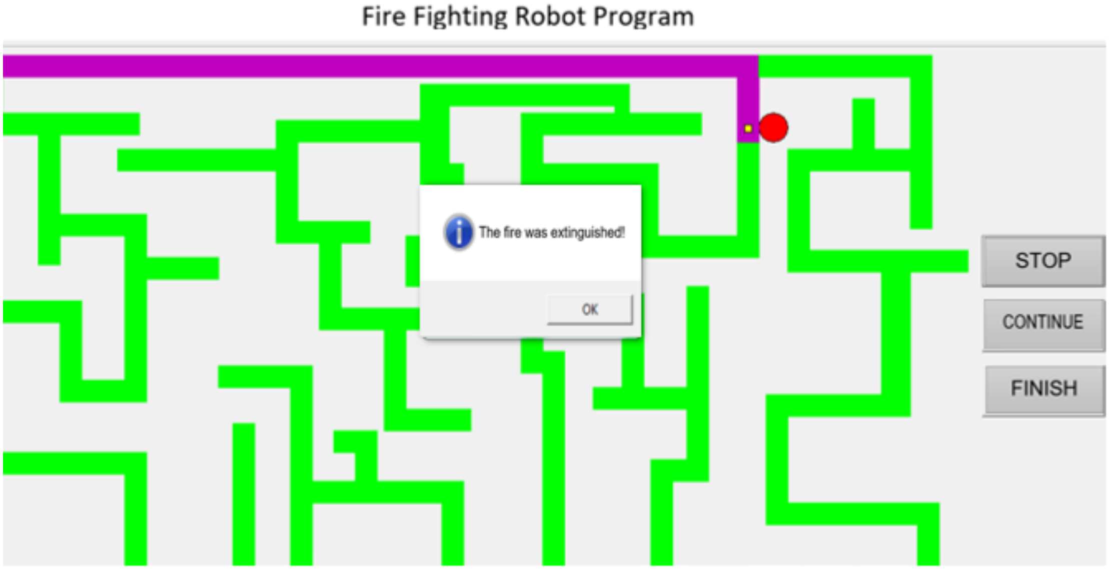


Figure 3. Fire fighting robot program display when start to extinguish the fire

The yellow shape is likened to a firefighting robot which is going to the source of the fire point. The display after it reaches the fire point will be shown in Figure 4:



The fire was extinguished!

Figure 4. Fire fighting robot program display when finish to extinguish fire

The page indicates that the fire has been extinguished and the red shape will turn to white which indicating that the fire has been extinguished. When clicking the command button CONTINUE After the firefighting robot program is run and then we click the CONTINUE command button, the back path will be green and the red shape will show the different hotspots locations. When we click the STOP command button again and it turns out that the fire signal shows a different location, then the path shown is the previous road that has been passed then forward it towards the next location of the hotspot. The process is the same as described in the STOP command button but the difference is that the path, that has been passed, has been saved so it only passes on to the source of the fire. The combination of this algorithm results that from 8 fire point inputs, the fire fighting robot is able to find all the points within 3.12 seconds. This value is obtained from 25 trial. As for the details of trial as follows.

Table 1. Trial time the robot to finds all the fire

Trial	Time (second)	Trial	Time (second)
1	3.69	14	3.17
2	5.01	15	3.16
3	5.03	16	3.18
4	3.52	17	3.15
5	3.30	18	3.16
6	3.62	19	3.14
7	3.43	20	3.15
8	3.33	21	3.13
9	3.20	22	3.12
10	3.21	23	3.12
11	3.19	24	3.12
12	3.18	25	3.12
13	3.18	⋮	Convergen to 3.12

4. CONCLUSION

In this study, we combined the brute-force and backtracking algorithms for the fire fighting robot program using visual basic. Brute force algorithms can be implemented to detect fire spots. Then, the determination of the shortest path to the hotspot location is done by saving the path that has been traversed and then continuing it so that you get the shortest path to the hotspot location. The implementation of backtracking algorithm for doing which using a recursive solution is the right decision because it is suitable for fire-fighting robot programs or the shortest pathfinding program. Based on the results of 25 trial time the robot to finds all the fire, we found that the combination of this algorithm results that from 8 fire point inputs, the fire fighting robot is able to find all the points within 3.12 seconds. Recommendations for further research should use combination between artificial intelligence algorithms with Internet of Things to determine hotspots in area.

REFERENCES

[1] M-H. Huang and R.T. Rust. Artificial Intelligence in Service. Journal of Service Research, Vol. 21, No.2, pp 155-172, 2018.

[2] S. Madakam, R.M. Holmukhe, and D.K. Jaiswal. The Future Digital Work Force: Robotic Process Automation (RPA). Journal of Information Systems and Technology Management. Vol. 16, pp 11-18, 2019.

[3] Q.C. Pham, R. Madhavan, L. Righetti, W.D. Smart, and R. Chatila. The Impact of Robotics and Automation on Working Condition and Employment[Ethical, Legal, and Societal Issues]. IEEE Robotics & Automation Magazine, Vol.25, No.2, 126-128, 2018.

[4] V. Alcacer and V. Cruz-Machado. Scanning the Industry 4.0: A Literatur Review on Technologies for Manufacturing Systems. Engineering Science and Technology, an International Journal, Vol.22, No.3, pp 899-919, 2019.

[5] Y. Tamura, H. Amano, and J. Ota. Analysis of firefighting skill with a teleoperated robot. ROBOMECH Journal, Vol. 7, No.26, pp 1-14, 2020.

[6] M. Aliff, M. Yusof, N.S. Sani, and A. Zainal. Development of Fire Fighting Robot (QRob). International Journal of Advanced Computer Science and Applications. Vol.10, No.1, pp 142-147, 2019.

[7] M.A. Koledoye, D.D. Martini, and Facchinetti. Design of a Mobile Robot for Air Duct Exploration. Robotics, Vol. 6, No.26, pp 1-17, 2017.

[8] E.R.M. Aleluya, A.D. Zamayla, and S.L.M. Tamula. Decision making system of soccer playing robots using finite state machine based on skill hierarchy and path planning through Bezier polynomials. Procedia Computer Science. Vol. 135, pp 230-237, 2018.

[9] R. Balogh and D. Zalek. Using Finite State Machines in Introductory Robotics. International Conference on Robotics and Education RiE. Vol. 829, pp 85-91, 2018.

[10] R.K. Singh and N. Jain. Hybrid Approach for Minimizing Exploration of Multi-Robot System. International Journal of Scientific & Technology Research. Vol. 9, No.4, pp 2428-2433, 2020.

[11] H. Ryu. Graph Search-Based Exploration Method Using a Frontier-Graph Structure for Mobile Robots. Sensors. Vol. 20, No.6270, pp 1-23, 2020.

[12] H.E. Hussieny, S.F.M Assal, and M. Abdellatif. Robotic Exploration: New Heuristic Backtracking Algorithm, Performance Evaluation and Complexity Metric. International Journal of Advanced Robotic System. Vol. 12, No.33. pp 1-12, 2014.

[13] Y.Chen, M.G.H.Bell, and K.Bogenberger. Risk-averse autonomous route guidance by a constrained A* search. Journal of Intelligent Transportation Systems: Technology, Planning, and Operation. Vol. 14, No.3, pp 188-196, 2010.

[14] H. Cipta, and R Widyasari. The Determination of Shortest Path Using Genetic Algorithm Assisted Matlab. International Journal of Information System & Technology. Vol.3, No. 2, pp 302-308, 2020.

[15] L. Shen, H. Shao, L. Zhang, and J. Zhao. The Global Optimal Algorithm of Reliable Path Finding Problem Based on Backtracking Method. Mathematical Problem in Engineering. Vol. 2017, pp 1-11, 2017.

[16] H.A. Priestley and M.P. Ward. A Multipurpose Backtracking Algorithm. Journal of Symbolic Computation, Vol.18, No.1, pp 1-40, 1994.

[17] H.V. Pham, P. Moore, P and D.X. Truong. Proposed Smooth-STC Algorithm for Enhanced Coverage Path Planning Performance in Mobile Robot Applications. Robotics. Vol.8, No.44, pp 1-19, 2019.

[18] L. Prasojo, P.T. Nguyen, O. Tanane, and N. Shahu. Design of Ultrasonic Sensor and Ultraviolet Sensor Implemented on a Fire Fighter Robot Using AT89S52. Journal of Robotics and Control. Vol. 1, No.2, pp 59-63, 2020.