# Laser-based Obstacle Avoidance Algorithm for Four Driving/Steering Wheels Autonomous Vehicle

B. Dumitrascu, A. Filipescu, G. Petrea
Department of Automation and Electrical Engineering,
"Dunarea de Jos" University of Galati ,
Galati, Romania

Eugenia Minca
Department of Automation, Computer Science and Electrical Engineering,
"Valahia" University of Targoviste,
Targoviste, Romania

S. Filipescu
Polytechnic University of Bucharest,
Bucharest, Romania

Alina Voda
"Joseph Fourier" University 1/CNRS, -GIPSA-lab,
Grenoble, France

***Abstract*— In this paper an algorithm for trajectory-tracking with obstacle avoidance for autonomous vehicles is presented. The algorithm tracks a trajectory composed of an intended global trajectory that was previously generated and local obstacle avoidance trajectories that are created when an obstacle is detected by the laser. A discrete-time sliding-mode controller is used by the autonomous vehicle to follow the global trajectory and the local obstacle avoidance trajectories. Quintic equations are used to generate the paths used by the vehicle to avoid the obstacles. The velocity, acceleration, angular velocity and angular acceleration profiles are calculated for the generated paths and fed to the sliding-mode controller.**

***Keywords—obstacle avoidance; formatting; style; styling; insert (key words)***

## I. INTRODUCTION

Most of the accidents are caused by human error and the use of autonomous vehicles can drastically reduce the number of accidents. Autonomous vehicles have the potential to replace drivers, thus eliminating the unpredictable behavior of the drivers. Autonomous vehicles can replace humans if they are capable of calculating a trajectory that will take them to the intended target and are capable of following this trajectory until they reach the destination. Consider that the map of the operating environment is known and a global trajectory can be calculated off-line so that the vehicle will follow it to its destination. Such a trajectory can be easily followed using a sliding-mode controller, like in [2]-[16]. Sliding-mode control is a robust approach that is well suited for applications involving nonlinear control. The sliding-mode control methodology has been presented in [1].

When using a map of the operating environment, it is imposible to take into account unknown obstacles that can block the intended trajectory and the vehicle can't reach its target. The solution to this problem is to include an obstacle avoidance module that will allow the vehicle to avoid the blocked part of the trajectory and return to the global trajectory.

*This work was supported by UEFISCDI, project number PN-II-ID-PCE-2011-3-0641

978-1-4799-2228-4/13/$31.00 ©2013 IEEE

This solution ensures that the vehicle reaches the destination, but any obstacle that is avoided will increase the time needed by the vehicle to reach the destination. One of the simplest obstacle avoidance algorithms is "the bug", proposed by Lumelsky et al. [20]. This algorithm requires the vehicle to circle the obstacle before determining the best point to continue towards its goal. Other obstacle avoidance algorithms use virtual potential fields, like in [18] to determine the path the vehicle has to take in order to avoid the obstacle. The algorithms consider that the vehicle is subjected to forces that pull the vehicle towards the goal and rejecting forces that pull the robot away from the obstacles. The robot's trajectory results from the composition of these forces. A vector field histogram is used in [17] for obstacle avoidance, that uses a histogram to reduce the errors from the sensors.

A bubble rebound algorithm is used in [19] to avoid obstacles. The algorithm uses a sensitivity bubble to determine if an obstacle is blocking the path. If an obstacle is detected the algorithm calculates the path that has the minimum density of obstacles and moves in that direction until the goal is visible or a new obstacle is detected.

An algorithm capable of tracking a desired trajectory and avoiding obstacles that block the desired path by generating local trajectories and by following them until the initial trajectory can be resumed, is proposed in this paper. The trajectory followed by the robot is generated by merging the global trajectory and the local avoidance trajectory. This trajectory can be tracked using only one discrete-time sliding-mode controller. Readings from a SICK LMS111 laser and the sensitivity bubble from [19] are used to detect if an obstacle is blocking the path. The algorithm is tested for the SEEKUR autonomous vehicle (Fig.1). Seekur is a holonomic, all-weather, outdoor robot platform for outdoor security, inspection and research. The tests are performed using MobileSim.


Fig. 1. SEEKUR autonomous vehicle

## II. OBSTACLE AVOIDNACE ALGORITHM

It is desired to track the global trajectory as long as no obstacle is detected. When an obstacle is detected a local trajectory for the avoidance of the obstacle is calculated and the robot will track this new trajectory until the obstacle has been avoided and afterwards will return to tracking the given trajectory. The robot uses the same discrete-time sliding-mode control for both the desired trajectory and the local avoidance trajectory.

The discrete-time sliding-mode controller used is presented in [14]:

$$v[k+1]=\frac{1}{\cos\theta_e[k]\cdot T_s}\cdot[-(1-q_1\cdot T_s)\cdot s_1[k]+\varepsilon_1\cdot \cdot T_s\cdot \mathrm{sgn}(s_1[k]-x_e[k+1]\cdot(1+k_1)-(v_d[k+1]- v_r[k]\cdot\theta_e[k+1]\cdot\sin\theta_e[k]-\omega_d[k+1]\cdot y_e[k]- \omega_d[k]\cdot y_e[k+1])\cdot T_s] \quad (1)$$

$$\omega[k]=\frac{1}{v_r[k]\cdot\cos\theta_e+k_0\cdot\mathrm{sgn}\,y_e[k+1]}\cdot[-(1- q_2\cdot T_s)\cdot s_2[k]+\varepsilon_2\cdot T_s\cdot\mathrm{sgn}(s_2[k])-\theta_e[k]- y_e[k+1]\cdot(k_2+1)-(v_r[k+1]\cdot\sin\theta_e[k]+ \omega_d[k+1]\cdot x_e[k]-\omega_d[k]\cdot x_e[k+1])\cdot T_s)]+\omega_d[k] \quad (2)$$

where $v$ is the calculated velocity ; $\omega$ is the calculated angular velocity; $v_r$ is the velocity of the vehicle; $v_d$ is the desired velocity; $\omega_d$ is the desired angular velocity; $s_1$ and $s_2$ are the sliding surfaces; $x_e$ is the trajectory-tracking error on the X-axis; $y_e$ is the trajectory-tracking error on the X-axis; $\theta_e$ is the trajectory-tracking error of the heading; $\varepsilon_1,\varepsilon_2>0$ are the reaching velocities, $q_1,q_2>0$ are the converging exponentials.

The detection of the obstacles is achieved using a SICK LMS111 laser. The laser covers an area of 180 degrees and allows the detection of any obstacle that can block the vehicle's path. Fig. 2 presents the detection area of the laser mounted on the SEEKUR. The resolution of the laser is $1^{\circ}$ and 181 readings are generated. In order to reduce the amount of calculations a resolution of $5^{\circ}$ is used and 37 distance measurements are considered.

For the obstacle detection the sensitivity bubble from [19] is used. This bubble is dependent on the vehicle's speed and allows for a good detection of the obstacles. Using safety coefficients and the robot's speed, a safety area is defined inside the sensibility bubble and any obstacle detected inside this area will trigger the obstacle avoidance procedure. The components for the sensitivity bubble are calculated using

$$b_i = k_i \cdot v \cdot T_s \,. \quad (3)$$

where: $i$ is the number of the reading; $b_i$ is the component of the sensitivity bubble corresponding to reading $i$; $k_i$ is the safety coefficient of the bubble corresponding to reading $i$; $v$ is the velocity of the robot; $T_s$ is the sample time.

Fig. 3 presents an example of an obstacle that was detected by the sensitivity bubble. An obstacle is detected when the distance to the obstacle measured by any of the laser is lower than the component of the sensitivity bubble corresponding to that laser reading

$$Range(i) < b_i \,. \quad (4)$$

First a global desired trajectory is generated off-line using a program that calculates the desired velocity, acceleration, angular velocity and angular acceleration for each time sample. A program for generating the global trajectory is presented in [6].

Before imposing the trajectory to the vehicle, the distances measured by the laser are collected and the sensitivity bubble is calculated. Using this information the algorithm determines if an obstacle was detected. As long as no obstacle is detected the desired trajectory is fed to the controller and the robot tracks this trajectory to its destination.

If an obstacle is detected the obstacle the obstacle avoidance procedure is triggered. In Fig. 3 an example of how the sensibility bubble detects an obstacle is presented. The vehicle avoids the obstacle if a new local trajectory is generated that avoids the obstacle and returns to the global trajectory. The local trajectory for the obstacle from Fig.3 is presented in Fig. 4.

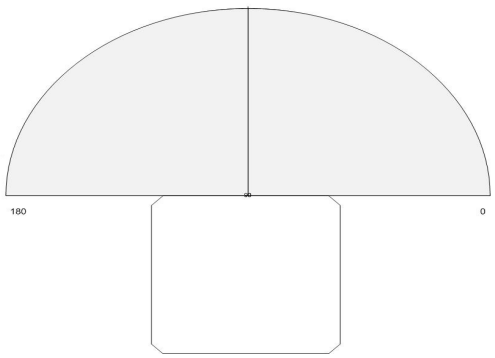


Fig. 2. Laser detection zone

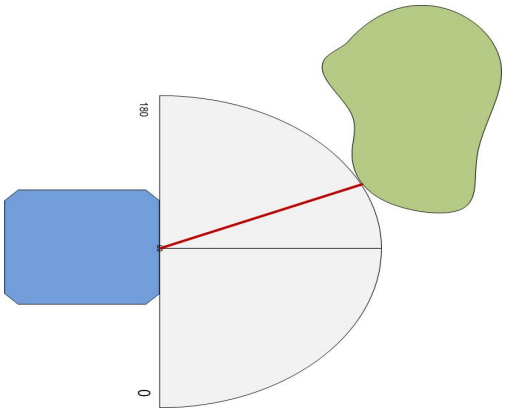


Fig. 3. Detected obstacle

We define 2 search areas for the new trajectory bounded by the direction of travel. The first 17 readings determine the right area for avoidance and the last 17 the left area for avoidance. The search will begin in the area opposite to the area with the closest obstacle. If a solution was not found in the first area a new search is performed for the other area. found. First the coordinates of the point of the trajectory that allows the obstacle to be avoided are calculated. This point is situated on the line perpendicular to the direction of travel from the point where the closest obstacle was detected.

Consider $h_{\min}$ the width of the SEEKUR plus a small safety margin, the minimum width of an area that allows the passage of the vehicle. After the search area was selected we search for the closest reading to the direction of travel that does not detect an obstacle. The algorithm determines the points where the laser beams intersect the perpendicular from the point where the obstacle was detected to the direction of travel. The algorithm tries to find a segment determined by the intersection points, starting from the intersection point of the selected beam, which is at least as long as $h_{\min}$. The desired point is located at the middle of this segment.

The segment is searched using

$$h = m\_range \cdot \tan(angle(j)) - Ze\,, \tag{5}$$

where: $h$ is the length of the segment, $m\_range$ is the distance to the closest obstacle projected on the direction of travel, $angle(j)$ is the angle formed by the beam, $Ze = m\_range \cdot \tan(angle(i))$ is the exclusion zone, $i$ and $j$ are the laser beams used at the current step.

The coordinates are determined using

$$\begin{cases} x = x_r + m\_range \cdot \cos(\theta_r) - (\frac{h}{2} + Ze) \cdot \sin(\theta_r) \\ y = y_r - m\_range \cdot \cos(\theta_r) - (\frac{h}{2} + Ze) \cdot \sin(\theta_r) \end{cases}. \tag{6}$$

The path from the current vehicle's position to $P(x, y,)$ is determined using Quintic equations [6]. The formula of the Quintic $G^2$-splines is

$$p_{i,i+1}(u) = \begin{bmatrix} \alpha_{i0} + \alpha_{i1} \cdot u + \alpha_{i2} \cdot u^2 + \ldots + \alpha_{i5} \cdot u^5 \\ \beta_{i0} + \beta_{i1} \cdot u + \beta_{i2} \cdot u^2 + \ldots + \beta_{i5} \cdot u^5 \\ \theta_i(u) \end{bmatrix} \tag{7}$$

$$\alpha_{i0} = x_i(0)\,, \tag{8}$$

$$\alpha_{i1} = g_1 \cdot \cos(\theta_i(0))\,, \tag{9}$$

$$\alpha_{i2} = \frac{1}{2}\left(g_3 \cdot \cos(\theta_i(0)) - g_1^2 \cdot k_i \cdot \sin(\theta_i(0))\right), \tag{10}$$

$$\begin{aligned} \alpha_{i3} = {} & 10 \cdot (x_{i+1} - x_i) - \left(6 \cdot g_1 + \frac{3}{2} \cdot g_3\right)\cos(\theta(0)) - \\ & \left(4 \cdot g_2 - \frac{1}{2} g_4\right)\cos(\theta_{i+1}(0)) + \frac{3}{2} \cdot g_1^2 \cdot k_i \cdot \sin(\theta_i(0))\,, \\ & -\frac{1}{2} g_2^2 \cdot k_{i+1} \cdot \sin(\theta_{i+1}(0)) \end{aligned} \tag{11}$$

$$\begin{aligned} \alpha_{i4} = {} & -15 \cdot (x_{i+1} - x_i) + \left(8 \cdot g_1 + \frac{3}{2} \cdot g_3\right)\cos(\theta(0)) + \\ & (7 \cdot g_2 - g_4)\cos(\theta_{i+1}(0)) - \frac{3}{2} \cdot g_1^2 \cdot k_i \cdot \sin(\theta_i(0)) + \\ & g_2^2 \cdot k_{i+1} \cdot \sin(\theta_{i+1}(0)) \end{aligned} \quad , \tag{12}$$

$$\begin{aligned} \alpha_{i5} = {} & 6 \cdot (x_{i+1} - x_i) - \left(3 \cdot g_1 + \frac{1}{2} \cdot g_3\right)\cos(\theta(0)) - \\ & \left(3 \cdot g_2 - \frac{1}{2} g_4\right)\cos(\theta_{i+1}(0)) + \frac{1}{2} \cdot g_1^2 \cdot k_i \cdot \sin(\theta_i(0))\,, \\ & -\frac{1}{2} g_2^2 \cdot k_{i+1} \cdot \sin(\theta_{i+1}(0)) \end{aligned} \tag{13}$$

$$\beta_{i0} = y_i(0)\ , \tag{14}$$

$$\beta_{i1} = g_1 \cdot \sin(\theta_i(0))\,, \tag{15}$$

$$\beta_{i2} = \frac{1}{2}\left(g_3 \cdot \sin(\theta_i(0)) - g_1^2 \cdot k_i \cdot \cos(\theta_i(0))\right), \tag{16}$$

$$\begin{aligned} \beta_{i3} = {} & 10 \cdot (y_{i+1} - y_i) - \left(6 \cdot g_1 + \frac{3}{2} \cdot g_3\right)\sin(\theta(0)) - \\ & \left(4 \cdot g_2 - \frac{1}{2} g_4\right)\sin(\theta_{i+1}(0)) + \frac{3}{2} \cdot g_1^2 \cdot k_i \cdot \cos(\theta_i(0)) - , \\ & \frac{1}{2} g_2^2 \cdot k_{i+1} \cdot \cos(\theta_{i+1}(0)) \end{aligned} \tag{17}$$

$$\beta_{i4} = -15\cdot(y_{i+1}-y_i)+\left(8\cdot g_1+\frac{3}{2}\cdot g_3\right)\sin(\theta(0))$$
$$+(7\cdot g_2-g_4)\sin(\theta_{i+1}(0))-\frac{3}{2}\cdot g_1^2\cdot k_i\cdot\cos(\theta_i(0)), \quad (18)$$
$$+g_2^2\cdot k_{i+1}\cdot\cos(\theta_{i+1}(0))$$

$$\beta_{i5} = 6\cdot(y_{i+1}-y_i)-\left(3\cdot g_1+\frac{1}{2}\cdot g_3\right)\sin(\theta(0))-$$
$$\left(3\cdot g_2-\frac{1}{2}g_4\right)\sin(\theta_{i+1}(0))+\frac{1}{2}\cdot g_1^2\cdot k_i\cdot\cos(\theta_i(0)), \quad (19)$$
$$-\frac{1}{2}g_2^2\cdot k_{i+1}\cdot\cos(\theta_{i+1}(0))$$

where $k_i$, $k_{i+1}$ are scalar curvatures and can be arbitrary set, $g_1$, $g_2$, $g_3$, $g_4$ are the parameters that determine the shape of the curve.

For each element of this path the velocity ($v[n]$), angular velocity ($\omega[n]$), acceleration ($a[n]$) and angular acceleration ($a\omega$) are calculated using:

$$l[n] = \sqrt{(x[n]-x[n-1])^2+(y[n]-y[n-1])^2}\ ; \quad (20)$$

$$v[n] = l[n]/T_s\ ; \quad (21)$$

$$\omega[n] = a\tan 2(y[n]-y[n-1], x[n]-x[n-1])/T_s\ ; \quad (22)$$

$$a[n] = v[n]/T_s\ ; \quad (23)$$

$$a\omega[n] = \omega[n]/T_s\ . \quad (24)$$

The desired trajectory for obstacle avoidance is presented in Fig. 4 with a green line; the blocked trajectory is marked with a red line.

When the vehicle reaches the end point of the trajectory it will move forward in a straight line with constant velocity until the obstacle has been cleared. Once the obstacle has been cleared a return trajectory is calculated and the initial trajectory is resumed. The schematic of the obstacle avoidance algorithm is presented in Fig. 5.

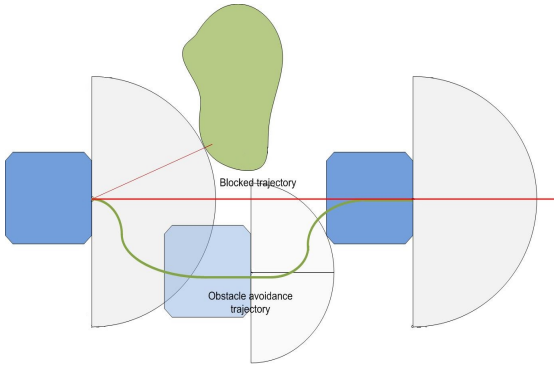


Fig. 4. The obstacle avoidance trajectory

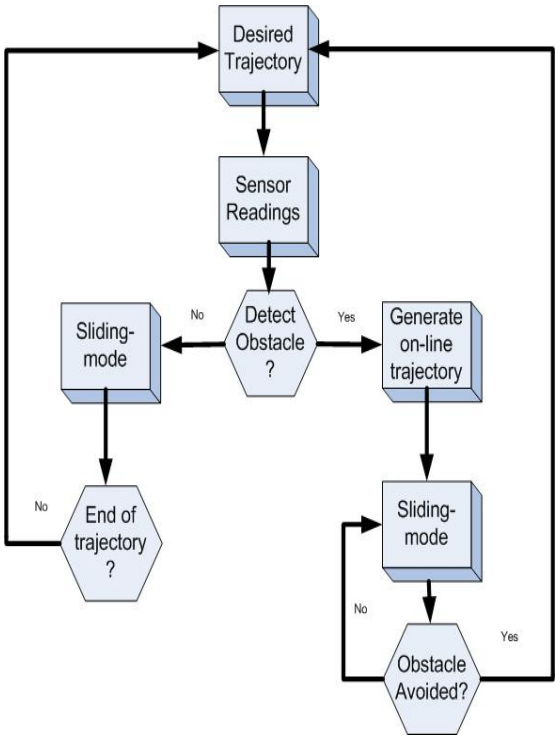


Fig. 5. Schematic of the obstacle avoidance algorithm

## III. Simulation Results

Simulation results are presented in order to validate the proposed algorithm for obstacle avoidance. The control program is written in C++ and run on a PC, with sampling time set to $T_s$=100 ms. All the simulations were made using MobileSim. MobileSim is a software designed to simulate the behavior of robots produced by Adept Mobile Robots Inc. A map with the simulated obstacles was created using Maper3Basic and loaded into Mobilesim.

The desired trajectory is presented in Fig. 6. The trajectory-tracking was first tested in the absence of any obstacle and the vehicle followed the intended trajectory with small tracking errors. The resulting trajectory is presented in Fig. 7. It can be seen that the controller can track the intended trajectory in the absence of obstacles.

The obstacle avoidance performance of the algorithm is tested using the trajectory from Fig. 6, as the global trajectory intended to be tracked and one obstacle blocking this path. Fig. 8 presents the MobileSim simulated trajectory of the vehicle obtained after avoiding the obstacle. Fig. 9 presents the complete simulated trajectory. It can be seen that the algorithm allows the vehicle to avoid the obstacles and continue on its initial trajectory when the path is clear.

Fig. 10 presents the simulated trajectory and the desired trajectory resulting from the initial trajectory and the local avoidance trajectories. Fig. 11 presents the trajectory-tracking errors for the X-axis. In Fig. 12 the trajectory-tracking errors for the Y-axis are presented. Fig. 13 presents the trajectory-

tracking errors for the heading. From these figures it can be determined that the algorithm is capable of tracking a desired trajectory and avoid obstacles, that block the intended path, and resume the initial trajectory afterwards. Forcind the change between the global and the local avoidance trajectory causes an increase in the trajectory-tracking errors but the controller is still able to reach the goal. One of the causes for the larger tracking errors can be attributed to the delay caused by the time required to obtain the laser readings and process the information, which leads to higher delays when sending command inputs as opposed to the case when only the sliding-mode control is used and the laser is not used.

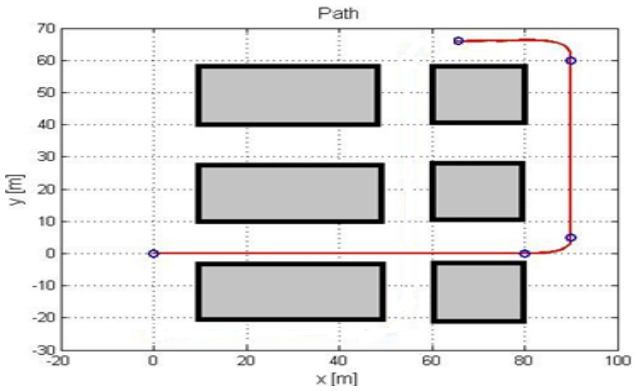


Fig. 6. Desired trajectory

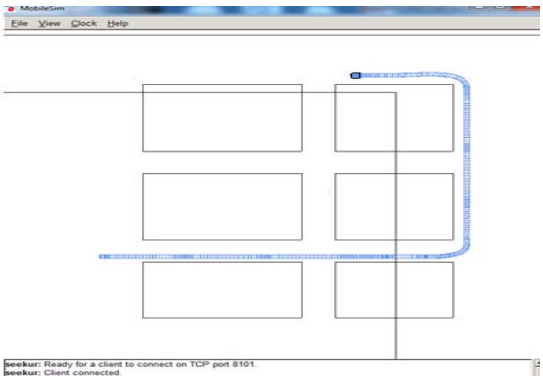


Fig. 7. The MobileSim simulated trajectory without obstacles

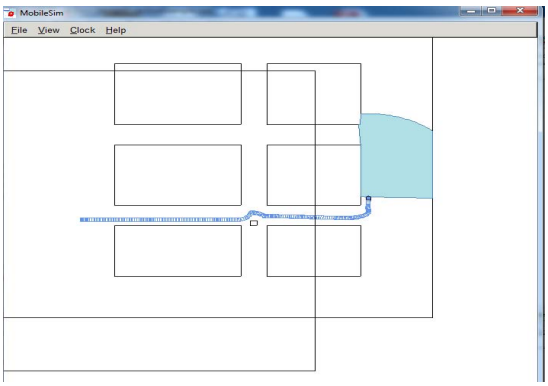


Fig. 8. MobileSim after the obstacle avoidance

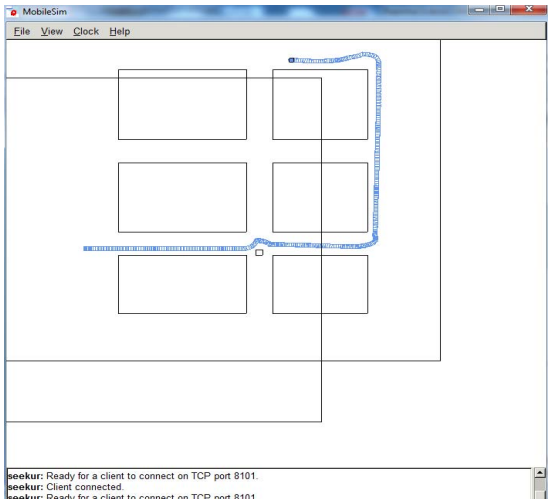


Fig. 9. The MobileSim simulated trajectory with 1 obstacle blocking the intended trajectory

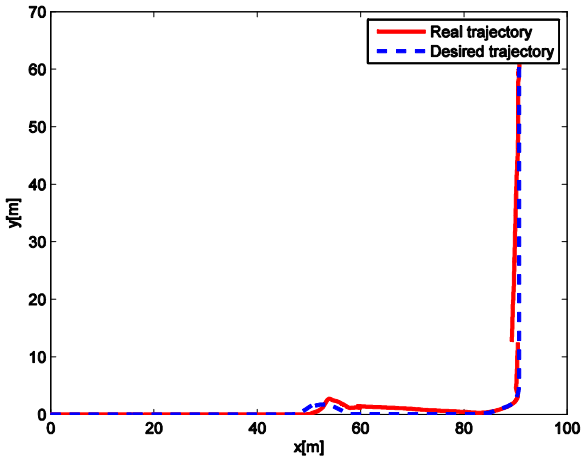


Fig. 10. The simulated trajectory and the desired trajectory

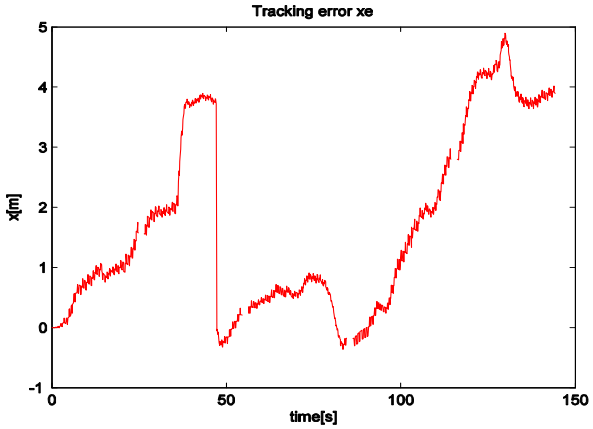


Fig. 11. The trajectory-tracking errors for the X-axis

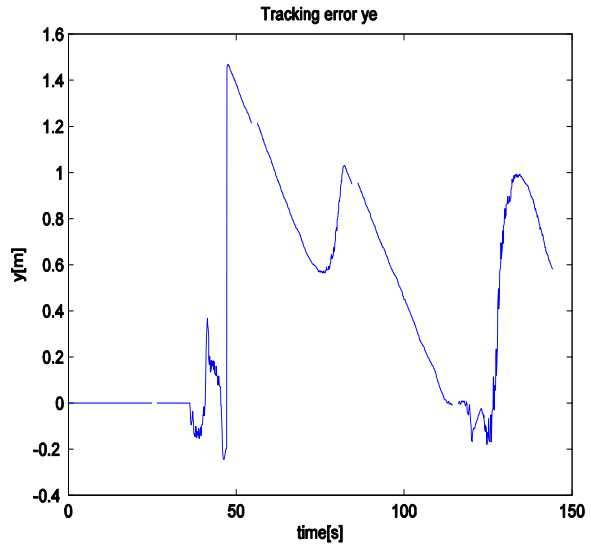


Fig. 12. The trajectory-tracking errors for the Y-axis

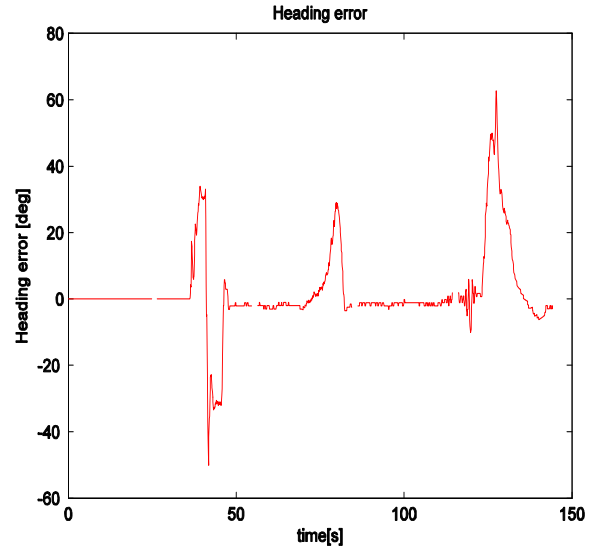


Fig. 13. The trajectory-tracking errors for the heading

## IV. CONCLUSION

An obstacle avoidance algorithm using a discrete-time sliding-mode control and Sick LMS111 laser for wheeled mobile robots is presented in this paper. The effectiveness of this algorithm is proven by simulation results. The vehicle tracks a global trajectory until an obstacle is detected or the goal is reached. If an obstacle is detected a local trajectory is generated and followed until the obstacle is cleared and the initial trajectory is resumed until a new obstacle is detected or the goal is reached. Both trajectories are followed using the same discrete-time sliding mode controller. An increase in the trajectory-tracking errors is caused by switching from the global and local trajectories but the robot can still follow the trajectory with sufficient precision. The calculations required to process the laser data and generate a new trajectory cause delays when sending control inputs to the vehicle and the control inputs are no longer sent at the desired 100ms intervals.

## REFERENCES


[1] V.I. Utkin, "Sliding modes in optimization and control". New York: Springer-Verlag, 1992.

[2] V.I. Utkin, J. Shi, "Sliding mode control in electromechanical systems". London: Taylor&Francis, 1999.

[3] W. Gao, J. C. Hung, "Variable structure control on nonlinear systems: A new approach," IEEE Transactions on Industrial Electronics, vol. 40, pp. 45–55, 1993.

[4] W. Gao, Y. Wang and A. Homaifa, ""Discrete-time variable structure control systems," IEEE Transactions on Industrial Electronics, vol. 42, pp. 117–122, April 1995.

[5] R. Solea, A. Filipescu and G. Stamatescu, "Sliding-mode real-time mobile platform control in the presence of uncertainties," in Proc. 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference(CDC/CCC2009), Shanghai, pp. 7747-7752, December 2009.

[6] R. Solea, "Sliding mode control applied in trajectory-tracking of WMRs and autonomous vehicles," Ph.D. Thesis Dept. of Electrical and Computer Engineering, University of Coimbra, Portugal, 2009.

[7] R. Solea, A. Filipescu, U. Nunes, "Sliding-mode control for trajectory-tracking of a wheeled mobile robot in presence of uncertainties," in Proc. 7th Asian Control Conference(ASCC2009), Hong Kong, pp. 1701-1706, August 2009.

[8] R. Solea, A. Filipescu, S. Filipescu and B. Dumitrascu, "Sliding-mode controller for four-wheel-steering vehicle: Trajectory-tracking problem," in Proc. 8th World Congress on Inteligent Control and Automation(WCICA2010), Jinan, pp. 1185-1190, July 2010.

[9] R. Solea, A. Filipescu, V. Minzu and S. Filipescu, "Sliding-mode trajectory-tracking control for a four," in Proc. 8th IEEE International Conference on Control and Automation(ICCA2010), Xiamen, pp. 382-387, June 2010.

[10] K. Furuta and Y. Pan, "Discrete-time variable structure control," Lecture Notes in Control and Information Sciences, vol. 274, pp 57-81, Berlin: Springer, 2006.

[11] B. Bandyopadhyay and S. Janardhanan, "Discrete-time sliding mode control: A multirate output feedback approach," Lecture Notes in Control and Information Sciences, vol. 323, Berlin: Springer-Verlag, 2005.

[12] B. Dumitrascu and A. Filipescu, "Discrete-time sliding-mode controller for wheeled mobile robots," in Proc. 18th International Conference on Control Systems and Computer Science, vol.1, Bucharest, pp. 397-403, May 2011.

[13] B. Dumitrascu and A. Filipescu, "Sliding mode control of lateral motion for four driving-steering wheels autonomous vehicle," Annals of the University of Craiova, vol(7), pp. 20-25, 2010.

[14] B. Dumitrascu and A. Filipescu, A. Radaschin, A. Filipescu Jr., E. Minca, "Discrete-time sliding mode control of four driving/steering wheels mobile platform," in Proc. 8th Asian Control Conference (ASCC2011), Kaohiung, Taiwan, pp 771-776, May 2011.

[15] A. Filipescu, V. Minzu, B. Dumitrascu, A. Filipescu, "Trajectory-tracking and discrete time sliding-mode control of wheeled mobile robots," The 2011 IEEE International Conference on Information and Automation(ICIA2011), Shenzhen, pp. 27-32, June 2011.

[16] B. Dumitrascu, "Contributions to control, navigation and obstacle avoidance for mobile robots and autonomous vehicles," Ph.D. Thesis Dept. Department of Automation and Electrical Engineering, University Dunarea de Jos of Galati, Romania, 2012.

[17] J. Borenstein, Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," IEEE Journal of Robotics and Automation, vol. 7, pp 278-288, 1991.

[18] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," IEEE International Conference on Robotics and Automation, pp 500-505, March 1985.

[19] Susnea I, A. Filipescu, G. Vasiliu, G. Coman, A. Radaschin, "The buble rebound obstacle avoidance algorithm for mobile robots," in Proc. 8th International Conference on Control and Automation (ICCA2010) , Xiamen, pp. 540-545, June 2010.