International Journal of Engineering & Technology, 7 (3) (2018) 1400-1404

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i3.13926
Research paper

Obstacle recognition and avoidance during robot navigation in unknown environment

Neerendra Kumar[1*] and Zoltán Vámossy[2]

[1,2]*John von Neumann Faculty of Informatics, Óbuda University, Budapest, Hungary.*
[1]*Department of Computer Science & IT, Central University of Jammu, India.*
[*]*neerendra.kumar@phd.uni-obuda.hu*

Abstract

In this paper, firstly, a model for robot navigation in unknown environment is presented as a Simulink model. This model is applicable for obstacles avoidance during the robot navigation. However, the first model is unable to recognize the re-occurrences of the obstacles during the navigation. Secondly, an advanced algorithm, based on the standard deviations of laser scan range vectors, is proposed and implemented for robot navigation. The standard deviations of the lasers scans, robot positions and the time of scans with similar standard deviations are used to obtain the obstacle recognition feature. In addition to the obstacle avoidance, the second algorithm recognises the re-appearances of the obstacles in the navigation path. Further, the obstacle recognition feature is used to break the repetitive path loop in the robot navigation. The experimental work is carried out on the simulated Turtlebot robot model using the *Gazebo* simulator.

Keywords: *Gazebo simulator; Laser scan; Obstacle avoidance; Obstacle recognition; Robot navigation; Simulink.*

1. Introduction

Autonomous navigation of robots is gaining popularity among the researchers day by day. Various requirements of obstacle avoidance can be satisfied for the changing distance between robot and the obstacles [1]. On the unmarked path, dynamic colour perception model for the autonomous navigation is explained in [2]. Nevertheless, this model is not suitable for the laser range finder sensors. For a robot, motion planning with obstacle avoidance is given in [3]. Using fusion of algorithms, an algorithm for navigation with obstacle avoidance is developed by [4]. Avoidance of unseen obstacles studied in [5]. The complexity of the object matching task rises for similar objects [6]. An algorithm for obstacles avoidance in unknown environment using bumper events of the robot is proposed in [7]. However, the algorithm presented in [7] has the limitation that the robot may struck in the corners and may follow the same loop of the path again and again. A fuzzy controller for obstacle avoidance in unknown environment is presented in [8]. Although, the fuzzy rules, for the obstacles in the central part of the laser scan area, are right or left biased. Therefore, the robot may trap in a repetitive navigation path loop in some situations. Using preceding occurrences, user desired behaviour can be achieved during robot navigation [9]. Priority actions and early predictions based robot navigation algorithm is presented in [10].

Although in the above studies, obstacle recognition based obstacle avoidance is not exploited so far. Therefore, In this paper, obstacle avoidance with obstacle recognition, to break the repetitive path loop during robot navigation in unknown and dynamic environment, is presented.

Remaining part of this paper is structured as follows: A robot navigation model to navigating a robot in unknown environment is presented in Section 2. Section 3 proposes an advanced solution for obstacle avoidance using obstacle recognition. The Experimental set-up and the results are given in Section 4. Section 5 is for the conclusion of the work.

2. Robot navigation model and problem definition

A robot navigation model with obstacle avoidance is given in Fig. 1. The robot and the navigation model communicate each other on robot operation system (ROS) topics using publisher-subscriber strategy. In Fig. 1, the *Odom Subscriber* block is subscribing the */odom* topic of the ROS. Using the *Odom Subscriber* block the given navigation model is receiving the positions of the robot during the navigation. The *'/scan' Subscriber* block of the navigation model is subscribing the */scan* topic of the ROS. On the other hand, the robot is publishing it's pose on the ROS topic */odom* and laser scan messages on the ROS topic */scan*. Therefore the robot positions and its lessor sensor readings are available to the robot navigation model presented by Fig. 1. The *Publish geometry_msgs/Twist* block is publishing the linear and angular velocities, computed by the given model, on the ROS topic *mobile_base/commands/velocity*. On the other side, the robot is subscribes the ROS topic *mobile_base/commands/velocity*, therefore, the robot is capable of receiving the linear and angular velocities published by the given Simulink model. The *Path positions data Writing* block contains a MATLAB function to write the robot positions in a text file for the future use. The Algorithm 1 is used in *MATLAB Function* block of the Fig. 1. The input parameters of the function defined in Algorithm 1 are corresponding to the input

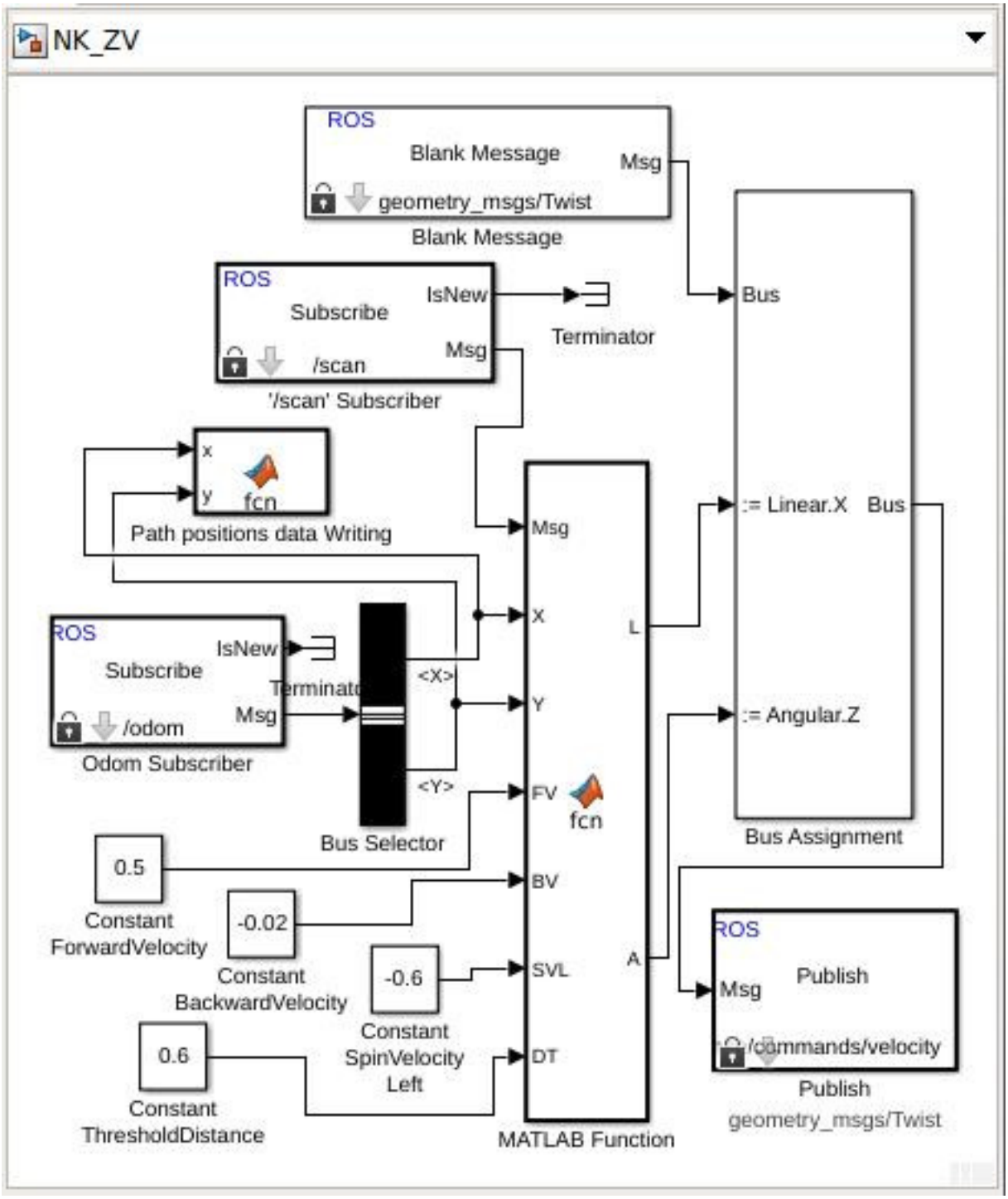

Figure 1: Robot navigation model in Simulink.

parameters of *MATLAB Function* block of Fig. 1 from *Msg* to *DT*, respectively in that order. The robot navigates on a repetitive path using the negative or positive angular velocity. The rest of the blocks in the Simulink model are self explanatory.

The robot navigation model (Fig. 2), using the Algorithm 1 in it's *MATLAB Function* block, is executed with the simulated Turtlebot robot in Gazebo world depicted by Fig. 2. The Gazebo world is constructed in the Gazebo simulator using the *Gray_wall* available in the model database. To increase the complexity,the Gazebo world is kept closed so that the robot can navigate in a path loop. In addition, the Gazebo world is built by a single type of walls so that the complexity of the obstacle recognition may be raised. Initially, The Turtlebot robot model is present at the $(0,0)$ coordinate position and heading towards the positive X axis direction. For better visibility and identification of the robot in the Gazebo world, a spot light is taken into consideration.

Following the Algorithm 1, the robot follows a straight path (i.e. with zero angular velocity) in the forward direction until it finds any obstacle closer than a threshold distance. When the robot finds any nearby obstacle (closer than threshold distance) then it is instructed to spin in left (using positive angular velocity) or right (using negative angular velocity) side of its heading direction. However, in either of the cases (i.e. left or right spins) the robot traps in a repetitive navigation path loop and continue to navigate on almost same path loop. In this situation the other possible paths remain unexplored.

The navigation path followed by the robot is shown in Fig. 3. In

Algorithm 1 Obstacles avoidance algorithm

1: **function** FCN($M, X, Y, v_c, v_b, \omega_c, D_t$)
2: Receive scan reading from robot;
3: $\mathbf{R} \leftarrow$ scan ranges received from M;
4: $D_{min} \leftarrow \min(\mathbf{R})$
5: **if** $(D_{min} < D_t)$ **then**
6: $\omega \leftarrow \omega_c$;
7: $v \leftarrow -|v_b|$;
8: **else**
9: $\omega \leftarrow 0$;
10: $v \leftarrow v_c$;
11: **end if**
12: Store X, Y;
13: **return** ω, v;
14: **end function**

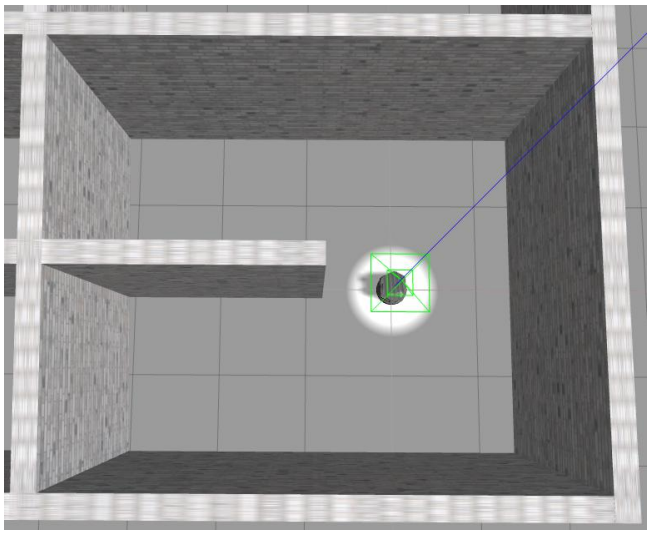

Figure 2: Simulated Turtlebot robot in a Gazebo world.

fact, in Fig. 3, the robot navigation path plot is combined with the Gazebo world (given in Fig. 2) to make the navigation path more understandable with respect to the coordinate system point of view as well. In Fig. 3, the upper path loop (i.e. in positive Y coordinates) is with positive angular velocity. On the other-hand, The lower path loop (i.e. in negative Y coordinates) is received by using negative angular velocity.
In the situation explained by Fig. 3, the robot will repeatedly follow the same path loop and will not turn to the other possible paths. However, for the searching robots, it may be required to explore the whole area. To overcome from the problem, an advanced solution is presented by Algorithm 2 in the Section 3 ahead.

3. Advanced algorithm for obstacle avoidance with obstacle recognition

For obstacle range detection, the laser scan is a trustworthy technique. Typically, a laser scan contains n numbers of distance range readings. The distance range vector $(\mathbf{R})$, of a laser scan, can be given by (1) as below:

$$\mathbf{R} = [r_i], \text{ for } i = 1 \text{ to } n. \tag{1}$$

Where, r is a distance range of an obstacle point.
Standard deviation $(\sigma_{\mathbf{R}})$ of $\mathbf{R}$ is expressed in (2) as follows:

$$\sigma_{\mathbf{R}} = \sqrt{\frac{\sum_{i=1}^{n}(r_i - \bar{r})^2}{n}}. \tag{2}$$

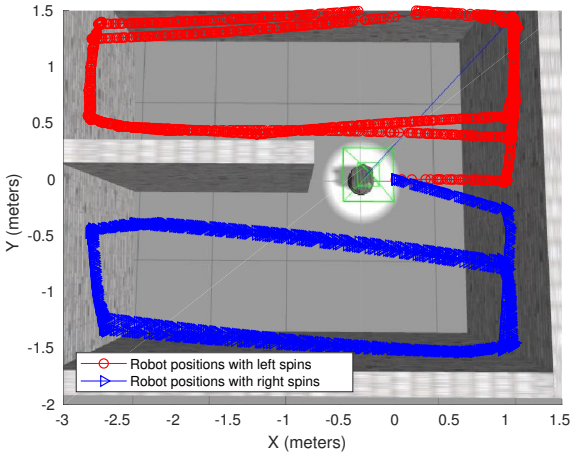

Figure 3: Repetitive paths followed by robot (using the proposed Simulink model given in Fig. 1) in Gazebo world.

Where, $\bar{r}$ is the mean of distance range vector $\mathbf{R}$.

The standard deviation of the laser scans of the similar objects will produce same results. Therefore, the standard deviation values of laser scans of objects can be used to identify and differentiate the objects scanned.

Using standard deviations, the Algorithm 2 gives the solution of the problem of repetitive paths (as discussed in Section 2) and breaks the repetitive path loop in robot navigation.

At the step 27 of the Algorithm 2, there is a call to the **REVVEL**(e) procedure. The **REVVEL**(e) procedure is explained in Algorithm 3.

Table 1 presents the variables' descriptions which are taken through Algorithms 1−3.

According to the statement 25 of the Algorithm 2, if the standard deviations, robot positions of the two scans are similar and the time difference between the two scans is larger than a threshold value then this is the re-occurrence of an obstacle.

Table 1: Variables' descriptions for Algorithms 1−3.

Variable	Description
T_c	Current time of the system
V	Y coordinates vector of positions when closer obstacles
U	X coordinates vector of positions when closer obstacles
v	Linear velocity of robot
Y	Y coordinates vector of all positions
X	X coordinates vector of all positions
ω	Angular velocity of robot
S	Standard deviations vector of scan ranges
T	Vector of scans times

Algorithm 2 Obstacles recognition and avoidance algorithm

Create vectors: **Y**, **S**, **U**, **T**, **V**, **X**;
Initialize: D_t, R_m, c, e, g, T_d, T_f, T_t, T_s, P_t, ω_c, σ_t, v_b, v_c;
while $((T_c - T_s) \leq T_f)$ **do**
    Read laser scan from robot;
    R $\leftarrow$ Distance range vector received from robot;
    for $j \leftarrow 1$ to length (**R**) **do**
        if (**R**(j) is not defined) **then**
            R$(j) \leftarrow R_m$;
        end if
    end for
    $D_{min} \leftarrow \min(\mathbf{R})$;
    Y$(g) \leftarrow Y$ coordinate of the robot position;
    X$(g) \leftarrow X$ coordinate of the robot position;
    Increment g by 1 ;
    if $(D_{min} < D_t)$ **then**
        V$(c) \leftarrow Y$ coordinate of the robot position;
        U$(c) \leftarrow X$ coordinate of the robot position;
        S$(c) \leftarrow \sigma_{\mathbf{R}}$;
        T$(c) \leftarrow$ time at which scan is performed;
        Increment c by 1 ;
        for $k \leftarrow 1$ to (length (**S**) $- 1$) **do**
            $T_d \leftarrow |\mathbf{T}(k) - \mathbf{T}(c)|$;
            $\sigma_r \leftarrow |(\mathbf{S}(c) - \mathbf{S}(k))|$;
            $D_{c,k} \leftarrow \sqrt{(\mathbf{V}(c) - \mathbf{V}(k))^2 + (\mathbf{U}(c) - \mathbf{U}(k))^2}$;
            if ($D_{c,k} < P_t$ & $\sigma_r < \sigma_t$ & $T_d > T_t$) **then**
                $e \leftarrow e + 1$;
                Call Procedure **REVVEL**(e);
            end if
        end for
        $\omega \leftarrow \omega_c$;
        $v \leftarrow -|v_b|$;
    else
        $\omega \leftarrow 0$;
        $v \leftarrow v_c$;
    end if
    Drive the robot using v, ω;
end while

Algorithm 3 Procedure to reverse the angular velocity of the robot

procedure REVVEL(e)
    if $(e == 1)$ **then**
        $\omega_c \leftarrow -\omega_c$;
        $T_1 \leftarrow$ current time
    end if
    $T_2 \leftarrow$ current time
    if $(T_2 - T_1 > T_d)$ **then**
        $e \leftarrow 0$;
    end if
end procedure

4. Experimental set-up and results

The proposed algorithms are implemented using a general purpose laptop equipped with *4xIntel Core i3 CPU @ 230 GHz*. The memory of the system is 3952MB. *Ubuntu 16.04.4 LTS* operating system is used. *Robotics System Toolbox* in *MATLAB (R2018a)* is applied for computer programs. Simulated *Turtlebot* robot model is taken in *Gazebo 7.0* simulator. The numerical values of the variables, initialized in Algorithms 1–3, are presented in Table 2.

The standard deviations of the distance range vectors of the laser scans taken through out the robot navigation are presented in the Fig. 4. It can be seen from the Fig. 4 that the fluctuations of the standard deviations are periodic for most of the duration. It is because of the same type of obstacles found in the navigation path. Fig. 5 shows the resultant path of the robot. It is clear from the Fig. 5 that robot automatically reverses the angular velocity on the re-appearance of the obstacle. In Fig. 4, the initialized angular velocity is negative (See the value of ω_c in Table 2). As a result, the robot navigates towards the negative Y coordinates and re-visits back to the position near about (1, 1.5) coordinate position. After reaching at this point, the robots completes its one cycle of the path in the negative Y axis coordinates. Here, the robot detects the re-visit to the obstacle using the standard deviations, position of itself and the time difference between the scans with similar standard deviations. At this position (i.e. near about (1, 1.5) coordinates position), the angular velocity of the robot reverses and the robot starts the spin in the opposite side (i.e. left side). In this manner, the robot does not enter the previous navigation path loop and heads towards the positive Y axis coordinates.

Table 2: Initialization of variables for the Algorithms 1–3.

Variable	Description	Initialized Value
T_f	Finishing time	190 (seconds)
T_d	Time difference of scans	10 (seconds)
T_t	Time threshold	10 (seconds)
T_s	Time of start	0
D_t	Distance threshold of robot and obstacles	0.6 (meter)
P_t	Distance threshold of two positions	0.2 (meter)
σ_t	Threshold fluctuation of σ of scans	0.0005
ω_c	Angular velocity of robot	-0.6 (radian/sec.)
v_b	Backward linear velocity	-0.02 (meter/sec.)
v_c	Forward linear velocity	0.5 (meter/sec.)
e	a counter variable	0
g	a counter variable	0
c	a counter variable	0
R_m	Maximum distance range of laser sensor	Default

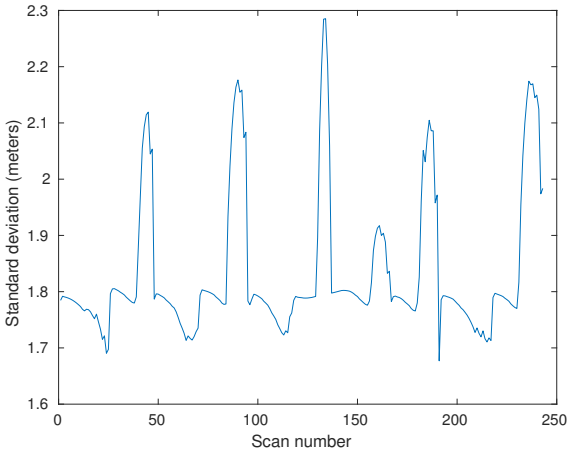

Figure 4: Standard deviations of the laser scans during robot navigation.

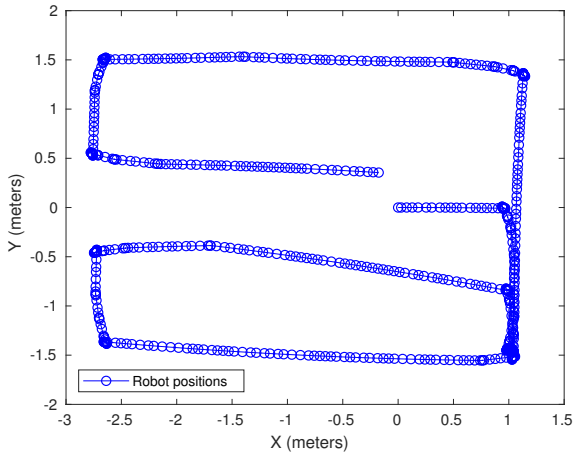

Figure 5: Path followed by robot using advanced Algorithms 2−3.

5. Conclusion

A Simulink model for robot navigation in unknown environment has been proposed and implemented. The Algorithm 1 is capable of obstacle avoidance with repetitive path results. By implementing the Algorithm 2 in combination of the procedure given by Algorithm 3, the robot comes out of the repetitive path and successfully reverses its angular velocity when the same obstacle is found on the next iteration of the path loop. The re-appearances of obstacles are successfully recognized and avoided. Moreover, the obstacle recognition is used to drive the robot to the yet unexplored path. In spite of the similar obstacles in the navigation path, the re-occurrences of the obstacles has been detected using the robot positions and the time of laser scans performed.

Acknowledgement

This study is sponsored by the Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University, Budapest, Hungary. The experimental work is carried out as a part of the TekiControl research project of the university.

References

[1] L. Zong, J. Luo, M. Wang, and J. Yuan, "Obstacle avoidance handling and mixed integer predictive control for space robots," *Advances in Space Research*, vol. 61, no. 8, pp. 1997 – 2009, 2018.

[2] A. Narayan, E. Tuci, F. Labrosse, and M. H. M. Alkilabi, "A dynamic colour perception system for autonomous robot navigation on unmarked roads," *Neurocomputing*, vol. 275, pp. 2251 – 2263, 2018.

[3] Y. Zhao, X. Chai, F. Gao, and C. Qi, "Obstacle avoidance and motion planning scheme for a hexapod robot octopus-iii," *Robotics and Autonomous Systems*, vol. 103, pp. 199 – 212, 2018.

[4] H. Mousazadeh, H. Jafarbiglu, H. Abdolmaleki, E. Omrani, F. Monhaseri, M. r. Abdollahzadeh, A. Mohammadi-Aghdam, A. Kiapei, Y. Salmani-Zakaria, and A. Makhsoos, "Developing a navigation, guidance and obstacle avoidance algorithm for an unmanned surface vehicle (usv) by algorithms fusion," *Ocean Engineering*, vol. 159, pp. 56 – 65, 2018.

[5] A. I. Ross, T. Schenk, J. Billino, M. J. Macleod, and C. Hesse, "Avoiding unseen obstacles: Subcortical vision is not sufficient to maintain normal obstacle avoidance behaviour during reaching," *Cortex*, vol. 98, pp. 177 – 193, 2018.

[6] G. Kertész, S. Szénási, and Z. Vámossy, "Multi-directional image projections with fixed resolution for object matching," *Acta Polytechnica Hungarica*, vol. 15, no. 2, pp. 211 – 229, 2018.

[7] N. Kumar, Z. Vámossy, and Z. M. Szabó-Resch, "Robot obstacle avoidance using bumper event," in *2016 IEEE 11th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, May 2016, pp. 485–490.

[8] N. Kumar, M. Takács, and Z. Vámossy, "Robot navigation in unknown environment using fuzzy logic," in *2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI)*, Jan 2017, pp. 279–284.

[9] L. Nardi and C. Stachniss, "User preferred behaviors for robot navigation exploiting previous experiences," *Robotics and Autonomous Systems*, vol. 97, pp. 204 – 216, 2017.

[10] F. Kamil, T. S. Hong, W. Khaksar, M. Y. Moghrabiah, N. Zulkifli, and S. A. Ahmad, "New robot navigation algorithm for arbitrary unknown dynamic environments based on future prediction and priority behavior," *Expert Systems with Applications*, vol. 86, pp. 274 – 291, 2017.