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™, 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.
Figure 2. Omnidirectional robot prototype.

Figure 3. Three wheel Omnidirectional robot.

Equation 1 allows to convert the linear velocities on the robot local axis.

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

\[
X_R = \begin{bmatrix} \cos(\theta(t)) & \sin(\theta(t)) & 0 \\ -\sin(\theta(t)) & \cos(\theta(t)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot X_0 \quad (1)
\]

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

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

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:

\[
v(t) = \left(\sqrt{3}/3\right) \cdot (v_2(t) - v_0(t)) \quad (3)
\]

\[
v_n(t) = \left(1/3\right) \cdot (v_2(t) + v_0(t)) - (2/3) \cdot v_1(t) \quad (4)
\]

\[
\omega(t) = \left(1/(3 \cdot d)\right) \cdot (v_0(t) + v_1(t) + v_2(t)) \quad (5)
\]

2.2.2 Dynamics

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

\[
M \cdot \frac{dv(t)}{dt} = \sum F_v(t) - F_{Bv}(t) - F_{Cv}(t) \quad (6)
\]

\[
M \cdot \frac{dv_n(t)}{dt} = \sum F_{vn}(t) - F_{Bvn}(t) - F_{Cvn}(t) \quad (7)
\]

\[
J \cdot \frac{d\omega(t)}{dt} = \sum T(t) - T_{B\omega}(t) - T_{C\omega}(t) \quad (8)
\]

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 \( v_n \);
- \( T_{B\omega}, T_{Bvn} \) [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 \( v_n \);
- \( T_{C\omega}, T_{Cvn} \) [N \cdot m] - Coulomb friction torque with respect to robot’s rotation axis.

Viscous friction forces are proportional to the robot’s velocities, \( F_{Bv}(t) = B_v \cdot v(t), F_{Bvn}(t) = B_{vn} \cdot v_n(t) \) and \( T_{B\omega}(t) = B_\omega \cdot \omega(t) \), where \( B_v, B_{vn}, B_\omega \) \{N/m/s\} are the viscous friction coefficients for directions \( v, v_n \) and \( \omega \) \{N \cdot m/(rad/s)\} is the viscous friction coefficient for \( \omega \).
The Coulomb friction forces are constant in amplitude $F_{Cv}(t) = C_{v} \cdot \text{sign}(v(t))$, $F_{Cvn}(t) = C_{vn} \cdot \text{sign}(vn(t))$ and $T_{Cw}(t) = C_{w} \cdot \text{sign}(w(t))$, where $C_{v}, C_{vn} [N]$ are Coulomb friction coefficient for directions $v$, $vn$ and $C_{w} [N \cdot m]$ is the Coulomb friction coefficient for $w$.

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

$$\sum F_{v}(t) = (f_{2}(t) - f_{0}(t)) \cdot \sin(\pi/3) \quad (9)$$

$$\sum F_{vn}(t) = -f_{1}(t) + (f_{2}(t) + f_{0}(t)) \cdot \cos(\pi/3) \quad (10)$$

$$\sum T(t) = (f_{0}(t) + f_{1}(t) + f_{2}(t)) \cdot d \quad (11)$$

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

$$f_{j}(t) = T_{j}(t)/r \quad (12)$$

$$T_{j}(t) = l \cdot K_{i} \cdot i_{j}(t) \quad (13)$$

- $l$ - Gearbox reduction;
- $r$ [\text{m}] - Wheel radius;
- $K_{i} [N \cdot m/A]$ - Motor torque constant;
- $i_{j}$ [A] - Motor current (\text{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]:

$$u_{j}(t) = L \cdot \frac{di_{j}(t)}{dt} + R \cdot i_{j}(t) + K_{e} \cdot \omega_{mj}(t) \quad (14)$$

$$T_{mj}(t) = K_{i} \cdot i_{j}(t) \quad (15)$$

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

$$u_{j}(t) = R \cdot i_{j}(t) + K_{e} \cdot \omega_{mj}(t) \quad (16)$$

### 2.3 Robot Model

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

$$(dx(t)/dt) = A \cdot x(t) + B \cdot u(t) + K \cdot \text{sign}(x(17))$$

$$x(t) = [v(t) \ \ vn(t) \ \ w(t)]^{T} \quad (18)$$

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

$$A = \begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & A_{33} \end{bmatrix} \quad (19)$$

$$A_{11} = -\frac{3 \cdot K_{t}^{2} \cdot l^{2}}{2 \cdot r^{2} \cdot R \cdot M} - \frac{B_{vn}}{M}$$

$$A_{22} = -\frac{3 \cdot K_{t}^{2} \cdot l^{2}}{2 \cdot r^{2} \cdot R \cdot M} - \frac{B_{vn}}{M}$$

$$A_{33} = -\frac{3 \cdot d^{2} \cdot K_{t}^{2} \cdot l^{2}}{r^{2} \cdot R \cdot J} - \frac{B_{vn}}{J} \quad (20)$$

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

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

### 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 ($B_{v}, B_{vn}, B_{w}$), the Coulomb friction coefficients ($C_{v}, C_{vn}, C_{w}$) 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 $\theta$ is estimated using:

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

$$x = [x_{1}(1) \ldots x_{1}(n) \ x_{2}(1) \ldots x_{2}(n) \ x_{3}(1) \ldots x_{3}(n)]^{T} \quad (24)$$
Estimated parameters can be skewed and for this reason instrumental variables are used to minimize the error, with state vector defined as:

\[ z = [x_1(1), \ldots, x_1(n), x_2(1), \ldots, x_2(n), x_3(1), \ldots, x_3(n)]^T \]  (25)

The parameter \( \theta \) is now calculated by:

\[ \theta = (z^T \cdot x)^{-1} \cdot z^T \cdot y \]  (26)

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

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d ) (m)</td>
<td>0.089</td>
</tr>
<tr>
<td>( r ) (m)</td>
<td>0.0325</td>
</tr>
<tr>
<td>( I )</td>
<td>5</td>
</tr>
<tr>
<td>( K_v ) (V/(rad/s))</td>
<td>0.0259</td>
</tr>
<tr>
<td>( R ) (Ω)</td>
<td>4.3111</td>
</tr>
<tr>
<td>( M ) (kg)</td>
<td>1.944</td>
</tr>
<tr>
<td>( J ) (kg \cdot m^2)</td>
<td>0.0169</td>
</tr>
<tr>
<td>( B_v ) (N/(m/s))</td>
<td>0.5082</td>
</tr>
<tr>
<td>( B_{vn} ) (N/(m/s))</td>
<td>0.4970</td>
</tr>
<tr>
<td>( B_\omega ) (N \cdot m/(rad/s))</td>
<td>0.0130</td>
</tr>
<tr>
<td>( C_v ) (N)</td>
<td>1.9068</td>
</tr>
<tr>
<td>( C_{vn} ) (N)</td>
<td>2.0423</td>
</tr>
<tr>
<td>( C_\omega ) (N \cdot m)</td>
<td>0.0971</td>
</tr>
</tbody>
</table>

### 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.

![Figure 4. Robot model validation experimental runs.](image1)

#### 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 losing 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.](image2)
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

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.

\[
v = V_{ref} \left( \frac{\sum s_i}{n.255} \right)
\]  

(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.

\[
y = 4.6779x + 0.1881
\]  

(28)

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.

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.

\[
v = m.d + b
\]  

(29)
This approximation was made in order to obtain the distance variance related with the known voltage variance, as presented in the next equation:

\[ \text{Var}(d) = \frac{\text{Var}(v)}{m^2} \]  

(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.

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