Distributed Control System for Mobile Robots With Differential Drive

Ján Jadlovský, Michal Kopčík
Department of Cybernetics and Artificial Intelligence,
Faculty of Electrical Engineering and Informatics, Technical University of Košice,
Košice, Slovak Republic
jan.jadlovsky@tuke.sk, michal.kopcik@tuke.sk

***Abstract*— This article describes universal control system mainly for mobile robots with differential drive. The first section is introduction into the problem of mobile robotics and contains requirements on control system. Second section describes individual parts of designed control system focusing on motion control and in the last section is devoted to testing of designed control system on real mobile robot.**

Keywords—mobile robot; motion control; differential drive; distributed control system

I. INTRODUCTION

Motion control is elementary task that every kind of robot must be able to fulfill. Our motion control system can be used on a several types of mobile robots; especially it is designed for mobile robots with differential wheeled or tracked chassis. This type of chassis is nowadays used in many types of mobile robots in many different fields such as service robotics, education, household appliances, industry, military or robotics competitions. Between most famous representatives of such a type of robots are: robotic vacuum cleaner Roomba, mobile robot for education Khepera or warehouse mobile robot Kiva (Fig. 1).

Fig. 1. Example of commercially available mobile robots with differential drive (Khepera III, Kiva, Zumo, Roomba).

The main motivation for this work was to create a universal control system for various mobile robots emerging at our Department of Cybernetics and Artificial Intelligence at Technical University of Košice.

Requirements for this control system were simple adaptability to various types of chassis, electronics and processing units. Next requirement was, that all the functions for control of the robot including main parameters and state values should be accessible from user program and selected communication interface. Last but not least requirement was that the control system shall be modular in the way, that change in one module won't affect another module.

II. MOBILE ROBOT CONTROL SYSTEM

It is all the software that the mobile robot uses to control peripherals, communicate with other devices and control the motion of the robot. In our case the control system consists of components, namely:

- Hardware abstraction layer.
- User program / ROS / communication interface.
- Localization.
- Motion control.

Block diagram of whole control system is shown in Fig. 2, where motion control is in the middle of figure.

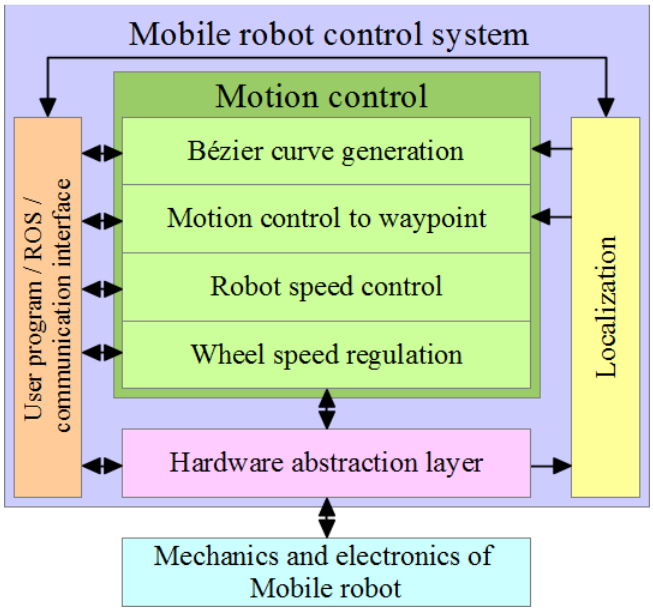

Fig. 2. Block diagram of mobile robot control system with the motion control in the middle of picture.

A. Hardware abstraction layer

This level serves to bridge the hardware and software of the robot such as motion control system, user program and other software modules. It includes all of the functions to control given peripherals of the robot such as motors, encoders, sensors, LEDs, buttons, etc.

B. User program / ROS / communication interface

The mobile robot can be controlled from user program using functions or using text messages. Also mobile robot can be controlled using any communication interface that allows communication using text messages such as USART, I2C, SPI, etc.

Received message is separated into individual arguments and forwarded to corresponding level, where it is executed. The response is sent back to sender to confirm, that the control system recognized and executed command. When there is received command to read or write one or more arguments, first they are written, then they are faulted back and returned to sender.

Communication interface also has system for subscribing of parameters that are periodically collected and sent to user program or trough communication interface. Using communication interface, mobile robot can be implemented into ROS (Robot Operating System) architecture, but using ROS system needs for its operation computer with operating system such as Linux, Android or OS X.

C. Localization

For feedback motion control it is necessary to know actual position and rotation (further referred only as position) of the mobile robot. The simplest way how to calculate relative position of the robot is using odometry, where the position is calculated based on speed of the wheels and kinematic model of the robot. Thus obtained position is however burdened by error that grows with traveled distance. To suppress or relieve this error, mobile robot must have additional sensors of absolute or relative position (external cameras, GPS, gyroscope, accelerometer, compass, etc.) or sensors of external environment (infrared, laser, ultrasound distance sensors or onboard cameras), which with the simultaneous localization and mapping can refine obtained position.

The data from multiple sensors can be merged fused together using simple or extended Kalman filter to obtain more robust and precise position of mobile robot. More information about localization using multisensory data fusion can be found in [1] and [2].

Also to get absolute position of robot, GPS or beacons can be used. In our case we use external camera to detect robot and get its absolute position [3].

D. Motion Montrol

This block can be divided into several levels, where output from higher level is the input to the lower one. Each of the levels is realized by separate program module and executed on the background of user program. Each level is accessible from user program and communication interface, alternatively from ROS system

1) Wheel speed regulation.

This is the lowest motion control level of mobile robots and is the most influenced by used hardware components like motors, motor drivers and rotary encoders. Inputs to this level are desired linear or angular velocities of the wheels w_{vx} and measured velocities v_{mx} ($x=r$ - right wheel, $x=l$ - left wheel). The outputs from this level are control actions to the motors u_x. Block diagram of velocity regulator is shown in Fig. 3.

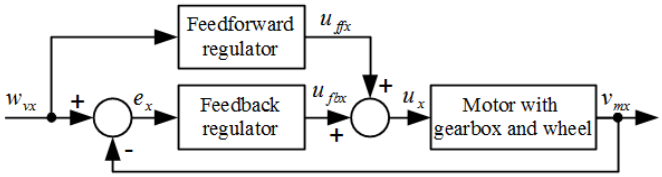

Fig. 3. Control structure with feedforward wheel speed regulation.

Control action that is applied to the motor consists of two parts u_{ffx} calculated by feedforward regulator and u_{fbx} calculated by feedback regulator. Feedback regulator can be almost any type of regulator, for example PI, PID or PSD regulator. Feedforward regulator is usually the function of steady state values of wheel velocities based on constant control action values. Individual values of this function can be also obtained as steady values of control action of feedback regulator at constant speed. Sample feedforward function measured on mobile robot ALFRED is shown in Fig. 4 [4].

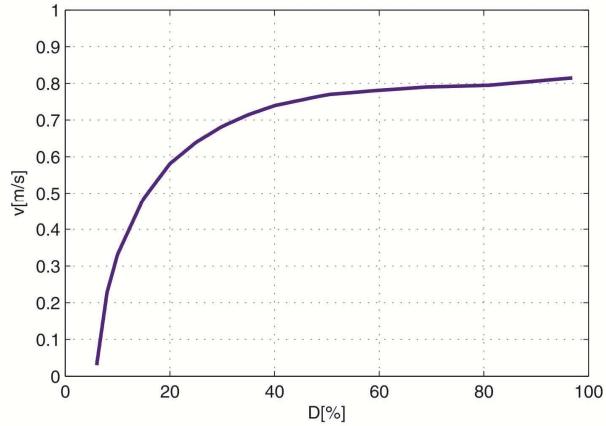

Fig. 4. Measured feedforward regulator function for mobile robot ALFRED.

In the case when the values of feedforward function are unknown or unobtainable, it is possible to use stand-alone feedback regulator, but the quality of regulation may be reduced.

Described regulation of wheel velocities can be used for DC motors. When using stepper motors or servos, this level is substituted by velocity controllers.

2) Robot speed control

This level in motion control of mobile robot serves for calculation of linear and angular velocity of mobile robot v and ω from circumferential speed (later referred as linear speed) of the wheels v_r and v_l and vice versa using follow equations.

$$v_r = v + \frac{(\omega * l)}{2} \quad (1)$$

$$v_l = v - \frac{(\omega * l)}{2} \quad (2)$$

$$\omega = \frac{v_r - v_l}{l} \quad (3)$$

$$v = \frac{v_r + v_l}{2} \quad (4)$$

Kinematic model of the mobile robot is also used calculation of relative position in the space based on measured linear speed of the wheels using equations (5) – (7) [5].

$$\varphi(k) = \varphi(k-1) + \omega(k) \quad (5)$$

$$X(k) = Y(k-1) + cos\left(\varphi(k) - \frac{\omega(k)}{2}\right) * v(k) \quad (6)$$

$$Y(k) = Y(k-1) + sin\left(\varphi(k) - \frac{\omega(k)}{2}\right) * v(k) \quad (7)$$

3) Motion control to waypoint

It is feedback regulation of position of mobile robot to desired point in space (waypoint). This motion control consists of two parts namely motion control along the line and rotation about an axis.

Input to this level are waypoints, i.e. desired points of motion along the robot will move. These waypoints hold data about desired position and rotation in which the mobile robot should be after reaching the waypoint and the information of type of motion (motion forward/back or rotation right/left). After reaching waypoint its position becomes baseline position of next waypoint.

Another approach to motion control can be found in [5].

a) Motion control along the line

Motion control uses the standard form of a line given by equation (8) to maintain direction and distance from the line of motion.

$$L_a x + L_b Y + L_c = 0 \quad (8)$$

Individual parameters of line L are calculated using equations (9) – (11) where *WP1* is starting waypoint and *WP2* is reference waypoint.

$$L_a = WP1_Y - WP2_Y \quad (9)$$

$$L_b = WP1_X - WP2_X EQ \quad (10)$$

$$L_c = (WP1_Y * WP2_Y) - (WP2_X WP1_Y) \quad (11)$$

From the parameters of the line can be calculated its angle L_α using (12).

$$L_\alpha = \text{atan}\,(L_a / L_b) \quad (12)$$

From the parameters of the line and actual position of the robot, the distance from the line s and angle β can be calculated using equations (13) and (14).

$$s(k) = \frac{L_a * X(k) - L_b * Y(k) + L_c}{\sqrt{{L_a}^2 + {L_b}^2}} \quad (13)$$

$$\beta(k) = L_\alpha - \varphi(k) \quad (14)$$

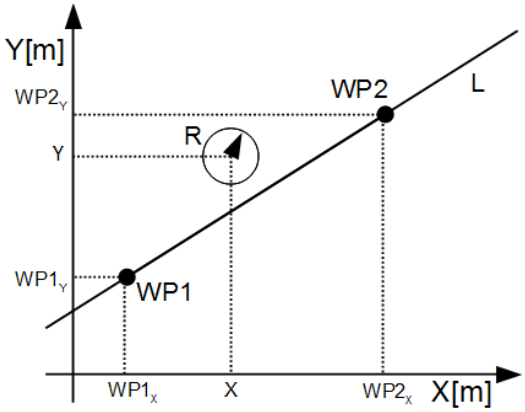

Fig. 5. Illustration of waypoints WP1, WP2, mobile robot R and line L.

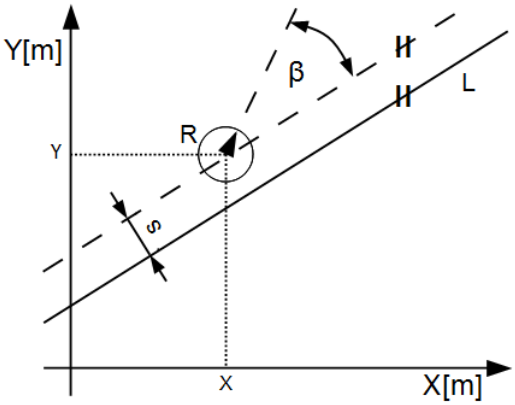

Fig. 6. Illustration of distance s and angle β.

From distance *s(k)*, angle *β(k)* and actual velocity of the robot *v(k)* it is possible to calculate desired angular velocity $w_\omega(k)$ using equation (15).

$$w_\omega(k) = \left(\beta(k) * k_1 - cos\left(\beta(k)\right) * s(k) * (1 - k_1)\right) * k_2 * w_v(k) \quad (15)$$

Parameter *k1* gives the ratio between individual parts of regulator and parameter *k2* sets the influence of desired linear velocity w_v on the resulting desired angular velocity w_ω.

Desired linear velocity is governed by uniform acceleration/deceleration with limited maximal velocity.

If there are multiple waypoints with the same type of motion in a row, motion control system ensure their concentration and fluent switching between reference waypoints, thus ensuring fluent motion from the starting waypoint to the end waypoint in this row. Alternatively, smoothing of the trajectory can be used to round sharp edges of trajectory given by waypoints [6].

The waypoint is considered accomplished, when the relative Y position of the waypoint according to robot position is less than zero.

b) Rotation along the axis

For this type of motion apply, that the desired linear velocity is zero and desired angular velocity is governed by uniform angular acceleration/deceleration with limited maximal angular velocity.

4) Bézier curve generation

This level is used for generating set of waypoints based on type and parameters of the curve. In our case we implemented in this layer function for generating waypoints based on Bézier curve. As a parameters for Bézier curve generation we used actual linear velocity and position of the robot (*v(k)*, *X(k)*, *Y(k)* and *φ(k)*) and desired end position and speed (X_e, Y_e, v_e a φ_e). These input parameters are converted into four points P0 – P3 using follow equations.

$$P_0 = \begin{bmatrix} X(k) \\ Y(k) \end{bmatrix}, P_3 = \begin{bmatrix} X_e \\ Y_e \end{bmatrix} \tag{16}$$

$$P_1 = \begin{bmatrix} X(k) + cos(\varphi(k)) * v(k) * k_b \\ Y(k) + sin(\varphi(k)) * v(k) * k_b \end{bmatrix} \tag{17}$$

$$P_2 = \begin{bmatrix} X_e - cos(\varphi_e) * v_e * k_b \\ Y_e - sin(\varphi_e) * v_e * k_b \end{bmatrix} \tag{18}$$

Points *P0 – P3* are then converted into set of waypoints using equation (19), where parameter *i* is within the interval <0, 1>, while *i=0* represents starting waypoint (actual position) and *i=1* is end waypoint.

$$WP_n = m_0(i) * P_0 + m_1(i) * P_1 + m_2(i) * P_2 + m_3(i) * P_3 \tag{19}$$

Parameters m_0 - m_3 can be calculated using equations (20)-(23).

$$m_0(i) = (1 - i)^3 \tag{20}$$

$$m_1(i) = 3i(1 - i)^2 \tag{21}$$

$$m_2(i) = 3i^2(1 - i) \tag{22}$$

$$m_3(i) = 3i^3 \tag{23}$$

The direction of motion along the whole Bézier curve is determined by the linear speed at the time the command to generate this curve arrived. Within generated waypoints are no waypoints for turning along the axis, so the motion is set of waypoints for motion along the line, which gives the impression of fluent motion along the curve.

In this layer can be also implemented other method for generating curves such as B-spline method [7].

III. Testing of Designed Motion Control on Real Mobile Robot

Designed control system for mobile robots was implemented and tested on mobile robots for robotic soccer of category MiroSot, which are shown on Fig. 7.

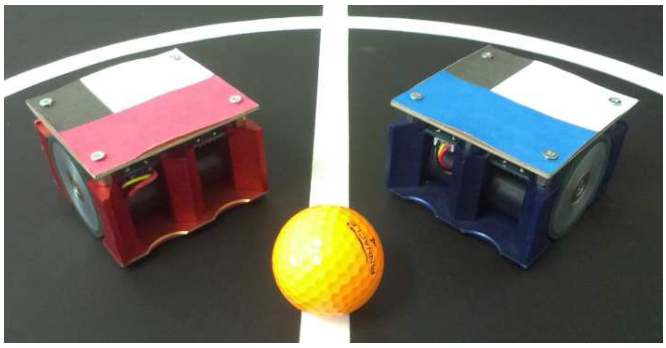

Fig. 7. Mobile robots for robotic soccer of category MiroSot.

A. Electronics

Electronics of this mobile robot consists of one board, which includes microcontroller (STM32F103), motor driver with the possibility of sensing currents that flows trough coils of motors, Bluetooth communication interface for remote control of the robot and expansion connector for external sensors (gyroscope accelerometer, compass) or another microcontroller or microcomputer (for example Raspberry Pi). Whole electronics is supplied from two cell Li-Pol battery with the capacity of 850mAh and approximate time of operation about 60 minutes. The electronics is designed in the way that it can be used in mobile robots with similar hardware.

B. Mechanics

The chassis of robot is made of aluminium metal and consists of five parts connected by screws. As actuators it has two Faulhaber DC-micromotors featuring integrated incremental encoders with resolution 4096 impulses per revolution. Transfer of torque to wheel provides gear which is integrated into wheel of the robot. Given configuration of wheel, gears and encoder gives resolution of 724 impulses per millimeter of linear motion.

C. Results of motion control

For obtaining of the position is used odometry. The data about real position of the robot are transferred to computer via UART using Bluetooth module.

Testing of motion control was made on several trajectories, two of them were chosen for this article. First of them is motion control using absolute coordinates and illustrates avoiding an obstacle. Motion along this trajectory consists of forward motion from coordinates [0, 0] to point [0, 500]. The generated waypoints and real trajectory is shown in Fig. 8.

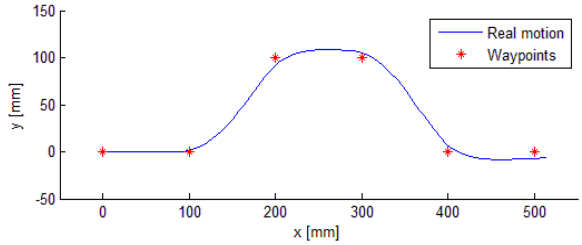

Fig. 8. Motion along defined waypoints.

From the Fig. 8 we can see that mobile robot did not pass trough waypoints precisely, but he turned before waypoint to perform rounded motion. This is done by gradually switching reference waypoint. At third waypoint mobile robot turned after waypoint, because he didn't have the room to fully stabilize trajectory along the line before this waypoint.

The second testing trajectory is Bézier curve from point [0, 0] with zero angle and zero speed to point [500, 500] with zero angle and 500 mm/s speed at the end of trajectory. The generated waypoints and real trajectory is shown in Fig. 9.

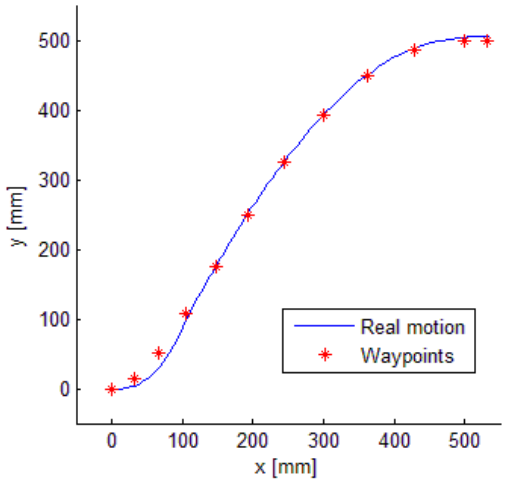

Fig. 9. Motion along Bézier curve.

This test show, that at the begging and at the end of trajectory, the real position compared to reference trajectory was a little off. This could be caused by small spacings between individual waypoints or because the control of desired angular velocity depends on actual linear speed.

IV. Conclusion

This paper presents a control system for differential wheeled mobile robots with motion control that can be separated into several layers beginning with wheel speed regulation trough robot speed control, motion control to waypoint and ending with Bézier curve generation. The results from tests on real mobile robot show the behavior of mobile robot motion using proposed control system. This control system can be used on other mobile robots with similar construction thanks to the general representation of proposed control.

In the future we plan to expand the Hardware abstraction layer to cower additional sensors such as gyroscope, accelerometer and compass to enhance obtaining of position and minimize the error caused by ruggedness of the surface and slippage between wheels (tracks) and surface. Also we want to implement B-spline curve generation.

Acknowledgment

This work has been supported by the Research and development Operational Program for project: University Science Park Technicom for innovative applications with knowledge technology support, ITMS code 26220220182, co-financed by the ERDF (80%) and by grant KEGA - 001TUKE-4/2015 (20%).

References

[1] M. Kopčík, "Multisensor data fusion for differential wheeled mobile robots," Proc. of the 14th Scientific Conference of Young Researchers – SCYR 2014, Herľany, Slovakia, pp. 223-226, ISBN: 978-80-553-1714-4.

[2] M. Kopčík, "Diagnostics of Sensors and Actuators Within Distributed Control System," Proc. of the 15th Scientific Conference of Young Researchers – SCYR 2015, Herľany, Slovakia, pp. 214-215, ISBN: 978-80-553-2130-1.

[3] J. Jadlovský, M.Varga and M. Kopčík, "Image Processing for Localization of Mobile Robots," in Electrical Engineering and - Informatics 6, Košice, 2015, pp. 547-557, ISBN: 978-80-553-2178-3.

[4] J. Jadlovský and and M. Kopčík, "Basic Motion Control of Differential-Wheeled Mobile Robot ALFRED," in Advances in Intelligent Systems and Computing, Switzerland, 2014 Vol. 316, pp. 73-80, ISSN: 2194-5357.

[5] T.Petrinić and I.Petrović, "Control of nonholonomial mobile robots with differential drives," In Int Proc. 33rd International Convention of Information and Communication Technology, Electronics and Micro-electronics – MIPRO, Opatija, 2010, pp. 108-113.

[6] S. Fleury, P. Soueres, J. P. Laumond and R.Chatila, "Primitives for smoothing mobile robot trajectories," In IEEE transactions on robotics and automation 11, 1995, pp. 441-448.

[7] K. Komoriya and K. Tanie, "Trajectory Design and Control of a Wheeled-type Mobile Robot Using B-spline Curve," Int. Workshop on Intelligent Robots and Systems, pp. 398-405, 1989.