Neural Network Based Correction of Odometry Errors in Mobile Robots

I. Petrović, E. Ivanjko, and N. Perić
Department of Control and Computer Engineering in Automation,
Faculty of Electrical Engineering and Computing,
University of Zagreb,
HR-10000 Zagreb, Croatia
{ivan.petrovic,edouard.ivanjko,nedjeljko.peric}@fer.hr

Abstract

Odometry is the most widely used method for estimation of the momentary position of mobile robots. It provides easily accessible real-time positioning information. However, with time, odometric localization accumulates errors in an unbounded fashion. This paper describes a way to decrease the odometry error by using a neutral network as a soft sensor. It makes a correction to the estimated orientation of the mobile robot, which is the most influenced on accumulated odometry errors. Trained neural network can be used instead of a gyro or a compass in some mobile robot applications.

1 Introduction

One of the major tasks of autonomous robots navigation is the mobile robot localization [1]. In a typical indoor environment with a flat floor plan, localization becomes a matter of estimating the Cartesian coordinates (x, y) and the orientation Θ. Odometry is one of the most important means of achieving this task. This method uses encoder data and is a simple, inexpensive and easy way to determine the offset from a known start position in real time. The encoder data are proceeded to the central processor that in turns continually updates the mobile robot's position using geometric equations. The disadvantage is its unbounded accumulation of errors due to wheel slippage, floor roughness, discretised sampling of wheel speed data, inaccessibility to the angular velocities of the wheels in some mobile robots etc. But improved odometry can significantly reduce the cost for installation of mobile robot systems because it simplifies the position estimation problem.

A lot of research works have been undergone in order to improve the reliability of odometry. The reliability of odometry can be improved using methods for correction of mobile robot position estimation errors [2, 3] and using better odometry or error models [4]. Systematic errors can be decreased using these methods. Sensor fusion using a Kalman Filter and calibration [5] is an another method to improve the position estimation of a mobile robot. Calibration to correct the systematic errors can be made with various experiments before and during the actual use of the mobile robot [6].

However, these methods use additional sensors to improve the odometry or they can take into account only systematic errors. Non-systematic errors are unpredictable and calibration techniques can't take them into account. The sources are uneven floors, unexpected objects on the floor, wheel slippage due to slippery floors etc. Work done at the theoretical level normally involves non-systematic errors quantification via modelling, so that some kind of mathematic treatment would be possible. For instance, many robust stochastic based techniques such as the Kalman Filter [5, 6] require that the odometry errors are statistically quantified in the form of an *error covariance matrix*, so that it can be fused with the information provided by the external reference to produce a linear *minimum mean square* estimate of the position and orientation. Values of the error covariance matrixes are obtained empirically from the sensor specifications or from different sets of experimental data.

Robot orientation Θ is the most significant of the localization parameters (x, y, Θ) in terms of its influence on accumulated dead-reckoning errors. For this reason, sensors that provide a measure of absolute orientation or relative angular velocity are extremely important in solving the real world navigation needs of an autonomous platform. The most commonly used sensors of this type are probably the magnetic compass and the gyro. Fusion of the odometry and compass or gyro information by using Kalman Filter can provide substantial increase of robot localization accuracy, e.g. [5]. However, the price of the system is a limiting factor in many commercial applications. Thus, in such applications it can be beneficial to avoid the use of compass/gyro, but at the same time to keep the accuracy of the robot pose estimation as high as possible.

In this paper we propose a system of odometry errors corrections based on a neural network. Actually the neural network is used as a soft sensor [7]. We have assumed that the robot manufacturer has a testing room for navigation systems that enables online measurement of robot actual location. The values of the localization parameters from such a system can then be used as reference value for neural network training. In our case reference values of the localization parameters are obtained by the fusion of odometry and compass signals using Kalman Filter.

The paper is organized as follows. In section 2 the robot model is given with short description of calibration procedure. Then, section 3 presents the fusion of calibrated odometry with compass using Kalman Filter. The description of the proposed neural network based odometry errors correction system is given in section 4. The section 5 contains the simulation results with all three systems.

2 The Robot Model and Odometry Calibration

Mobile robot used in our experiments is a three-wheeled robot (Pioneer 2DX of ActivMedia Robotics). Two front wheels are drive wheels with encoders mounted on them and the third wheel is a castor wheel to ensure stability. Drive wheels can be controlled independently from each other. The encoders can measure the speed or the travelled distance of the wheel. We are using the encoders to measure the speed of the wheel. The kinematics of the robot are given by the following relations (Figure 1):

$$x_{k+1} = x_k + v_{tot_k} \cdot T \cdot \cos \Theta_{k+1}, \tag{1}$$

$$y_{k+1} = y_k + v_{tot_k} \cdot T \cdot \sin \Theta_{k+1}, \tag{2}$$

$$\Theta_{k+1} = \Theta_k + \Delta\Theta_k, \tag{3}$$

$$v_{tot_k} = \frac{v_{L_k} + v_{R_k}}{2}, \tag{4}$$

$$\Delta\Theta_k = \frac{180}{\pi} \cdot \frac{v_{R_k} - v_{L_k}}{b} \cdot T. \tag{5}$$

where: x_k and y_k denote the position of the center of axle [m]; v_{tot_k} the total translational speed [m/s]; T the sampling time step [s]; Θ_k [°] the angle between the vehicle and the x axis; v_{L_k} and v_{R_k} denote the velocities of the left and right wheel, respectively [m/s] and b the vehicle axle length [m].

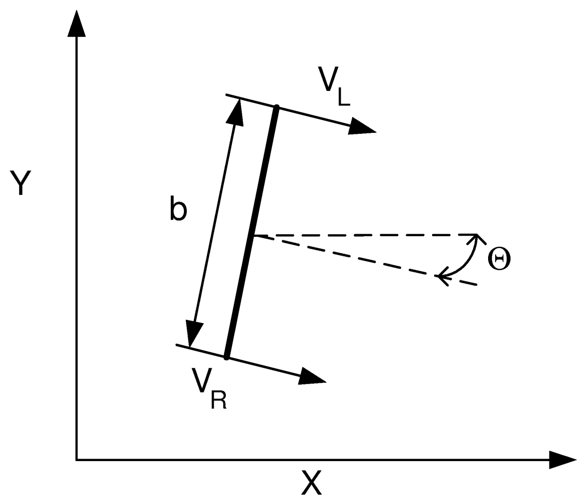

Figure 1: The Mobile Robot Kinematics.

The robot specifications give the value of axle length b, but during the turns the wheels don't pivot at their center. This changes of the effective axle length produce a systematic error. To compensate this systematic error connected with the axle length, the Equation 5 of robot model is expanded with a correction parameter β:

$$\Delta\Theta_k = \frac{180}{\pi} \cdot \frac{v_{R_k} - v_{L_k}}{\beta \cdot b} \cdot dt. \tag{6}$$

The correction parameter β is obtained empirically in various turn experiments.

3 Odometry and Compass Fusion

Kalman filtering is a well known technique for state and parameter estimation. It is a recursive procedure that uses sequential sets of measurements. Prior knowledge of the state is expressed by the covariance matrix and it is improved at each step by taking prior estimates and new data for the subsequent state estimation. Kalman Filter based localization is a common practice in mobile robotics [5, 6].

To improve the estimate of orientation in the x-y plane a compass was used (TCM2-20 from Precision Navigation, Inc.). Before using it for estimating orientation its measurement noise covariance and system noise covariance matrices were determined using sensor observations and empirical observations. The equal procedure was repeated for the encoders.

In the experiments reported here the measurement vector used in the localization is composed of the translational speeds of left and right wheels v_L, v_R and the mobile robot orientation Θ_C measured by the compass. The state estimate $\hat{\mathbf{x}}$, the measurement estimate $\hat{\mathbf{z}}$, the measurement vector $\mathbf{z}$ and the residual vector $\mathbf{r}$ are defined as:

$$\hat{\mathbf{z}} = \hat{\mathbf{x}} = \begin{bmatrix} \hat{v}_L & \hat{v}_R & \hat{\Theta} \end{bmatrix}^T,$$
$$\mathbf{z} = \begin{bmatrix} v_L & v_R & \Theta_c \end{bmatrix}^T, \mathbf{r} = \mathbf{z} - \hat{\mathbf{z}}. \tag{7}$$

Kalman Filter consists of two different steps: **propagation** and **update**. The equations for the propagation step are:

$$\hat{\mathbf{x}}_{k+1/k} = \Phi \cdot \hat{\mathbf{x}}_{k/k}, \tag{8}$$

$$P_{k+1/k} = \Phi \cdot P_{k/k} \cdot \Phi^T + Q. \tag{9}$$

The equations for the update step are:

$$K = P_{k+1/k} \left(P_{k+1/k} + R\right)^{-1}, \tag{10}$$

$$\hat{\mathbf{x}}_{k+1/k+1} = \hat{\mathbf{x}}_{k+1/k} + K \cdot \mathbf{r}, \tag{11}$$

$$P_{k+1/k+1} = (I - K)P_{k+1/k}. \tag{12}$$

In the above equations Φ is the system matrix, P is the error covariance matrix, Q is the system noise covariance matrix, K is the Kalman gain matrix and R is the measurement noise covariance matrix.

From the equations 1-4 and equation 6, the system matrix Φ is obtained:

$$\Phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{T}{\beta \cdot b} \cdot \frac{180}{\pi} & \frac{-T}{\beta \cdot b} \cdot \frac{180}{\pi} & 1 \end{bmatrix}. \tag{13}$$

4 Structure of the Neural Network Based Odometry Errors Correction

The structure of the neural network based odometry errors correction is shown in Figure 2. The proposed system consists of a odometry block and a neural network. The odometry block calculates the mobile robot position using the robot model described in section 2. The neural network calculates the mobile robot orientation change and compensates the systematic and non-systematic odometry errors. Thus, neural network is used in robot modelling instead of Equation 6. The benefit of using neural network is possibility to establish nonlinear relation between the wheel velocities and the change in the robot orientation. We used a Radial Basis Function (RBF) neural network that can be described as:

$$\begin{aligned} \Delta \hat{\Theta}(k) &= f_N(\varphi(k), \mathbf{w}) = \\ &= f_N \left(\Delta v(k), \Delta v(k-1), \right. \\ &\qquad \left. \hat{\Theta}(k), \hat{\Theta}(k-1), \Delta \hat{\Theta}(k-1), \mathbf{w}\right) \end{aligned} \tag{14}$$

where $f_N(\cdot)$ is given by:

$$f_N = \mathbf{w_2} \cdot e^{-(\varphi(k) - \mathbf{w_1})^2 \cdot \mathbf{b_1}^2} + b_2 \tag{15}$$

and where $\mathbf{w} = \begin{bmatrix} \mathbf{w_1} & \mathbf{b}_1 & \mathbf{w}_2 & b_2 \end{bmatrix}$ is vector of neural network parameters and φ neural network input regression vector.

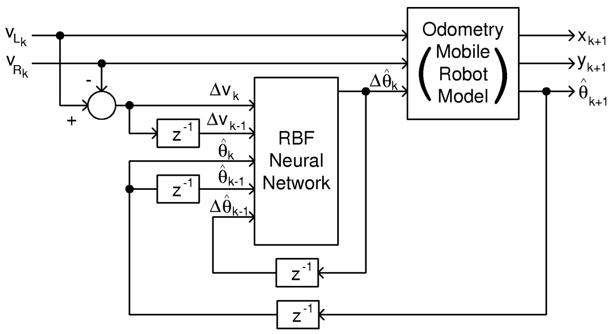

Figure 2: The Neural Network Odometry Error Correction-Operational Structure.

As shown in Figure 2 the neural network has five inputs and one output. The inputs are the current mobile robot orientation and the orientation from the previous time frame, the current wheel speed difference and the wheel speed difference from the previous time frame and mobile robot orientation change from the previous time frame. The calculated orientation change $\Delta \hat{\Theta}_k$ is passed to the mobile robot model given with Equations 1 - 4 to calculate the new mobile robot position $(x,\, y)$ and orientation Θ.

The training of neural network is performed off-line. The neural network is trained to capture the errors coupled with the mobile robot orientation. We used a compass to measure the actual mobile robot orientation and the change in orientation. During the network training, the structure from Figure 2 becomes like shown in Figure 3, and Equation 15 becomes:

$$\Delta \overset{\wedge}{\Theta}(k) = f_N(\Delta v(k), \Delta v(k-1), \Theta_C(k), \Theta_C(k-1), \Delta\Theta_C(k), \mathbf{w}) . (16)$$

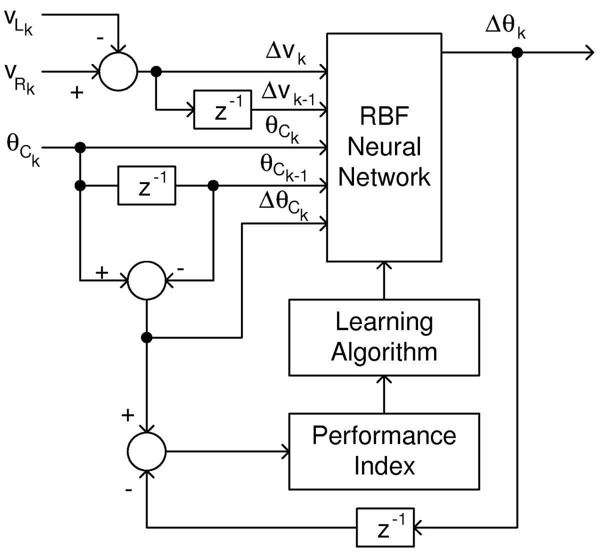

Figure 3: The Neural Network Odometry Error Correction-Training Structure.

The performance index that is minimized by neural network training is:

$$J = \sum_{\nu=1}^{N} \left(\Delta\Theta_c(\nu) - \Delta\hat{\Theta}(\nu) \right)^2 \qquad (17)$$

where N is the number of training data.

To design the neural network we used the Neural Network Toolbox for Matlab [8]. Using the Matlab-Simulink two Band Limited White Noise Signals (BLWNS) were generated to randomly and independently change the left and right wheel speeds. It is essential that the mobile robot orientation cover the whole range from 0 to 360 degrees in the experiment. The wheel speed difference must also cover the whole assumed range. The learning data directly influence the quality of the mobile robot orientation correction so special attention should be paid to the preparation of the experiments and collected data processing.

We collected the measured left and right speed values, the mobile robot orientation and orientation change measured with a compass. These data were used to train the neural network using the *newrb* function in Matlab. This function creates a two layer network. It adds one neuron to the neural network at a learning step until the sum-squared error (Equation 17) falls beneath an error goal or a maximum number of neurons is reached. The learning method is the orthogonal least square algorithm [9]. That is a standard neural network learning algorithm implemented in the Matlab Neural Network toolbox. It uses an input vector and a target class vector to design a new radial basis neural network. That means that the whole neural network structure, the input weights values, the layer weights values and the bias value have been obtained using this algorithm. A number of 45 neurons with the radial basis functions (spread = 1.0) in the hidden layer was necessary to accomplish the presented results.

5 Simulation Results

The experiments were carried out in the Saphira simulation environment for the Pioneer DX2 mobile robot platform. We performed 5 standard experiments, each with a different trajectory shown in Figures 4 - 8. The mobile robot initial position in every experiment is at (0,0,0). All measurements of the mobile robot position estimation are made in millimeters. The exact final mobile robot position was measured by the mobile robot simulator itself, but the resolution was only 0.1 [m]. Each figure shows three different position estimation algorithms: a) calibrated odometry that uses a calibration on the effective wheel axle length, b) Kalman Filter that uses sensor fusion (wheel speeds and a compass) for the mobile robot position estimation and finally c) our neural network position estimation algorithm. The results presented by a dashed line are for calibrated odometry, the results presented by a dash-dotted line are for Kalman Filter and the results presented by a solid line are for neural network. Each figure has also a cross mark for the actual final mobile robot position and dot marks for the estimated final mobile robot positions.

The first and second experiments are the cases of the triangular trajectories. The triangle is rectangular with 3 [m] sides. In the first experiment the robot did only left turns and in the second only right turns. The third and fourth experiments are the cases of the square trajectories. The square side has a length of 3 [m]. These two tests were also made with only left or only right turns. The last experiment is the case of a strait line trajectory. The mobile robot had to travel in a strait line with a 6 [m] length. In each experiment the mobile robot had to return to the start position and in an ideal case the final mobile robot position would be the same as the start position, i.e. position

Experiment	Calibrated Odometry	Neural Network	Kalman Filter
Left Triangle	3.8 %	2.3 %	0.5 %
Right Triangle	0.5 %	0.5 %	0.3 %
Left Square	3 %	1.2 %	0.6 %
Right Square	3.7 %	0.8 %	0.8 %
Strait Line	4.8 %	1.5 %	0.4 %

Table 1: Error comparison of three different techniques.

(0,0,0). The travelled distance is 12 [m] for the square and the strait line experiment and 10.2 [m] for the triangle experiments.

Table 1 summaries the results of five experiments conducted using three different localization techniques. All errors presented in table 1 are calculated as:

$$Error = \frac{Pos_{act} - Pos_{est}}{Dist} \cdot 100\% \qquad (18)$$

where Pos_{act} is the actual final position, Pos_{est} is the estimated final position and $Dist$ is the total distance traversed by the mobile robot.

As it is expected the best results are achieved using the Kalman Filter. But this technique uses an additional sensor (compass). The neural network based odometry system gives much better results then the calibrated odometry, without additional sensors. When we compare the trajectories in the Figures 4 - 8, we can also see that the neural network technique estimates much better the real trajectory of the mobile robot then the calibrated odometry. The proposed mobile robot position estimation technique can be used to decrease the number of sensors without a great decrease of the position estimation accuracy.

6 Conclusion

A neural network based odometry errors correction system for mobile robots is developed and experimentally compared in simulations to the common used position estimation methods. It is shown that the proposed system can give an improvement of the position estimation of a mobile robot. Using this method there is no need to use additional sensors during the mobile robot operation. Also the neural network can learn systematic errors so there is no need for expensive and time consuming calibration measurement. Before the actual use of the mobile robot in new environment, just a simple test to adapt the neural network to the new environment has to be done. The new environment can change the value and nature of the systematic and non-systematic errors but the neural network can be easily adapted to this new conditions.

References

[1] J. Borenstein, H.R. Everett, *Where am I? Sensors and Methods for Mobile Robot Positioning*, University of Michigan, 1996.

[2] J. Borenstein, L. Feng, Measurement and Correction of Systematic Odometry Errors in Mobile Robots, *IEEE Transactions on Robotics and Automation*, Vol. 12, No 6, 1996.

[3] J. Borenstein, L. Feng,UMBmark - A Method for Measuring, Comparing and Correcting Dead-reckoning Errors in Mobile Robots,*University of Michigan*, Technical Report UM-MEAM-94-22, 1994.

[4] K.S. Cong, L. Kleeman, Accurate Odometry and Error Modeling for a Mobile Robot, *Monash University, Department of Electrical and Computer Systems Engineering*, Technical Report MECSE-1996-6, 1996.

[5] P. Goel, S.I. Roumeliotis, G.S. Sukhatme, *Robust Localization Using Relative and Absolute Position Estimates*, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 1999.

[6] T.D. Larsen, M. Bak, N.A. Andersen, O. Ravn, *Location Estimation for an Autonomously Guided Vehicle using an Augmented Kalman Filter to Autocalibrate the Odometry*, First International Conference on Multisource - Multisensor Information Fusion, FUSION'98 , 1998.

[7] Y. Yang, T. Chai, Soft sensing based on artificial neural network, *American Control Conference*, 1997.

[8] H. Demuth, M. Beale, *Neural Network Toolbox For Use with Matlab*, The MathWorks Inc., 2000.

[9] S. Chen, A. Billings, M. Grant, Recursive Hybryd Algorithm for Non-Linear Identification using Radial Basis Function Networks, *International Journal of Control*, Vol. 5, pp. 1051-1070, 1992.

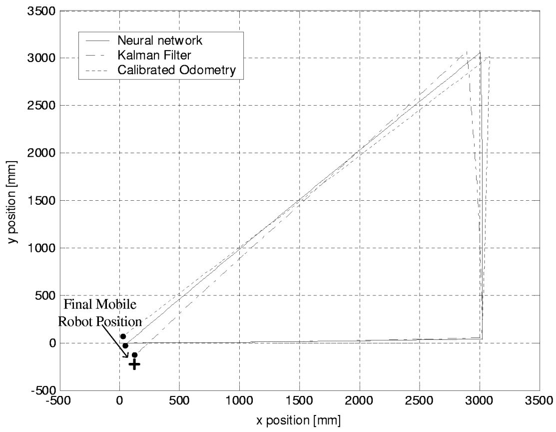

Figure 4: The left triangle experiment.

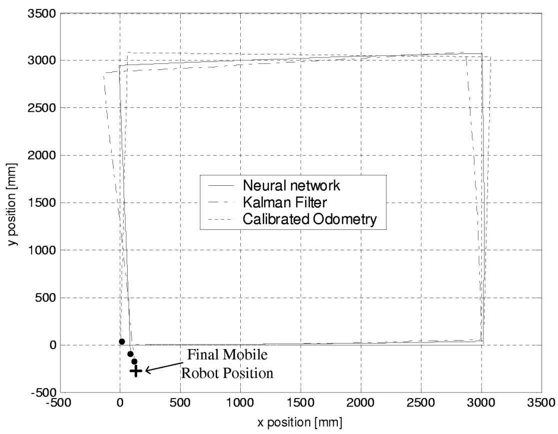

Figure 5: The left square experiment.

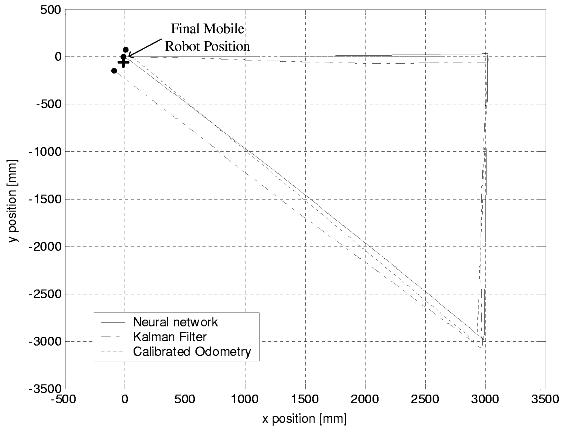

Figure 6: The right triangle experiment.

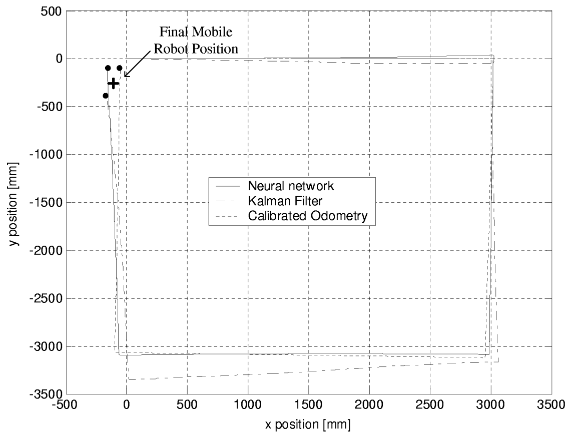

Figure 7: The right square experiment.

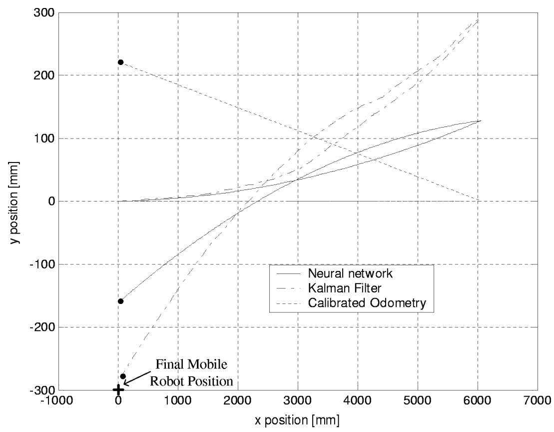

Figure 8: The strait line experiment.