2014 14th International Conference on Control, Automation and Systems (ICCAS 2014)
Oct. 22-25, 2014 in KINTEX, Gyeonggi-do, Korea

Wheel Angular Velocity Stabilization using Rough Encoder Data

Margarita Sotnikova*, Evgeny Veremey and Natalia Zhabko

Department of Computer Applications and Systems, Saint-Petersburg State University,
Saint-Petersburg, Russia (m.sotnikova@spbu.ru) * Corresponding author

Abstract: The paper is devoted to the problem of wheel angular velocity stabilization using encoder data. The differential drive robot with four wheels is used as a mobile robot. The left wheels and right wheels are controlled separately. This allows robot to move forward or backward if we apply equal control signal for both sides' wheels, or to turn left or right if the control for left and right wheels is different. It is well-known that for differential drive robots the robot wheels angular velocity is slightly different even if the same control input is applied for left and right wheels. This lead to the arc trajectories of the robot instead of straight lines. To overcome this problem it's necessary to use some internal low-level controller for wheels angular velocity stabilization. In order to stabilize angular velocity we need estimation of its current value, which can be obtained using encoder data. The algorithm for angular velocity estimation and stabilization using rough encoder data is proposed. The simulation study and practical real-time implementation for a particular differential drive robot are presented using Matlab and Arduino environment.

Keywords: differential drive robot, wheel angular velocity, encoder data, stabilization

1. INTRODUCTION

The autonomous mobile robotics represents the area of intensive research activity at the current moment. There are a lot of control problem arises here. For example, dynamic positioning, trajectory tracking, path following, visual feedback control, etc. A lot of mobile robots are under actuated, that should be taken into account during control design procedure.

In this paper we consider a particular class of wheeled mobile robots – differential drive robots with four wheels [4,5]. The left wheels are rotated with the same angular velocity and the right wheels are also have the same angular velocity, which can differ from the one for left wheels. The robot motion is determined by two control inputs, which are the voltages applied to motors of the left wheels and right wheels correspondently. If the both control inputs are equal then left and right wheels should rotate with the same velocity. If the control input value for left wheels is greater than one for right wheels then left wheels should rotate faster than right wheels and robot moves on arc of a circle. Thus, such two control inputs allows robot to move forward or backward on the straight line, rotate left or right on arc of a circle and even rotate around itself.

Differential drive wheeled mobile robots are under actuated since they have three degrees of freedom and two control inputs. This leads to a restriction of possible state space trajectories that can be realized. The high-level control laws mentioned above, such as dynamic positioning or trajectory tracking control laws, use linear and angular velocity of the robot as a control input. Clearly, the left and right wheels angular velocities are directly derived from the linear and angular velocities of the robot. So, the implementation of the particular control algorithm requires the exact tracking of the corresponding given wheels angular velocities.

The general problem of the differential drive robots is how to provide given wheels angular velocity. This problem is very important for implementing high level control algorithms. For example, if we apply the same input voltage for left and right wheels then the robot should move straightforward, but in practice the robot move along the arc of a circle. This is due to friction and may be some other disturbances that influence wheels velocity. This paper is devoted to low-level controller design for wheels angular velocity stabilization. Actually, this internal feedback control loop provides given angular velocity of the wheels and thus allows to design general control algorithms taking robot linear and angular velocities as a control inputs.

The main problem appeared here is the estimation of the current wheel angular velocity. Such estimation can be obtained using encoder data [1]. Some encoders have a small number of pulses per revolution (ppr) and as a consequence provide rough data for angular velocity computation. In this paper we propose algorithm for angular velocity estimation using rough encoder's data. This estimation is used in the internal feedback control loop in order to provide given velocity. As a result, digital algorithms for wheels velocity estimation and control input computation are proposed. These algorithms can be implemented in real-time for a particular mobile robot.

The paper is organized as follows. Firstly, the mathematical problem formulation is given. Secondly, the problem of wheel angular velocity estimation using encoder data is considered. In the third part stabilizing controller synthesis is performed in order to provide given angular velocity of the wheels. The last section is devoted to practical real-time implementation of the proposed algorithms and simulation study examples using Matlab environment.

2. PROBLEM FORMULATION

Let consider the following mathematical model of wheel rotations:

$$\dot{\gamma} = k_\gamma p, \tag{1}$$

where γ – turning angle of the wheel, p is a voltage applied to the wheel motor and k_γ is a constant positive value, which is depend on the characteristics of the particular mobile robot. We will suppose that the pulse-width-modulation (PWM) technique is used to control voltage applied to the wheels motors. In this case the control input p is a positive integer number from the range [0, 255]. Maximum value of p correspond to a highest wheel speed. So, any continues control input $\tilde{p}$ should be mapped to the mentioned range. It should be noted that the minimal number of p, such that robot can move, usually greater than zero.

The value k_γ can be estimated for a particular mobile robot experimentally. Let denote R as a wheel radius. It's evident that

$$\dot{\gamma} = k_\gamma p = \omega_w = v_w / R,$$

where ω_w and v_w are angular and linear velocities of the wheel correspondently. If $v_w^{\max}$ is a maximal angular velocity corresponding to maximal applied voltage $p_{\max}$ then we have

$$\omega_w^{\max} = \frac{v_w^{\max}}{R} = k_\gamma p_{\max}. \tag{2}$$

From Eq. (2) the estimation for k_γ is derived:

$$k_\gamma \approx \frac{v_w^{\max}}{R p_{\max}}.$$

Let now consider the measurement model, representing the information obtained from the encoders. We suppose that differential drive robot has four wheels. Left pair and right pair of wheels are rotated with the same speed. The robot has two encoders. First encoder is attached to the front left wheel, second encoder – to the front right wheel. The main characteristic of the encoder is the number N of pulses per revolution (PPR) [1]. The PPR value can vary in the broad range. For cheap encoders this value can be about 20 pulses. More comprehensive encoders have greater than 1000 pulses per wheel revolution. We will suppose that encoder has a PPR number not more than 100.

The encoder allows to acquire data with a fixed sample time Ts. Let $k(t)$ be the number of pulses during Ts seconds received at the current moment t, $\gamma(t)$ and $\gamma(t - Ts)$ are angle of wheel rotation at the time instants t and $t - Ts$ correspondently. Hence, we get following measurement model:

$$k(t) = \left[\frac{\gamma(t) - \gamma(t - Ts)}{2\pi} N \right], \tag{3}$$

where braces [] denote integer part. Together Eqs. (1) and (3) constitute mathematical model of wheel rotations and encoder measurements.

In this paper we consider two problems related to this model. First problem is the estimation of wheel velocity using encoder data at each sample instant t with a fixed time step Ts. It's supposed that encoder is quite rough as mentioned above. Second problem is wheel angular velocity stabilization using obtained velocity estimation. The resulting feedback controller must provide given angular velocity of the wheel.

3. WHEEL ANGULAR VELOCITY ESTIMATION USING ENCODER

3.1 Velocity estimation using encoder

The simplest wheel angular velocity estimation using encoder data can be derived as a ratio of rotation angle to the time of rotation, that is

$$\hat{\omega}_w(t) = \frac{2\pi \cdot k(t)}{N \cdot Ts}, \tag{4}$$

where $\hat{\omega}_w$ is a velocity estimation. This formula represents a well-known method M for speed measurement [3]. But Eq. (4) gives appropriate results only for encoders with large PPR value, where number of pulses $k(t)$ per sample period Ts varies slightly at the constant speed of mobile robot motion. In a case of encoder with small PPR, Eq. (4) gives velocity estimations, which essentially vary from one sample time to another when robot moves with the same speed. As a result, Eq. (4) is not reliable approach for wheel velocity estimation with rough encoders.

Instead of Eq. (4) let consider another approach, which is based on the mean wheel angular velocity over some time period. Let denote T as a time period in seconds over which mean angular velocity of the wheel is computed. It can be noted that T is convenient to choose as a multiply of sample period Ts, that is $T = s \cdot Ts$, where $s > 0$ is an integer number. Then the mean value of wheel angular velocity estimation over period T is given by

$$\tilde{\omega}_w(t) = \frac{1}{T} \int_{t-T}^{t} \hat{\omega}_w(\tau) d\tau = \frac{2\pi}{T \cdot N \cdot Ts} \int_{t-T}^{t} k(\tau) d\tau. \tag{5}$$

Here $\tilde{\omega}_w(t)$ is the averaged estimation. So, the Eq. (5) allows to estimate wheel angular velocity using mean value of $k(t)$ over time period T. From Eq. (5) we can obtain estimation for wheel linear velocity:

$$\tilde{v}_w(t) = \frac{2\pi R}{T \cdot N \cdot Ts} \int_{t-T}^{t} k(\tau) d\tau. \tag{6}$$

Let us transform Eq. (5) to the more suitable form from real-time implementation point of view. Let denote the integral value by

$$I(t) = \int_{0}^{t} k(\tau) d\tau. \tag{7}$$

Taking into account Eqs. (6) and (7) we get

$$\tilde{v}_w(t) = \frac{2\pi R}{T \cdot N \cdot Ts} \left(I(t) - I(t - T) \right). \tag{8}$$

It can be noticed from Eq. (8) that in order to compute velocity estimation it is necessary to store in memory values of function $I(t)$ over time interval $[t-T,t]$.

Let perform simple analysis of velocity estimation given by Eq. (8). Suppose that total number of encoder pulses over the time period T has changed by ± 1 pulse. This leads to following difference in estimations

$$\Delta\widetilde{v}_w = \frac{2\pi R}{T \cdot N}. \tag{9}$$

So, if PPR number N is large then this difference is small. In other case, the greater period T we choose the smaller is difference $\Delta\widetilde{v}_w$. But, we must keep in mind that the large values of T results in accumulation of encoders data over long time interval. As a result, if encoders' data is changed we can fix velocity changes with a time delay compared with T. Thus, there is a trade-off between the variance of estimation $\Delta\widetilde{v}_w$ and fast velocity tracking.

It can be noted also that the sample time Ts is not influence variance $\Delta\widetilde{v}_w$. So, this parameter can be selected from other practical issues such as feedback control loop requirements.

The relative error of velocity estimation is equal to

$$e_r = \frac{\Delta\widetilde{v}_w}{\widetilde{v}_w} = \frac{Ts}{I(t)-I(t-T)} = \frac{Ts}{\int_{t-T}^{t} k(\tau)d\tau} = \\ = \frac{1}{\frac{T}{Ts}\cdot\frac{1}{T}\int_{t-T}^{t} k(\tau)d\tau} = \frac{1}{\frac{T}{Ts}\widetilde{k}(t)} = \frac{1}{s\cdot\widetilde{k}(t)}. \tag{10}$$

Here $\widetilde{k}(t)$ is the mean number of encoder pulses per sample time Ts calculated over time period T. As a result, in denominator we have a mean number of encoder pulses over T seconds. From Eq. (10) we can conclude that the relative error depends on the velocity of the mobile robot: the greater velocity the smaller is error and vice versa. Thus, if velocity is large then we can choose small period T and if velocity is small we must select large T in order to provide the same relative error.

The Eqs. (9) and (10) can be used for appropriate parameter T selection in order to provide desirable velocity estimation properties. Let e_r^* and $\Delta\widetilde{v}_w^*$ be given admissible variance and relative error. Then chosen parameter value must satisfy inequalities

$$\frac{2\pi R}{N \cdot T} < \Delta\widetilde{v}_w^*, \quad \frac{Ts}{I(t)-I(t-T)} < e_r^*. \tag{11}$$

These inequalities can be checked for different tunable parameter values during the number of experiments with fixed robot velocities.

It important to note that the proposed velocity estimation algorithm differs from the well-known method M/T in the following: we replace in Eq. (4) number of impulses $k(t)$ over sample time Ts with its mean value over time period T. This is necessary for rough encoders with small PPR number. Besides that, in this case parameters T and Ts are independent.

In this section for simplicity we had describe the algorithm for velocity estimation in continues time. The next section is devoted to real-time implementation of the proposed algorithm.

3.2 Real-time algorithm for velocity estimation

Taking into account that $T = s \cdot Ts$ we can derive digital analog of Eq. (6). As a result we get:

$$\widetilde{v}_w(t) = \frac{2\pi R}{s \cdot N \cdot Ts}\sum_{i=0}^{s-1} k(t_i). \tag{12}$$

Here t is a current moment of time, $t_i = t - i \cdot Ts$ is a previous time instants. So, the linear velocity of the wheel is estimated using last s measurements acquired from encoder. The discrete analog of integral in Eq. (7) takes the form of finite sum:

$$I_d(t) = \sum_{i=0}^{p} k(t_i),$$

where $t = p \cdot Ts$. Hence, variable $I_d(t)$ accumulate encoder data. Finally, from Eqs. (8) and (12) formula for linear velocity estimation is derived:

$$\widetilde{v}_w(t) = \frac{2\pi R}{s \cdot N \cdot Ts}\left(I_d(t) - I_d(t-T)\right). \tag{13}$$

It's important to note that the function $I_d(t)$ is a digital function with sample period Ts, which saves accumulated encoder data. But, in order to compute velocity estimation at the current time instant it's not necessary to save all previous values of this function. We only need to save last s values over time interval from $t-T$ to T.

Summarizing, the algorithm for wheel linear velocity estimation is as follows.

1) Create double array $Id[s+1]$ to save $s+1$ last values of the digital function $I_d(t)$. In this way $Id[0]$ is equal to $I_d(t-T)$ and $Id[s] = I_d(t)$.

2) At each sample instant t move elements of the array Id one position left and add current encoder data to the last array element, that is $Id[0] = Id[1], \ldots, Id[s-1] = Id[s], Id[s] = Id[s] + k(t)$.

3) Compute estimation of the wheel linear velocity using Eq. (13). This estimation is easily computed by subtraction of last and first elements of the array Id.

Presented algorithm allows computing wheel linear or angular velocity estimations in real-time onboard of a mobile robot.

4. WHEEL VELOCITY STABILIZATION

The general model of an underactuated wheeled mobile robot is represented by the following kinematic equations [2],[4]:

$$\dot{x} = v\cos\theta, \ \dot{y} = v\sin\theta, \ \dot{\theta} = \omega, \tag{14}$$

where (x, y) is a robot position on the plane with respect to some fixed coordinate system, θ is a heading angle of the robot, v and ω are control inputs representing robot linear and angular velocities correspondently.

It can be noted that the values v and ω in Eq. (14) are used as a control inputs while designing general control algorithms, such as positioning, path following, etc. At that time ones don't take into account how this algorithm will be implemented internally, that is by means of robot wheels and motors.

This section is devoted to the problem of given angular ω^* and linear v^* velocities realization for differential drive robots by means of wheels and motors.

The linear and angular velocities of the robot are connected with the linear wheels velocities by the following formulas [5]:

$$v_r = \frac{1}{2}(2v + L\omega), \quad v_l = \frac{1}{2}(2v - L\omega), \tag{15}$$

where v_r and v_l are linear velocities of the right and left wheels correspondently, L is a distance between left and right wheels of the robot. Let denote v_r^* and v_l^* given linear wheels velocities corresponding to the angular ω^* and linear v^* velocities of the robot in accordance with Eq. (15).

The pair of left wheels and the pair of right wheels are rotated with the same speed. The speed of rotations is determined by the voltage applied to the motors. This voltage is determined by the integer PWM value $p \in [0,255]$. We can approximate the dependence between wheel linear velocity and control value p. Let $v_{\min} > 0$ and $v_{\max}$ are minimal and maximal wheel velocities corresponding to values $p_{\min} \in [0,255)$ and $p_{\max} = 255$. Notice that the value $p_{\min}$ is a minimal voltage when the robot can move with nonzero speed. Now, let suppose that the relation between linear wheel velocity and applied voltage is linear. Then we obtain

$$p_i(v_i) = p_{\min} + \frac{p_{\max} - p_{\min}}{v_{\max} - v_{\min}}(v_i - v_{\min}), \quad i = l, r. \tag{16}$$

It important to note that Eq. (16) can be obtained experimentally for a particular mobile robot.

In order to provide given velocity for left and right wheels the following control algorithm can be used:

$$\begin{aligned} &\dot{u}_i = v_i^* - \hat{v}_i, \quad i = l, r, \quad k_u > 0, \; k_p > 0, \\ &\tilde{p}_i = k_u u_i + k_p\left(v_i^* - \hat{v}_i\right), \quad \tilde{p}_i(0) = p_i\left(v_i^*\right). \end{aligned} \tag{17}$$

Here k_u and k_p are positive real numbers determining the duration of the transient processes in closed-loop system, $\hat{v}_i$ is wheel linear velocity estimation obtained using encoder data as described above. The algorithm (17) is a simple PI-controller. In principle we can use here some other control algorithm providing zero tracking error with the better performance indexes.

The implemented PWM value p_i is computed as an integer part of $\tilde{p}_i$:

$$p_i = [\tilde{p}_i]. \tag{18}$$

So, the Eqs. (16) ~ (18) represents stabilizing controller for wheel linear velocity used to control speed of robots left and right wheels.

The real-time implementation of the proposed control algorithm (16) ~ (18) is straightforward and consist of the following steps.

1) Compute linear velocity estimations $\hat{v}_i$ for left and right wheels using Eq. (13) at each sample instant.

2) Compute control input u_i using Eq. (17) and evaluate $\tilde{p}_i$ by means of Euler formula.

3) Calculate PWM value using Eq. (18).

Presented digital control algorithm can be implemented onboard of a wheeled mobile robot.

5. PRACTICAL EXAMPLE

Let now consider practical example, illustrating proposed algorithms. Firstly, we are doing simulation examples and after that compare it with the real-time onboard trials. In our experiments we use DFRobot 4WD Rover as a particular differential drive robot. It has following features: wheel radius $R = 0.033$ m, distance between left and right wheels (width) $B = 0.145$ m, distance between forward and backward wheels axis (length) $L = 0.132$ m, minimum robot velocity $v_{\min} \approx 0.132$ m/s correspond to PWM value $p_{\min} = 100$ and maximum velocity $v_{\max} \approx 0.420$ m/s correspond to PWM value $p_{\max} = 200$. Robot also can move with a higher PWM values, but we use PWM values from 100 to 200 to obtain experimental data.

Encoder PPR value is $N = 20$. This is the actual PPR value of DFRobot Wheel Encoders for 4WD Rovers that we use in practice.

Using Eq. (2) coefficient k_γ of a wheel model (1) can be estimated. From a number of experiments with a mentioned mobile robot we get value $k_\gamma \approx 0.05$. In these experiments we apply constant PWM value p and measure approximate robot velocity.

The simulation example is performed in Matlab environment using Simulink-model shown in Fig. 1. This model contains blocks representing wheel model, velocity estimator, feedback controller and PWM module. The following parameters values are used in this model: sample period $Ts = 0.1$ s, control coefficients $k_u = 100, k_p = 1$.

Suppose that robot initially have zero velocity and the given velocities are $v^* = 0.2$ m/s, $\omega^* = 0$ rad/s. The corresponding initial PWM value according to Eq. (16) is equal to $p(v^*) = 124$.

The desirable estimation accuracy is determined by relative error $e_r^* = 0.01$. Using simulation experiments and Eq. (11) we find that minimal time period providing this accuracy is $T = 5$ s. It is important to note that the time period T depend on the robot velocity: the same time period results in different accuracy for the different velocity values.

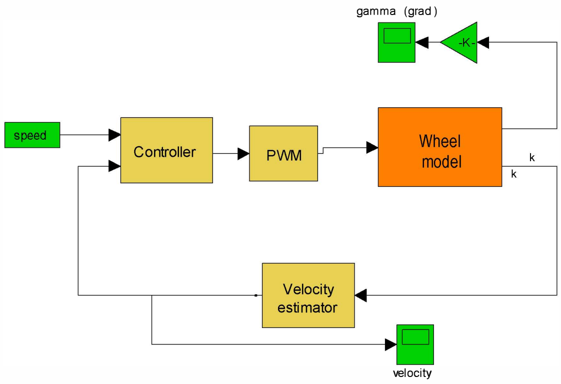

Fig. 1 Simulink diagram of the wheel model.

The transient process for mentioned given velocity $v^* = 0.2$ m/s and period $T = 5$ s is shown on Fig. 2 (wheel velocity) and Fig.3 (applied PWM values).

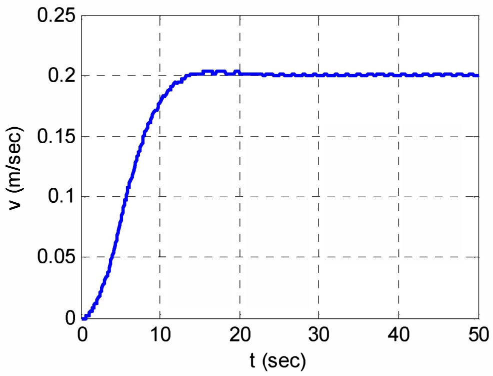

Fig. 2 Transient process for wheel velocity ($T = 5$ s).

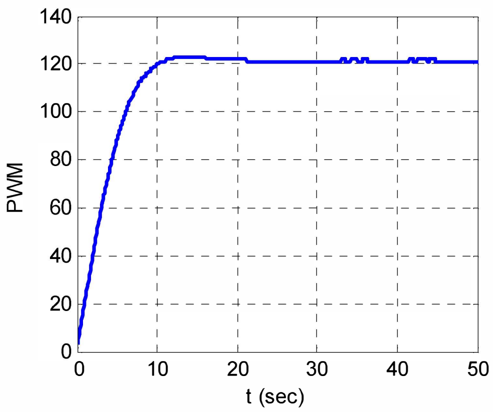

Fig. 3 Transient process for applied PWM ($T = 5$ s).

It can be seen from the pictures that the wheel velocity is oscillating around given linear velocity. This oscillation appears due to rough encoder data. If the PPR value of the encoder is larger the oscillations become smaller. Notice that the given linear velocity is tracked with a small error and relative estimation errors satisfy imposed condition (11).

Let now consider the case when interval over which velocity is estimated is $T = 2$ s. The correspondent transient process is presented on Figures 4 and 5.

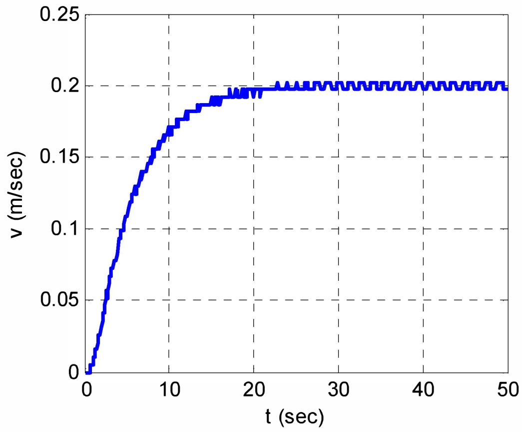

Fig. 4 Transient process for wheel velocity ($T = 2$ s).

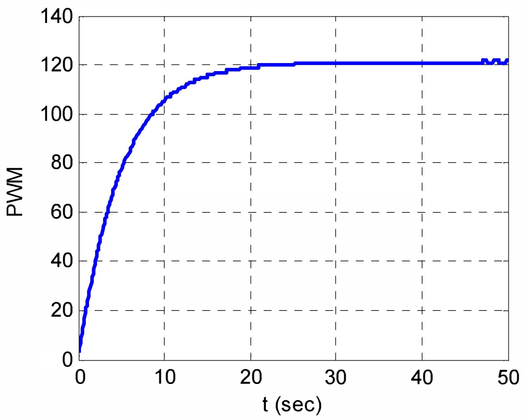

Fig. 5 Transient process for applied PWM ($T = 2$ s).

It can be seen from pictures that the oscillations become greater than in the previous case. Nevertheless, given linear velocity is tracked practically without errors and the transient time is about 20 seconds. In this case the relative error value is $e_r \approx 0.026$ as can be calculated from Eq. (10).

From analogous experiments with other given linear velocities we can make following conclusion. If the given velocity increases then the time interval length T can be decreased without essential losses. Vice versa, if the given velocity decreases then it is necessary to increase interval length T in order to provide appropriate velocity estimations. The inequality (11) can be used to provide desirable estimation properties.

Now let perform real-time experiments with a mobile platform. The proposed velocity estimation and control algorithms are implemented by means of Arduino development environment using Arduino Mega2560 board. Data from encoders and computed velocity estimations are obtained via serial communication port.

Let assume following parameters: given velocity $v^* = 0.2$ m/s and time period $T = 5$ s. The results of experiment are presented on Figs. 6 and 7. As can be seen from the pictures there is an overshoot in transient processes and the relative error e_r is greater than theoretical estimation given by Eq. (10). This is mainly due to unknown model dynamic that is the mathematical model (1) represents only ideal wheel behavior and can be improved. Let also notice that left and right wheels rotate with approximately the same speed. But, this demand different values of applied PWM because friction for left and right wheels is not equal.

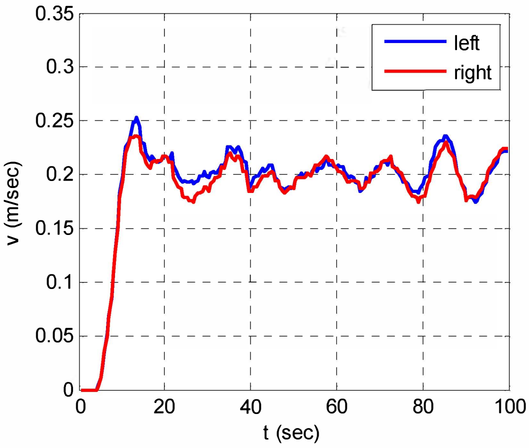

Fig. 6 Experimental data for wheels velocity ($T = 5$ s).

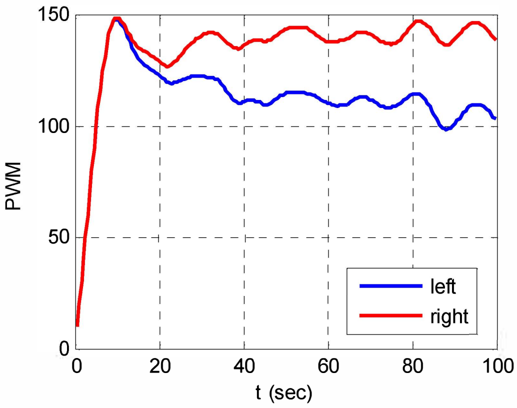

Fig. 7 Experimental data for applied PWM ($T = 5$ s).

Let now consider experiment with estimation period value $T = 2$ s. The correspondent experimental results are presented on Figs. 8 and 9. Comparing Figs. 6 and 8 we can conclude that the relative error remains approximately the same. But, in the second case the velocities estimations are more oscillate and the difference between left and right wheels velocities are greater than in the first case.

6. CONCLUSIONS

The problem of wheels angular velocity estimation and stabilization using encoder data is considered. The algorithm for wheel velocity estimation is proposed. The internal feedback controller synthesis for wheel velocity stabilization is performed. The corresponding digital algorithms for real-time implementation on-board of a mobile robot are described. The practical applicability of the approach is demonstrated using Matlab simulation study examples. The algorithms are implemented in Arduino development environment for a particular differential drive robot and the corresponding results are presented.

The reported study was partially supported by Russian Foundation for Basic Research (RFBR), research project No. 14-07-00083 a.

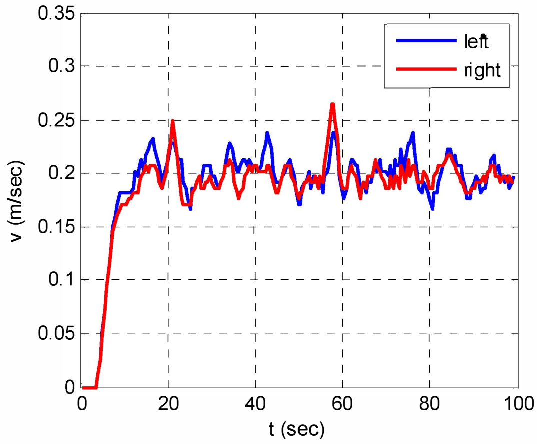

Fig. 8 Experimental data for wheels velocity ($T = 2$ s).

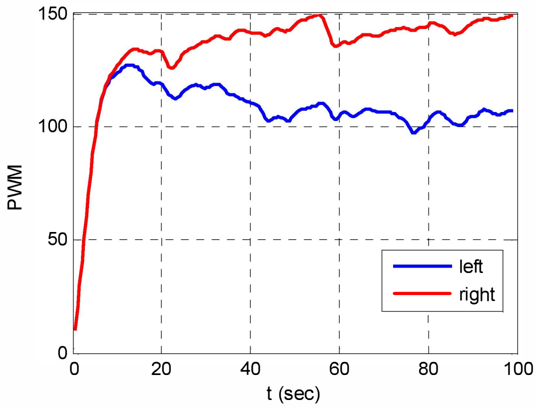

Fig. 9 Experimental data for applied PWM ($T = 2$ s).

REFERENCES

[1] T. Braunl, *Embedded Robotics: Mobile Robot Design and Applications with Embedded Systems,* Springer, 3rd edition, 2008, 546 p.

[2] P. Corke, Robotics, Vision and Control. Fundamental algorithms in MATLAB, Springer, 2011, 558 p.

[3] T. Ohmae, T. Matsuda, K. Kamiyama, M. Tachikawa. *A Microprocessor controlled High-Accuracy Wide-Range Speed Regulator for Motor Drives.* IEEE Trans. on Industrial Electronics, vol.29, No.3, 1982, p.207-221.

[4] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, *Robotics: Modelling, Planning and Control,* Springer, 2009, 632 p.

[5] B. Siciliano, O. Khatib (Eds.), *Handbook of Robotics,* Springer, 2008, 1611 p.