Recent Innovations in Mechatronics (RIiM) Vol. 3. (2016). No. 1-2.
DOI: 10.17667/riim.2016.1-2/3.

Navigation of differential drive mobile robot on predefined, software designed path

Áron Papp
Technical Institute, Faculty of Engineering
University of Szeged
Szeged, Hungary
papparon@mk.u-szeged.hu

Dr. László Szilassy
Gamma Digital Kft.
Budapest, Hungary
szilassy.laszlo@gmammadigital.hu

József Sárosi PhD
Technical Institute, Faculty of Engineering
University of Szeged
Szeged, Hungary
sarosi@mk.u-szeged.hu

This paper will be presenting the process of mobile robot movement controlling, from the task of collecting sensor data until the problem of controlling data to the servo motor controllers. In details, the first part will show the mechanism of converting CAD data to routes, and the processing of the navigation data read from the sensors and calculated from former controlling commands. The second part will explain the processing of navigation data, the applying of the actual robot position and orientation on the predefined virtual path and the production of the controller's input variables. The Fuzzy controller and the rule base will be introduced in the third part.

Keywords — mobile robot, differential drive, robot navigation, laser navigation, Fuzzy control, path planning

I. INTRODUCTION

Automation of logistic processes has been made a considerable progress in the recent decades. One of the most frequent focuses of these developments can be linked to warehouse automation-particularly high bay warehousing, pallet transporting and the use of pneumatic artificial muscle actuators, especially alongside the sorting processes- as experienced while examining different researcher- and student-projects ([1], [2], [3], [4]).

This paper will focus on controlling a differential driven two-wheeled robot on a predefined path, which has the following prerequisites:

- Planning the paths, which the robot can drive on, and make the data ready to use by algorithms. In this case the paths will be designed in AutoCAD, and then converted to a directed, weighted graph.
- Collecting information about the robot's current position, and the relative angle of its velocity vector.
- Calculating input data for a Fuzzy controller using the current position and the path-data.
- Feeding the preconfigured controller with the calculated data, and reading out the return of the Fuzzy controller.

With special thanks to the following companies:

Gamma Digital Kft.
Sick Értékesítő és Szolgáltató Kft.
Technomed Orvosi Műszergyártó Kft.

The controller's outputs can be used to directly control the robot's drivetrain, in the current case the two wheels of the differential drive mechanism.

References [5], [6] and [7] have been used during the design of the above detailed control circle.

II. PATHPLANNING

A. Reading room's contour

After studying various methods of robot navigation ([8], [9]) for finding out the robot's actual position the decision was made on the Sick NAV-350 laser sensor- as in many other robotics and AGV projects so far ([10], [11]).

As a Light Detection and Ranging sensor it is capable to return the array of distances measured after a 360 degree vertical scan of the room. An example scan result can be seen on Fig. 1.

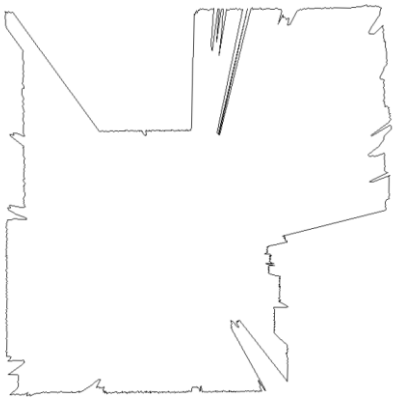

Fig. 1. The room's horizontal contour

B. Designing the allowed path's

The contour will be imported to an AutoCAD drawing, where the user can create the routes allowed for the robot to use.

The route consists of LINE and SPLINE objects, a special user-defined „STATION" object, and the permitted driving direction of the sections. The SPLINE objects are later converted to POLYLINE's using the AutoCAD's built in function for this conversion, as later it will be easier to handle a series of points, than a mixed set containing points and curves. Fig. 2 shows the result of contour scan extended with the above named path objects.

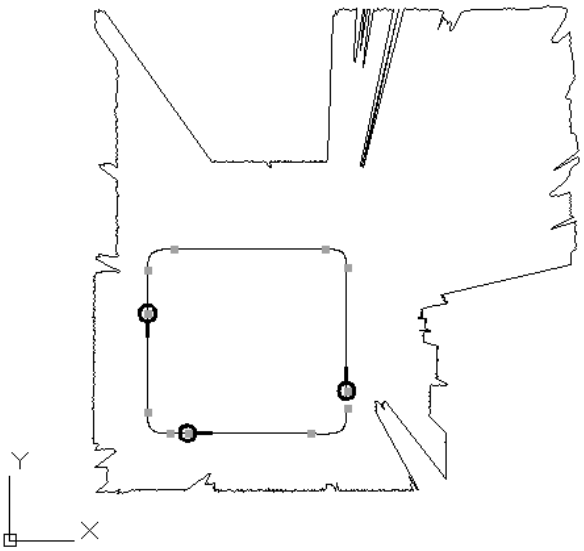

Fig. 2. The contour extended with route-objects

III. PROCESSING THE AUTOCAD MODEL

The model will be exported as a .dxf file, which will be processed by algorithms. In the current project an open-source dxf manipulator library – dxflib – has been used to parse the model.

The following steps have been implemented to perform the processing task:

- The first phase is to collect the sections from the file, by storing the lines' endpoints into an array.
- Second phase is to find adjacent sections, which can form a continuous path for the robot. Besides comparing the endpoints of the linear sections, the two adjacent section's relative angle must be 0 or π, as well.
- Third phase is to find and save the special "STATION" endpoints, which can be used as goal positions.
- Forth phase calculates the length of all the sections.
- The fifth phase extracts the permitted direction for each section.

At this stage a directed, weighted graph is available for searching the shortest path from the robot's current position to a desired place. As the map is stored using coordinates of a set's edges, the entire map can be kept in the memory consuming only few kilobytes (or megabytes for really complex cases).

IV. CALCULATING THE ROUTE TO THE GOAL

A. *Finding the nearest section to the robot*

It is elementary to superpose the robot at a permitted position over a section. The distance of a point to a line can be calculated by substituting the point's coordinates to the equation of the line, making its absolute value, than dividing it by the square root of the sum of the squares (1).

$$\frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} \tag{1}$$

Equation (1) can only be used for lines, but not for sections, since it is necessary to determine the place where the perpendicular is intersecting the line (which is containing the segment).

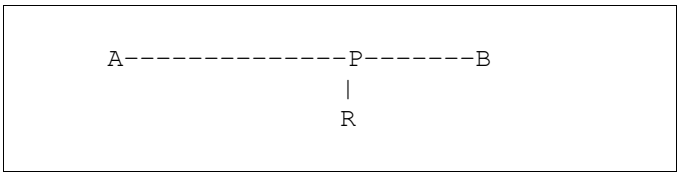

Fig. 3. Calculating the robot's distance to the segments

On the Fig. 3 point A and point B are representing the section endpoints, point R is the robot's current position, and point P is the place, where the perpendicular started from R have intersect with the section. Using the dot product of the vectors **AB** and **AR**, the placement of the intersection can be calculated. If the result is less than zero, the intersection is before point A, and if the result is greater than the length of **AB**, the intersection is after the B point.

Using a normalized form of the calculation explained above allows determining the coordinates of the point P, as shown in (2), (3) and (4).

$$r = \frac{\overline{AR} \cdot \overline{AB}}{\|\overline{AB}\|^2} \tag{2}$$

$$P_x = A_x + r(B_x - A_x) \tag{3}$$

$$P_y = A_y + r(B_y - A_y) \tag{4}$$

The value of „r" indicates the location of the point P along the AB section:

- r = 0 : P = A
- r = 1 : P = B
- r < 0 : P is on the backward extension of AB
- r > 1 : P is on the forward extension of AB

- 0 < r < 1 : P is interior to AB

To make the programming of the algorithm more efficient, the general formulas for *the dot product of the vectors* and *the calculation of the distance between points* are converted into2D form allowing use only basic mathematical operations, as shown by (5), (6) and (7).

$$L = \sqrt{(B_x - A_x)^2 + (B_y - A_y)^2} \tag{5}$$

$$r = \frac{(R_x - A_x)(B_x - A_x) + (R_y - A_y)(B_y - A_y)}{L^2} \tag{6}$$

$$AP = Lr \tag{7}$$

Finally, a new variable „s" will be introduced, to indicate the location of R along the line containing the section PR.

$$s = \frac{(A_y - R_y)(B_x - A_x) + (A_x - R_x)(B_y - A_y)}{L^2} \tag{8}$$

The sign of „s" indicates the side of the AB section, where point R belongs to:

- s < 0: R is left of AB
- s > 0: R is right of AB
- s = 0: R is on AB

After finding the value of „s" the distance to the path (PR) and the distance to the section endpoint (PB) can be calculated as shown by (9) and (10).

$$PR = sL \tag{9}$$

$$PB = L(1 - r) \tag{10}$$

Calculating the robot-section distances can be done using the algorithm presented above. Selecting the section with the lowest distance value is trivial, but it is still not sure, that the selected section will be the one to use as a start section. The initial section must be within a predefined distance from the robot, and the section's orientation must be matching to the robot's orientation. Moreover, if the orientation is matching, the direction must be permitted in the graph, as well.

B. Dividing the nearest section

At the point where the perpendicular posed from the robot to the nearest section has and intersection with the section line, the section is divided into two parts (Fig. 3, sections AP and PB). The intersection point's distance to the section endpoints will be calculated, and in the graph section AB will be substituted with section AP and PB, inheriting metadata, like allowed directions from the former section AB. The length of the new sections will be calculated, too.

C. Finding the shortest path

A graph searching algorithm can be used at this stage to determine the shortest path to the goal station. The widely known A-star algorithm was selected for this purpose, because of its advantageous features in the aspect of path finding and graph traversal, as proved in [12], [13].

V. CALCULATING THE ROBOT'S DISPLACEMENT

From the Sick navigation laser sensor the robot's position can be requested 3-times a second, which is not enough frequent information source to control the robot's center within a desired distance from the predefined route at a desired movement speed. To calculate the actual position data the robot's kinematics needs to be defined. Fig. 4 and Table 1are intended to illustrate and explain the kinematics of a 2-wheeled, differential driven robot.

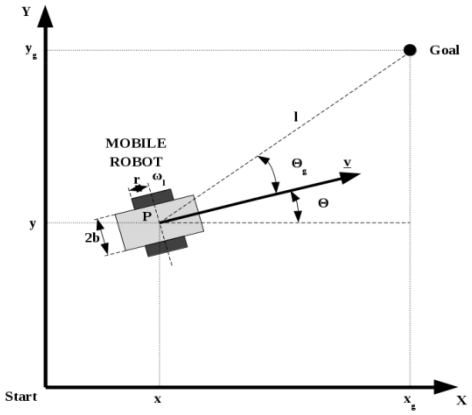

Fig. 4. Kinematics of a differential drive robot

TABLE 1.LEGEND FOR FIG. 4

Notation	Explanation
x, y	Robot's Current position
x_g, y_g	Goal position
Θ	Robot's current angle
Θ_g	Goal angle relative to robot's angle
ω_l, ω_r	Wheels' current angular speed
r	Wheel radius
$\underline{v}$	Robot's velocity vector
l	Robot - goal distance

The robot's actual position and orientation depending on the wheels' average angular speed since the last measurement point, and the elapsed time since the last measurement can be calculated using (11), (12) and (13).

$$\dot{x} = \frac{r\cos\theta\,(\omega_r + \omega_l)}{2} \tag{11}$$

$$\dot{y} = \frac{r\sin\theta\,(\omega_r + \omega_l)}{2} \tag{12}$$

$$\dot{\theta} = \frac{r(\omega_r - \omega_l)}{2b} \tag{13}$$

VI. CALCULATING THE CONTROLLER'S INPUTS

A wide range of Fuzzy based controlling mechanisms of differential driven mobile robots has been published so far - [14], [15], [16] and [17] were used as basis for the controller design.

This controller needs to be feed at least with two data: goal distance and goal angle -allowing the robot follow the route and drive, slow down or stop where it is necessary. The calculation of the inputs consists of the following steps:

1. As the path-finding algorithm returned the series of coordinates (i.e. the section endpoints) needed to touch by the robot, the coordinates are pushed into a FIFO stack.
2. Initially the first element is the nearest point to the robot.This point will be taken out of the stack, and made the first temporary goal, so*goal angle* and *goal distance* will be calculated for this point.
3. The controller is now feed with the calculated data until the next iteration.
4. If the robot is approaching the temporary goal, the algorithm pops out a new coordinate from the stack, and make it the new temporary goal.
5. Step 2 and Step 4 are repeated until the stack gets empty.

VII. DEFINITION OF THE FUZZY CONTROLLER

A. *Membership functions*

Fuzzy membership functions are used to label ranges of data into overlapping or non-overlapping sets. This regulation uses two input- and two output variables:

Input 1 : Goal angle (Right, Left)
Input 2 : Goal distance (Near)
Output 1 : Average angular speed (Stop, Drive)
Output 2 : Angular speed difference (Goleft, Goright)

The actual definitions of the variable's membership function areillustrated on Fig. 5-8.Fig. 5-8 are captured from the software QtFuzzyLite4 -a graphical interface for the fuzzylite open-source Fuzzy Logic Control library, written in C++. Reference [18] has been used for verifying the underlying design and operation of the "fuzzylite" library.

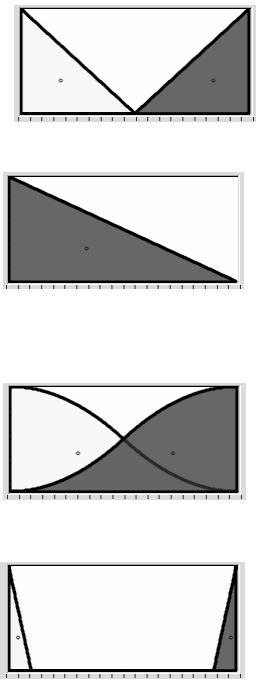

Fig. 5. Membership functions of "Goal angle" input variable

Fig. 6. Membership function of "Goal distance" input variable

Fig. 7. Membership functions of "Average angular speed" output variable

Fig. 8. Membership functions of "Angular speed difference" output variable

B. *Fuzzy rule set*

The six rules are intended to solve the following problems:

- Make the robot turn to the right direction, if the „Goal angle" is „Right" or „Left" (i.e. not zero).
- Make the robot drive, if the „Goal distance" is not „Near", and stop, when the distance is 100% „Near".
- Make the robot decrease its speed, if it performs turning movement.

The rules are summarized in Fig. 9.

1. IF **GOALANGLE** IS **LEFT**
   THEN **ANGULARSPEEDDIFFERENCE** IS **GOLEFT**
2. IF **GOALANGLE** IS **RIGHT**
   THEN **ANGULARSPEEDDIFFERENCE** IS **GORIGHT**
3. IF **GOALDISTANCE** IS **NEAR**
   THEN **ANGULARSPEED** IS **STOP**
4. IF **GOALDISTANCE** IS **NOT NEAR**
   THEN **AVGANGULARSPEED** IS **DRIVE**
5. IF **GOALANGLE** IS **RIGHT** AND **GOALDISTANCE** IS **NOT NEAR**
   THEN **AVGANGULARSPEED** IS **STOP**
6. IF **GOALANGLE** IS **LEFT** AND **GOALDISTANCE** IS **NOT NEAR**
   THEN **AVGANGULARSPEED** IS **STOP**

Fig. 9. The Fuzzy rule set

VIII. CONCLUSION

This paper presented a solution for driving 2-wheeled differential drive mobile robots on a software designed path. The industrial grade LIDAR sensor (through its safety certifications) makes the solution embeddable into real-world applications, like automation of forklifts and other industrial trucks. For testing purposes, an individual mobile robot was also manufactured. Running the software implementation of the detailed algorithms on the test-robot is showing promising results, although plans are available for further improvements of positioning accuracy and for supporting other types of drive trains.

REFERENCES

[1] J. Sárosi, A. Gergely and F. Tölgyi, "Student Project on Pneumatically Driven Muscle-like Actuators", 3rd International Conference and Workshop Mechatronics in Practice and Education - MECHEDU 2015, Subotica, Serbia, 14-15 May, 2015, pp. 153-157.

[2] I. Fürstner and L. Gogolák, "Modification of Technical Documentation Prepared by Students for Building a Product Prototype", 3rd International Conference and Workshop Mechatronics in Practice and Education - MECHEDU 2015, Subotica, Serbia, 14-15 May, 2015, pp. 60-65.

[3] J. Sárosi, "Elimination of the Hysteresis Effect of PAM Actuator: Modelling and Experimental Studies", Technical Gazette, vol. 22, no. 6, 2015, pp. 1489-1494.

[4] J. Sárosi, I. Bíró, J. Németh and L. Cveticanin, "Dynamic Modelling of a Pneumatic Muscle Actuator with Two-direction Motion", Mechanism and Machine Theory, vol. 85, 2015, pp. 25-34.

[5] Gy. Mester, "Adaptive Force and Position Control of Rigid Link Flexible- Joint Scara Robots", International Conference on Industrial Electronics, Control and Instrumentation, 20th Annual Conference of the IEEE Industrial Electronics Society, IECON'94, Bologna, Italy, 5-9 September 1994, pp. 1639-1644.

[6] Gy. Mester, "Intelligent Mobile Robot Controller Design", 10th Intelligent Engineering Systems, INES 2006, London, United Kingdom, 26-28 June, 2006, pp. 282-286.

[7] Gy. Mester, "Introduction to Control of Mobile Robots", YUINFO'2006, Kopaonik, Serbia and Montenegro, 6-10 March, 2006, pp. 1-4.

[8] J. Simon and M. Goran, "Navigation of Mobile Robots Using WSN's RSSI Parameter and Potential Field Method", Acta Polytechnica Hungarica, Journal of Applied Sciences, vol.10, no.4, 2013, pp. 107-118.

[9] G. Kovács and G. Péter, "Marker Based Visual Navigation of Mobile Robots on Hybrid Embedded Platform", Workshop on the Advances of Information Technology, WAIT 2015, Budapest, Hungary, 19 May, 2015, pp. 118-125.

[10] M. Pinto, H. Sobreira, A. P. Moreira, H. Mendonça and A. Matos, "Self-localisation of Indoor Mobile Robots Using Multi-hypotheses and a Matching Algorithm", Mechatronics, vol. 23, 2013, pp. 727-737.

[11] A. B. Beck, "Situation Assessment for Mobile Robots", PhD thesis, Technical University of Denmark, 2012, 187 p.

[12] W. Y. Loong , L. Z. Long and L. C. Hun, "A Star Path Following Mobile Robot", 4th International Conference OnMechatronics (ICOM), Kuala Lumpur, 17-19 May, 2011, pp. 1-7.

[13] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek and T. Fico, "Path Planning with Modified a Star Algorithm for a Mobile Robot", Procedia Engineering, vol. 96, 2014, pp. 59-69.

[14] R. Rashid, I. Elamvazuthi, M. Begam and M. Arrofiq, "Differential Drive Wheeled Mobile Robot (WMR) Control Using Fuzzy Logic Techniques", Fourth Asia International Conference onMathematical/Analytical Modelling and Computer Simulation (AMS), Kota Kinabalu, Malaysia, 26-28 May, 2010, pp. 51-55.

[15] A. M. Almeshal, M. R. Alenezi and M. Moaz, "Intelligent Path Tracking Hybrid Fuzzy Controller for a Unicycle-Type Differential Drive Robot", International Journal of Computer, Electrical, Automation, Control and Information Engineering,vol:9, no:4, 2015, pp. 901-904.

[16] R. Rashid, I. Elamvazuthi, M. Begam and M. Arrofiq, "Fuzzy-based Navigation and Control of a Non-Holonomic Mobile Robot", Journal of Computing, vol. 2, no. 3, 2010, pp. 130-137.

[17] V. M. Peri, "Fuzzy Logic Controller for an Autonomous Mobile Robot", Jawaharlal Nehru Technological University, India, 2002, 161 p.

[18] J. Rada-Vilelahttp,"A Fuzzy Logic Control Library in C++" www.fuzzylite.com/download/fuzzylite-paper-3.1.pdf