Sensor and actuator modeling of a realistic wheeled mobile robot simulator

José Gonçalves, José Lima, Hélder Oliveira and Paulo Costa
Faculdade de Engenharia da Universidade do Porto, DEEC
Porto, Portugal
Email: {goncalves, jllima}@ipb.pt,{helder.oliveira, paco}@fe.up.pt

Abstract

This paper describes the sensor and actuator modeling of a realistic wheeled mobile robot simulator. The motivation of developing such simulator is to produce a personalized versatile tool that allows production and validation of robot software reducing considerably the development time. The mobile robot simulator was developed in Object Pascal with its dynamics based on the ODE (Open Dynamics Engine), allowing to develop robot software for a three wheel omnidirectional robot equipped with Infra-Red distance sensors and brushless motors.

1. Introduction

Code migration from realistic simulators to real world systems is the key for reducing development time of robot control, localization and navigation software [3]. Due to the complexity of robot, world, sensors, and actuators modeling it is not an easy task to develop such simulator. The presented case study is the simulation of a three wheel omnidirectional robot equipped with brushless motors and Infra-Red distance sensors (illustrated by a red beam). A snapshot of the 3D mobile robot simulator is presented in Figure 1, where is shown a three wheel omnidirectional robot with some distance sensors, some obstacles and a ball.

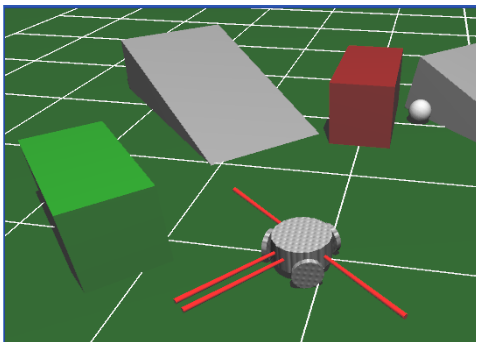


Figure 1. Robot simulator snapshot.

One of the most popular robot simulator software is Webots^{TM}, it is used in over 500 universities and research centers worldwide to model, program and simulate mobile robots. The included robot libraries enable the transfer of control programs to several commercially available real mobile robots [1] [12]. In order to have full control, the authors developed an robot simulator software, despite of the commercially available present excellent results. The main motivation to develop such software is to have the possibility of adding new features, like: sensors, actuators and omnidirectional wheels that are not included in the available commercial softwares. Introducing new features in the mobile robot simulator is important for the authors because they do not usually use standard commercial robots, they prototype their own robots, always reaching higher performances.

2. Robot dynamics modeling

It is essential, in order to achieve controllers that provide higher performances, to obtain precise dynamical models. Models are based on linear and non-linear dynamical systems, its parameters estimation has been subject of continuous research [6] [11] [15] [7]. The most common methods for dynamical systems parameters identification are the Least Squares method and Instrumental Variables [4]. The modeled robot is a three wheel omnidirectional robot [10], equipped with brushless motors and some distance sensors [5], being presented in Figure 2. The option of using brushless motors was made because of its higher performance, when compared with the typically used DC motors.

Omnidirectional vehicles are widely used in robotics soccer, allowing movements in every direction, where the extra mobility is an important advantage. The fact that the robot is able to move from one place to another with independent linear and angular velocities contributes to minimize the reaction time, the number of maneuvers is reduced and consequently the game strategy can be simplified [9].

2.1 Mechanical Configuration

Figure 3 presents the configuration of the three wheel robot, as well as all axis, relevant forces and velocities of the robotic system. The wheels are separated by 120 degrees.

1-4244-1506-3/08/$25.00 ©2008 IEEE

Figure 2. Omnidirectional robot prototype.

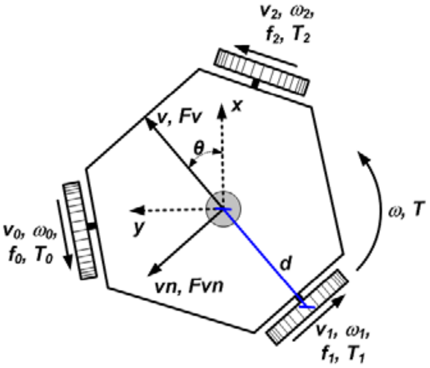


Figure 3. Three wheel Omnidirectional robot.

Figure 3 also shows the notation used through-out this paper, detailed as follows:

- x, y, θ - Robot's position (x,y) and θ angle to the defined front of robot;
- $d\ [m]$ - Distance between wheels and center robot;
- $v_0, v_1, v_2\ [m/s]$ - Wheels linear velocity;
- $\omega_0, \omega_1, \omega_2\ [rad/s]$ - Wheels angular velocity;
- $f_0, f_1, f_2\ [N]$ - Wheels traction force;
- $T_0, T_1, T_2\ [N \cdot m]$ - Wheels traction torque;
- $v, vn\ [m/s]$ - Robot linear velocity;
- $\omega\ [rad/s]$ - Robot angular velocity;
- $F_v, F_{vn}\ [N]$ - Robot traction force along v and vn;
- $T\ [N \cdot m]$ - Robot torque (respects to ω).

2.2 Model

2.2.1 Kinematic

The kinematic model of an omnidirectional robot located at (x, y, θ) can be written as $v_x(t) = dx(t)/dt$, $v_y(t) = dy(t)/dt$ and $\omega(t) = d\theta(t)/dt$ (please refer to Figure 3 for notation issues). Equation 1 allows to convert the linear velocities v_x and v_y on the static axis to linear velocities v and vn on the robot local axis.

$$X_R = \begin{bmatrix} v(t) \\ vn(t) \\ \omega(t) \end{bmatrix} \;;\; X_0 = \begin{bmatrix} v_x(t) \\ v_y(t) \\ \omega(t) \end{bmatrix}$$

$$X_R = \begin{bmatrix} cos(\theta(t)) & sin(\theta(t)) & 0 \\ -sin(\theta(t)) & cos(\theta(\pi)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot X_0 \qquad (1)$$

The relationship between the wheel velocities and the robot velocities is:

$$\begin{bmatrix} v_0(t) \\ v_1(t) \\ v_2(t) \end{bmatrix} = \begin{bmatrix} -sin(\pi/3) & cos(\pi/3) & d \\ 0 & -1 & d \\ sin(\pi/3) & cos(\pi/3) & d \end{bmatrix} \cdot \begin{bmatrix} v(t) \\ vn(t) \\ \omega(t) \end{bmatrix} \qquad (2)$$

It is possible to obtain the robot velocity equations related with wheels velocity, applying the inverse of the kinematics matrix presented in equation 2. The obtained equations are:

$$\begin{aligned} v(t) &= (\sqrt{3}/3) \cdot (v_2(t) - v_0(t)) && (3) \\ vn(t) &= (1/3) \cdot (v_2(t) + v_0(t)) - (2/3) \cdot v_1(t) && (4) \\ \omega(t) &= (1/(3 \cdot d)) \cdot (v_0(t) + v_1(t) + v_2(t)) && (5) \end{aligned}$$

2.2.2 Dynamics

The dynamical equations relative to the acceleration can be described as follows:

$$\begin{aligned} M \cdot \frac{dv(t)}{dt} &= \sum F_v(t) - F_{Bv}(t) - F_{Cv}(t) && (6) \\ M \cdot \frac{dvn(t)}{dt} &= \sum F_{vn}(t) - F_{Bvn}(t) - F_{Cvn}(t) && (7) \\ J \cdot \frac{d\omega(t)}{dt} &= \sum T(t) - T_{B\omega}(t) - T_{C\omega}(t) && (8) \end{aligned}$$

where the following parameters relate to the robot:

- $M\ [kg]$ - mass;
- $J\ [kg \cdot m^2]$ - inertia moment;
- $F_{Bv}, F_{Bvn}\ [N]$ - viscous friction forces along v and vn;
- $T_{B\omega}\ [N \cdot m]$ - viscous friction torque with respect to the robot's rotation axis;
- $F_{Cv}, F_{Cvn}\ [N]$ - Coulomb frictions forces along v and vn;
- $T_{C\omega}\ [N \cdot m]$ - Coulomb friction torque with respect to robot's rotation axis.

Viscous friction forces are proportional to the robot's velocities, $F_{Bv}(t) = B_v \cdot v(t)$, $F_{Bvn}(t) = B_{vn} \cdot vn(t)$ and $T_{B\omega}(t) = B_\omega \cdot \omega(t)$, where $Bv, Bvn\ [N/(m/s)]$ are the viscous friction coefficients for directions v, vn and $B_\omega\ [N \cdot m/(rad/s)]$ is the viscous friction coefficient for ω.

The Coulomb friction forces are constant in amplitude $F_{Cv}(t) = C_v \cdot sign(v(t))$, $F_{Cvn}(t) = C_{vn} \cdot sign(vn(t))$ and $T_{C\omega}(t) = C_\omega \cdot sign\ \omega(t)$, where $Cv, Cvn\ [N]$ are Coulomb friction coefficient for directions v, vn and $C\omega\ [N \cdot m]$ is the Coulomb friction coefficient for ω.

The relationship between the robot traction forces and rotation torque, with the traction forces on the wheels, is described by the following equations:

$$\begin{aligned} \sum F_v(t) &= (f_2(t) - f_0(t)) \cdot \sin(\pi/3) && (9)\\ \sum F_{vn}(t) &= -f_1(t) + (f_2(t) + f_0(t)) \cdot \cos(\pi/3) && (10)\\ \sum T(t) &= (f_0(t) + f_1(t) + f_2(t)) \cdot d && (11) \end{aligned}$$

The traction force on each wheel is estimated relating it with the traction torque, which can be determined using the motor current, as described in the following equations:

$$\begin{aligned} f_j(t) &= T_j(t)/r && (12)\\ T_j(t) &= l \cdot K_t \cdot i_j(t) && (13) \end{aligned}$$

- l - Gearbox reduction;
- $r\ [m]$ - Wheel radius;
- $K_t\ [N \cdot m/A]$ - Motor torque constant;
- $i_j\ [A]$ - Motor current (j=motor number).

2.2.3 Motor

The omnidirectional robot prototype uses brushless motors for its locomotion. This type of motors has been more common in the last years, because of its higher performance when compared with the common DC motors, since they don't have mechanical switching. The model for brushless motors is similar to the common DC motors, based on [13]:

$$\begin{aligned} u_j(t) &= L \cdot \frac{di_j(t)}{dt} + R \cdot i_j(t) + K_v \cdot \omega_{mj}(t) && (14)\\ T_{mj}(t) &= K_t \cdot i_j(t) && (15) \end{aligned}$$

- $L\ [H]$ - Motor inductance;
- $R\ [\Omega]$ - Motor resistor;
- $K_v\ [V/(rad/s)]$ - EMF motor constant;
- $u_j\ [V]$ - Motor voltage (j=motor number);
- $\omega_{mj}\ [rad/s]$ - Motor angular velocity (j=motor number);
- $T_{mj}\ [N \cdot m]$ - Motor torque (j=motor number).

The previous electrical motor model (equation 14) includes an electrical pole and a much slower, dominant mechanical pole - thus making inductance L value negligible. Thus, the motor model can be rewritten as follows:

$$u_j(t) = R \cdot i_j(t) + K_v \cdot \omega_{mj}(t) \quad (16)$$

2.3 Robot Model

By combining previously presented equations, it is possible to write the model equations in state space:

$$\begin{aligned} (dx(t)/dt) &= A \cdot x(t) + B \cdot u(t) + K \cdot sign(x) && (17)\\ x(t) &= [v(t)\ \ vn(t)\ \ w(t)]^T && (18) \end{aligned}$$

Using equations shown on subsection 2.2.2 and equation 16 the following equations can be achieved:

$$\begin{aligned} A &= \begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{bmatrix} && (19)\\ A_{11} &= -\frac{3 \cdot K_t^2 \cdot l^2}{2 \cdot r^2 \cdot R \cdot M} - \frac{B_v}{M} \\ A_{22} &= -\frac{3 \cdot K_t^2 \cdot l^2}{2 \cdot r^2 \cdot R \cdot M} - \frac{B_{vn}}{M} \\ A_{33} &= -\frac{3 \cdot d^2 \cdot K_t^2 \cdot l^2}{r^2 \cdot R \cdot J} - \frac{B_w}{J} \end{aligned}$$

$$B = \frac{l \cdot K_t}{r \cdot R} \cdot \begin{bmatrix} -\sqrt{3}/(2 \cdot M) & 0 & \sqrt{3}/(2 \cdot M) \\ 1/(2 \cdot M) & 1/M & 1/(2 \cdot M) \\ d/J & d/J & d/J \end{bmatrix} \quad (20)$$

$$K = \begin{bmatrix} -C_v/M & 0 & 0 \\ 0 & -C_{vn}/M & 0 \\ 0 & 0 & -C_w/J \end{bmatrix} \quad (21)$$

2.4 Parameter Estimation

To estimate the model parameters it is necessary to measure the motor current, robot position and velocity. Currents are measured by the electronics drive, position is measured using external global vision [8] and velocities are estimated from positions resorting to a first order approximation.

The parameters that must be identified are the viscous friction coefficients $(Bv, Bvn, B\omega)$, the Coulomb friction coefficients $(Cv, Cvn, C\omega)$ and inertia moment J. The robot mass was measured, and it was 1.944 kg.

The experimental runs were made using a step voltage with an initial acceleration ramp.

As shown in subsection 2.3 the model was defined by equation 17 and the Least Squares method was used to estimate the parameters. The system model equation can be rewritten as equation 22, where $x_1 = x(t)$, $x_2 = u(t)$, $x_3 = 1$ and $y = dx(t)/dt$.

$$y = \theta_1 \cdot x_1 + \theta_2 \cdot x_2 + \theta_3 \cdot x_3 \quad (22)$$

The parameter θ is estimated using:

$$\theta = \left(x^T \cdot x\right)^{-1} \cdot x^T \cdot y \quad (23)$$

$$x = [x_1(1) \ldots x_1(n)\ x_2(1) \ldots x_2(n)\ x_3(1) \ldots x_3(n)]^T \quad (24)$$

Estimated parameters can be skewed and for this reason instrumental variables are used to minimize the error, with state vector defined as:

$$z = [\overline{x_1}(1) \ldots \overline{x_1}(n)\ x_2(1) \ldots x_2(n)\ x_3(1) \ldots x_3(n)]^T \quad (25)$$

The parameter θ is now calculated by:

$$\theta = \left(z^T \cdot x\right)^{-1} \cdot z^T \cdot y \quad (26)$$

The numerical values for all estimated and fixed parameters are presented in Table 1.

Table 1. Dynamical model parameters.

Parameters	Values
$d\ (m)$	0.089
$r\ (m)$	0.0325
l	5
$K_v\ (V/(rad/s))$	0.0259
$R\ (\Omega)$	4.3111
$M(kg)$	1.944
$J\ (kg \cdot m^2)$	0.0169
$B_v\ (N/(m/s))$	0.5082
$B_{vn}\ (N/(m/s))$	0.4870
$B_\omega\ (N \cdot m/(rad/s))$	0.0130
$C_v\ (N)$	1.9068
$C_{vn}\ (N)$	2.0423
$C_\omega\ (N \cdot m)$	0.0971

2.4.1 Model Validation

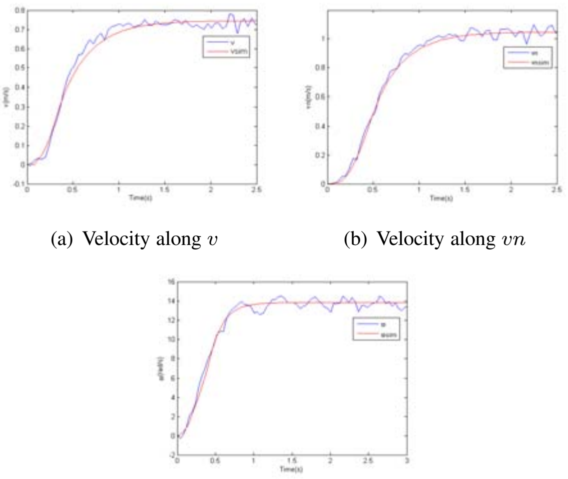
The simulator dynamics is based in the ODE (Open Dynamics Engine). ODE is a platform independent C++ library for simulating articulated rigid body dynamics, ground vehicles, legged creatures, or moving objects. It also supports advanced joints, contact with friction, and built-in collision detection. ODE is Free Software licensed under the GNU LGPL [14].

The model was validated with experimental tests on using a step voltage with an initial acceleration ramp, as shown in Figure 4.

3 Sensors modeling

The Sharp family of Infra-Red range finders are very popular for robotics distance measurement applications. Some drawback of these sensors are their non-linear response and the mandatory minimum distance measurement requisites. The presented study is about the Sharp Infra-Red distance sensor GP2D120.

In this section it will be presented in the first place the experimental setup to acquire sensor data, then it will be presented the sensor modeling based on the data collected and finally it will be shown how this model was included in the robot simulator.



(a) Velocity along v

(b) Velocity along vn

(c) Angular velocity ω

Figure 4. Robot model validation experimental runs.

3.1 Experimental setup

In order to model the distance sensor it was necessary to collect a considerable amount of data for different distances, for this task it was used the industrial robot ABB IRB 1400, as shown in Figure 5. Industrial robots allow executing repetitive operations normally performed by human operators, without getting bored and without loosing precision. The introduction of an industrial robot to place the obstacle in different known positions allows to increase the speed, repeatability and reduces errors in the process of distance sensor data collecting.

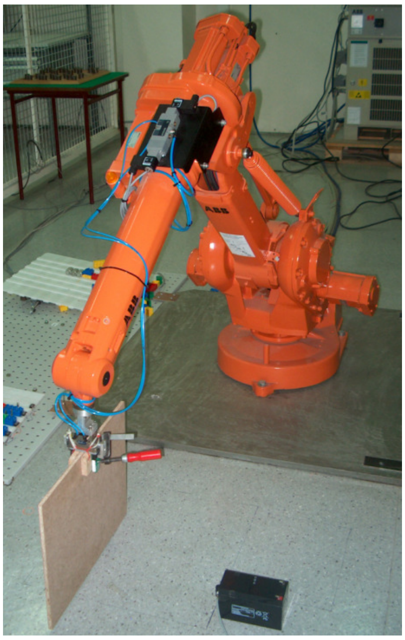


Figure 5. IRB 1400 placing the obstacle.

The robot interface is built upon the *RobComm* ActiveX provided by ABB [2]. This interface uses the following primitive to access the robot controller:

- `S4ProgramNumVarWrite`- Allows to write to a register of the industrial robot program.

Above is shown, a fragment of the industrial robot program, written in *Rapid*, that allows the robot to move the obstacle to a destination position.

```
IF move=1 THEN
MoveL Offs(p10,0,-dy,0),vmax,fine,tool0;
move:=0;
ENDIF
```

The robot program contains 2 registers:

- `dy` - distance from the sensor to the obstacle.
- `move` - authorization to move the robot.

The indication of a request to execute a movement is given by the register 'move': if this register has the value 1 it means that a request has arrived then a robot movement is performed. Point p10 is the reference for a zero distance, being all the other movements made relatively to this point applying offsets in the y axis.

The sensor data is acquired using the internal analog to digital converter (ADC) of the Atmel AVR ATMega8 with 8 bit precision. At each mobile robot sample time the ADC registers 10 samples for each sensor, which are summed and sent to a personal computer. In order to evaluate the sensor noise it are registered 256 mobile robot sample times data for each distance.

3.2 Distance sensors modeling

As the used analog to digital converter was the provided internally with the micro-controller ATMega8, and since there is available an internal reference voltage of 2.56 V (V_{ref}), it is possible to have a precision increase using this reference, when compared to the alternative of using an external reference voltage of 5 V. To use this approach, a voltage divisor must be applied in order to lower the sensor voltages to values below the converter reference (2.56 V), since its maximum is nearly 3.2 V. This is important if the user wants to use sensor values from 7 to 10 cm. If the user makes the choice of using a minimal distance of 10 cm then it is not necessary to apply a voltage divisor because the value for 10 cm corresponds to nearly 2.33 V, which is below the internal converter reference, and the voltage decreases with the distance. For this application it was considered that it was important to use the sensor range from 7 to 100 cm, thus a voltage divisor was applied. The obtained voltage characteristic of the InfraRed distance sensor is presented in Figure 6. It was calculated resorting to equation 27, where v is the voltage, si is the *ith* sample, n is the number of acquired samples and V_{ref} is the micro-controller internal reference voltage.

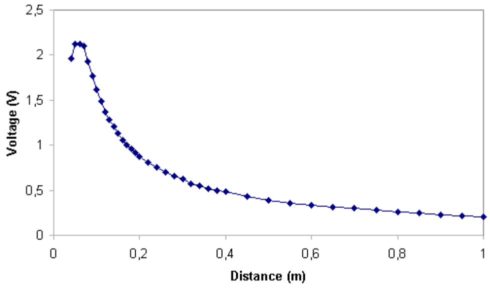


Figure 6. IR distance sensor characteristic.

$$v = V_{ref}(\frac{\sum si}{n.255}) \tag{27}$$

The relation between the inverse voltage and the distance can be approximated to a line as shown in Figure 7, where the real and the approximated curve are presented. The used values to achieve the presented curve were from 7 to 100 cm, taking in account the chosen sensor minimal distance.

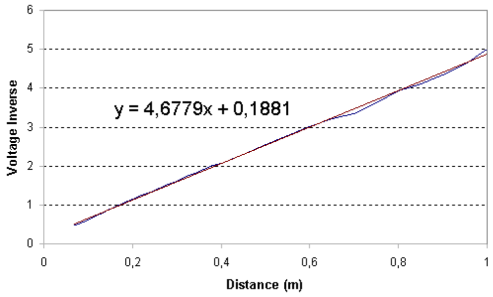


Figure 7. Voltage Inverse VS Distance.

In order to obtain the distance in the real robot, having in mind the shown approximation, equation 28 can be applied, where d is the distance expressed in m and v is the sensor voltage, k_1 equals 0.1881 and k_2 equals 4.6779.

$$d = \frac{\frac{1}{v} - k_1}{k_2} \tag{28}$$

One important information to extract from the sensor data is its variance, which expresses the confidence on a sensor measure. The voltage variance was approximated to a line using a linear regression and it is possible to observe that it increases with the distance, as shown in Figure 8.

In order to obtain the distance variance it was made the approximation shown in equation 29, with a different derivative for each distance, where m is the voltage derivative presented in equation 30 and in Figure 9. The voltage equation was obtained from the approximation presented in equation 28.

$$v = m.d + b \tag{29}$$

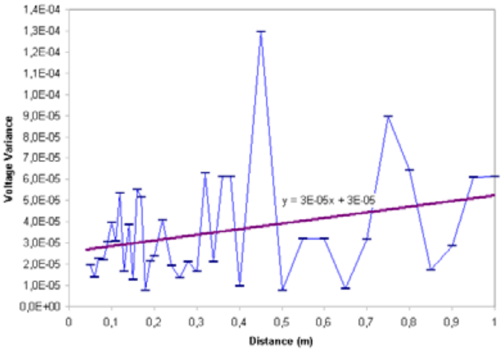


Figure 8. Voltage variance.

$$m = \frac{-k_2}{(d.k_2 + k_1)^2} \tag{30}$$

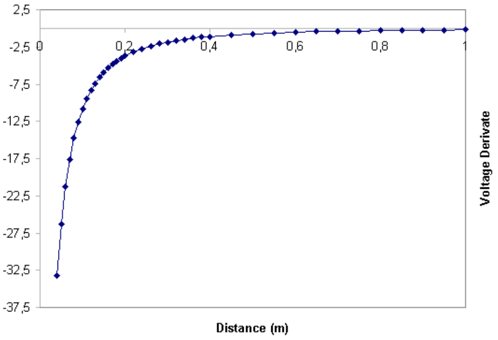


Figure 9. Voltage derivative.

This approximation was made in order to obtain the distance variance related with the known voltage variance, as presented in the next equation:

$$Var(d) = \frac{Var(v)}{m^2} \tag{31}$$

The simulated Infra-Red distance sensors provide the distance with the noise. Its variance is shown in Figure 10, which was obtained resorting to equation 31.

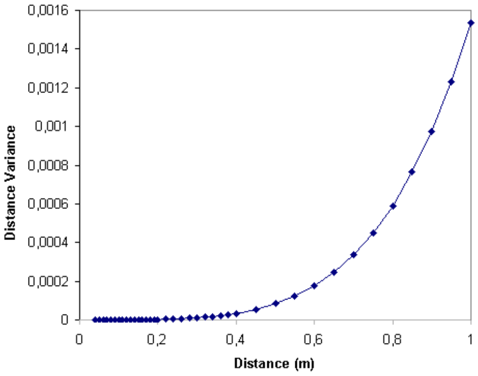


Figure 10. Simulated sensors variance.

4 Conclusions and future work

This paper described the sensor and actuator modeling of a realistic wheeled mobile robot simulator. The purpose of such simulator is to allow the production of robot software that can be migrated to real world systems with minimal overhead. As future work the authors intend to migrate robot software developed with the simulator to a real world system.

References

[1] *Webots^{TM}*. http://www.cyberbotics.com/, 2008.

[2] ABB. *RobComm Users Guide*. ABB Flexible Automation inc, 1996.

[3] A. K. an Thomas Braunl. Mobile robot simulation with realistic error models. In *Proceedings of the 2nd International Conference on Autonomous Robots and Agents, Palmerston North, New Zealand*, 2004.

[4] K. J. Astrom and B. Wittenmark. *Computer Controlled System Theory and Desing*. Prentice Hall, Information and System Sciences Series, 1984.

[5] Borestein, Everett, and Feng. *where am I, Sensores and Methods for Mobile Robot Positioning*. Prepared by the University of Michigan, 1996.

[6] G. Campion, G. Bastin, and B. Dandrea-Novel. Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE Transactions on Robotics and Automation*, 12(1):47–62, 1996. 1042-296X.

[7] A. S. Conceição, A. P. Moreira, and P. J. Costa. Model identification of a four wheeled omni-directional mobile robot. In *Controlo 2006, 7th Portuguese Conference on Automatic Control*, Instituto Superior Técnico, Lisboa, Portugal, 2006.

[8] P. Costa, P. Marques, A. Moreira, A. Sousa, and P. Costa. *Tracking and Identifying in Real Time the Robots of a F-180 Team*. Springer, 1999.

[9] J. Gonçalves, P. Pinheiro, J. Lima, and P. Costa. Tutorial introdutório para as competições de futebol robótico. *IEEE RITA - Revista Iberoamericana de tecnologias da aprendizagem*, 2(2):63–72, November 2007.

[10] T. Kalmar-Nagy, R. D'Andrea, and P. Ganguly. Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle. Sibley School of Mechanical and Aerospace Engineering Cornell University Ithaca, NY 14853, USA, April 8 2002.

[11] P. K. Khosla. Categorization of parameters in the dynamic robot model. *IEEE Transactions on Robotics and Automation*, 5(3):261–268, 1989. 1042-296X.

[12] O. Michel. Webots^{TM}: Professional mobile robot simulation, issn 1729-8806. *International Journal of Advanced Robotic Systems*, 1(1):39–42, November 2004.

[13] P. Pillay and R. Krishnan. Modeling, simulation, and analysis of permanent-magnet motor drives, part 11: The brushless dc motor drive. *IEEE transactions on Industry applications*, 25(2):274–279, 1989.

[14] R. Smith. *Open Dynamics Engine*. http://www.ode.org, 2008.

[15] I. Williams, R. L., B. E. Carter, P. Gallina, and G. Rosati. Dynamic model with slip for wheeled omnidirectional robots. *IEEE Transactions on Robotics and Automation*, 18(3):285–293, 2002. 1042-296X.