TRAJECTORY CONTROL AND MODELLING OF AN OMNI-DIRECTIONAL MOBILE ROBOT

André Scolari Conceição, A. Paulo Moreira, Paulo J. Costa
Department of Electrical and Computer Engineering
University of Porto - Porto - Portugal.

Keywords: Trajectory control, modelling and simulation, omni-directional mobile robot.

Abstract: This paper presents a trajectory controller for an omni-directional mobile robot. The controller presents important features, as the possibility of defining different translation velocities and angular positions to the robot during the trajectory following. The parameters of the controller are optimized based on trajectory following simulations, with the mobile robot model. Simulation and real results of trajectory following are presented.

1 INTRODUCTION

Omni-directional mobile robots have the ability to move simultaneously and independently in translation and rotation (Pin and Killough, 1994). However, non-linearities, like motor dynamic constraints, and other characteristics like friction, inertia moment and mass of the robot, should be modelled, because can greatly affect the robot behaviour. Hence, dynamic modelling of mobile robots is very important to design of controllers, as in (Liu et al., 2003)(Watanabe, 1998)(Fraga et al., 2005), mainly when the robots need to follow trajectories at higher velocity, with sudden change in its direction and orientation.

The suggested controller presents interesting features to follows the path correctly, how the possibility to define different linear velocities and angular positions to the robot during the trajectory following. A trajectory can be approximated with line segments. A line segment has two distinct endpoints. We use in this paper the name of a line segment with endpoints \( A \) and \( B \) as "line segment \( \overline{AB} \)". Hence, we can define linear velocities and angular position to the robot in each endpoint of the line segment, moreover we can adjust its velocities and angular position to the long of the line segment. Another feature of the controller is low computational time, which is essential in real time applications.

The optimization of the parameters of the controller is based on robot model. Due to values of time and errors of position and orientation of the robot, in trajectory following simulations with the robot model, we can calculate the best parameters to the controller.

We focus attention on a omni-directional mobile robot with four motors, as shown in Fig.1, built for the 5dpo Robotic Soccer team from the Department of Electrical and Computer Engineering at the University of Porto at Porto, Portugal (Moreira et al., 1999).

The organization of the paper is as follows. In section 2, the omni-directional mobile robot model is developed. The controller for trajectory following is presented in section 3. In section 4, the optimization of the controller parameters, simulation results and real results are presented. Finally, the conclusion is drawn in section 5.

2 THE MOBILE ROBOT MODEL

The omni-directional mobile robot model is developed based on the dynamics, kinematics and DC motors of the robot.

The World frame \((X, Y, \theta)\), the robot’s body frame and the geometric parameters is shown in Fig. 2. The following symbols, in SI unit system, are used to modelling:

- \( b \ [m] \rightarrow \) distance between the point P(center of chassis) and robot’s wheels
- \( M \ [kg] \rightarrow \) robot mass
- \( r \ [m] \rightarrow \) wheel radius
- \( l \rightarrow \) motor reduction
- \( J \ [kg.m^2] \rightarrow \) robot inertia moment
2.1 Robot Dynamics

By Newton’s law of motion and the robot’s body frame, in Fig. 2, we have

\[ F_v(t) = M \frac{dV(t)}{dt} + B_v V(t) + C_v \text{sgn}(V(t)) \]  
\[ F_{vn}(t) = M \frac{dV_n(t)}{dt} + B_{vn} V_n(t) + C_{vn} \text{sgn}(V_n(t)) \]  
\[ \Gamma(t) = J \frac{dW(t)}{dt} + B_w W(t) + C_w \text{sgn}(W(t)) \]

where, \( \text{sgn}(\alpha) = \begin{cases} 1, & \alpha > 0, \\ 0, & \alpha = 0, \\ -1, & \alpha < 0. \end{cases} \)

The relationships between the robot’s traction forces and the wheel’s traction forces are,

\[ F_v(t) = f_4(t) - f_2(t) \]  
\[ F_{vn}(t) = f_1(t) - f_3(t) \]  
\[ \Gamma(t) = (f_1(t) + f_2(t) + f_3(t) + f_4(t))b \]  

The wheel’s traction force \((f)\) and the wheel’s torque \((T)\), for each DC motor, is as follow:

\[ f(t) = \frac{T(t)}{r} \]  
\[ T(t) = lK_\theta i_a(t) \]

where \(i_a(t)\) is the armature current and \(K_\theta\) is motor torque constant. The dynamics of each DC motor can be described using the following equations,

\[ u(t) = L_a \frac{di_a(t)}{dt} + R_a i_a(t) + K_v w_m(t) \]  
\[ T(t) = K_i i_a(t) \]

where \(L_a\) is the armature inductance, \(R_a\) is the armature resistance, \(u(t)\) is the applied armature voltage, \(w_m(t)\) is the rotor angular velocity in rad/sec, and \(K_i\) is the emf constant.

2.2 Robot Kinematics

By geometric parameters of the robot and the robot’s body frame, in Fig. 2, is possible to derive the motion equations,

\[ \frac{dx(t)}{dt} = V(t) \cos(\theta(t)) - V_n(t) \sin(\theta(t)) \]  
\[ \frac{dy(t)}{dt} = V(t) \sin(\theta(t)) + V_n(t) \cos(\theta(t)) \]  
\[ \frac{d\theta(t)}{dt} = W(t) \]

The relationships between wheel’s linear velocities \((v_1, v_2, v_3\) and \(v_4)\) and robot velocities \((V, V_n\) and \(W)\)
Where \( x(t) \) and \( y(t) \) is the localization of the point \( P \), and \( \theta(t) \) the orientation angle of the robot.

### 3 LINE SEGMENT CONTROLLER

The proposed controller adjusts the position and orientation of the robot to follow a line segment, defined in the plane \( XY \), as shown in Fig.3. From a line segment and the position of the robot, we can define the velocity vectors to the robot. The velocity vectors, robot position(\( P \)) and a line segment(\( AB \)) are, as shown in Fig.3. The robot position is \( P(x_r, y_r) \) and \( \theta \) is the orientation angle of the robot in the plane \( XY \), hence the robot can follow the orientation angle of the robot.

#### Figure 3: Schematic of the controller.

![Schematic of the controller](image)

The velocity vectors \( V \) and \( V_n \) are perpendicular, and represent the linear velocities of the robot. The angular velocity of the robot is % of the line segment angle (\( \alpha \)) and the robot angle (\( \theta \)):

\[
\varphi = \alpha - \theta. \quad (13)
\]

The velocity vector \( v_r \) is the desired linear velocity to robot, called too reference velocity. The velocity \( v_r \) can receive different values in both points\( (A \) and \( B \) of the line segment, hence the robot can follow trajectories with different reference velocities. The reference velocity to the long of the line segment is:

\[
v_r = v_{r1}(1-d) + v_{r2}d. \quad (14)
\]

Where \( v_{r1} \) is the reference velocity of the point \( A \) and \( v_{r2} \) is the reference velocity of the point \( B \), of the line segment \( AB \). The variable \( d \) is the projection from robot position(\( P(x, y) \)) to line segment \( AB \), it is normalized to length of the line segment. The vector velocities of the controller, are as follow:

\[
v_c = e_s k_1, \quad (15)
\]

\[
v_a = \begin{cases} 0 & v_r^2 - v_c^2 < 0, \\ \sqrt{v_r^2 - v_c^2}, & v_r^2 - v_c^2 > 0. \end{cases} \quad (16)
\]

With the vector \( v_a \) parallel to \( AB(v_a \parallel AB) \) and the vector \( v_c \) perpendicular to \( AB(v_c \perp AB) \). The distance between the robot and the line segment is \( e_s \), and \( k_1 \) is a gain. We can calculate the vector velocities \( V \) and \( V_n \) using a rotation matrix:

\[
\begin{bmatrix} V \\ Vn \end{bmatrix} = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix} \begin{bmatrix} v_a \\ v_c \end{bmatrix}
\]

The robot angular velocity(\( W \)) is calculated based on robot angular position(\( \theta \)) and desired angular positions(\( \theta_d \)) in both points\( (A \) and \( B \) of the line segment. For all line segment, the angular velocity \( W \) is calculated with a similar variation used in equation 14, in function of \( d \). Therefore, when is defined the line segment \( AB \), we define the desired angular positions(\( \theta_1 \) and \( \theta_2 \)) in each point of the line segment. The controller to robot angular position is defined as follow:

\[
W = e_\theta k_2. \quad (17)
\]

With:

\[
e_\theta = \theta_{ref} - \theta, \quad (18)
\]

\[
\theta_{ref} = \phi_1(1 - d) + \phi_2 d. \quad (19)
\]

where \( k_2 \) is a gain, \( e_\theta \) is the error between desired angular position(\( \theta_{ref} \)) and robot angular position(\( \theta \)).

### 4 OPTIMIZATION OF THE CONTROLLER

After we define the controller structure, we need to choose the appropriate values to gain \( k_1 \) and gain \( k_2 \). A cost function (\( C \)) was created to measure the performance of trajectory following. The cost function is described as follow:

\[
C(k_1, k_2) = E_d P_d + E_a P_a + (T_r - T_i) P_t.
\]

Where:

\[
E_d \leftrightarrow \text{Mean square error (MSE) related to } e_s, \text{ for all trajectory following;}
\]

\[
E_a \leftrightarrow \text{Mean square error related to } e_{\theta}, \text{ for all trajectory following;}
\]

\[
T_i \leftrightarrow \text{ideal time to follow the trajectory;}
\]

\[
T_r \leftrightarrow \text{robot time to follow the trajectory;}
\]
\( P_d, P_a, P_t \rightarrow \) gain related to errors.

We used the robot model, described in section 2, to define \( k_1 \) and \( k_2 \) gains. Trajectory following simulations, see Fig. 4, with different values of the \( k_1 \) and \( k_2 \), make possible obtain the values of the cost function. The objective is found the values of the \( k_1 \) and \( k_2 \) that result the minimum value of the cost function.

Through real experiments with the robot, we know that gains \( (k_1 \) and \( k_2) \) above of 15 do not cause effect in trajectory following, due to saturation of the actuators. Hence, we used values between 0 and 20, with resolution of 0.5 in trajectory following simulations.

The trajectory following simulations were made for 3 values of the linear velocity \( v_r \): 1, 0.7 e 0.4 \([m/s]\). So, it computed 1600 simulations for each velocity, resulting in 4800 total simulations. Without the robot model, it will be impossible. Furthermore, the total simulation time for each velocity(1600 simulations) is about 10 seconds, therefore we did not need to use minimization algorithms.

The trajectory used in the simulations has special features, as sudden change of direction and orientation to the robot, in order to test the controller in hard condition. The Fig. 4 shows the trajectory used in the simulations, with the points\((x,y)\). The Fig. 5 shows values of the desired angular positions\((\phi)\) for each point\((x_i,y_i)\), \(i=1,...,13\) of the trajectory.

The cost functions for 3 linear velocities \( v_r \) and the gains \( (k_1 \) and \( k_2) \) are shown in Figs. 6, 7 and 8. The minimum values of the cost function\((C)\) and corresponds gains are shown in table 1. The values of

<table>
<thead>
<tr>
<th>( v_r [m/s] )</th>
<th>( C )</th>
<th>( k_1 )</th>
<th>( k_2 )</th>
<th>( P_d )</th>
<th>( P_a )</th>
<th>( P_t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.57</td>
<td>8</td>
<td>12.5</td>
<td>1000</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>0.7</td>
<td>3.87</td>
<td>7</td>
<td>10.5</td>
<td>1000</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>0.4</td>
<td>1.21</td>
<td>7.5</td>
<td>9</td>
<td>1000</td>
<td>10</td>
<td>0.5</td>
</tr>
<tr>
<td>mean(\approx)</td>
<td></td>
<td>7.5</td>
<td>10.5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4: Trajectory to calculate the cost function.

Figure 5: Angular positions\((\phi)\) of the trajectory.

Figure 6: Cost function, \( v_r = 1 \) m/s.

Figure 7: Cost function, \( v_r = 0.7 \) m/s.

The gains \( P_d, P_a \) and \( P_t \) were defined in order to be even the values of the errors \( E_d, E_a \) and \( (T_r - T_i) \). Currently we use the mean of the gains values, shown in table 1. We tested this values in robot trajectory following for the linear velocities \( v_r = 0.4, 0.7, 1[m/s] \). It had a good performance so, the use of different gains for different linear velocities is not necessary. Real and simulated results with gain \( k_1 = 7.5 \), gain
Figure 8: Cost function, \( v_r = 0.4 \, m/s \).

\( k_2 = 10.5 \) and linear velocity \( v_r = 1\,[m/s] \) are shown if Figs. 9, 10 and 11.

Figure 9: Simulated and real trajectory.

The procedure to define the gains \( k_1 \) and \( k_2 \) was repeated with a trajectory like "8", see Fig. 12(a), this trajectory has a different feature than first one, in Fig. 4. The robot does not need to do sudden change of direction and orientation. The table 2 shows the minimum values of the cost function(\( C \)) and corresponding gains. In this simulations we kept the same gains to \( P_d, P_a \) and \( P_t \), to get a comparison. The results for the gain \( k_2 \) were equal in both procedures. The results for the gain \( k_1 \) were not equal, the gain diminished with the reduction of the linear velocity \( v_r \). It happened due to characteristics of the trajectory "8", this trajectory is more soft than the first trajectory, consequently the error \( E_d \) diminished too.

Finally, we tested the gains \( k_1 \) and \( k_2 \) obtained with the first trajectory, in the robot to follow the trajectory "8", the robot had a satisfactory performance. We decided to use the bigger gains (\( k_1 = 7.5 \) and \( k_2 = 10.5 \)), because in robotic soccer application the mobile robot needs to execute trajectories quickly and with a perfect position to the objective, for example, positioning to the ball, or to the goal, or to avoid dynamic obstacles.

Table 2: Minimum values of the cost function, trajectory "8".

<table>
<thead>
<tr>
<th>( v_r ,[m/s] )</th>
<th>( C )</th>
<th>( k_1 )</th>
<th>( k_2 )</th>
<th>( P_d )</th>
<th>( P_a )</th>
<th>( P_t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.84</td>
<td>9</td>
<td>12.5</td>
<td>1000</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>0.7</td>
<td>0.72</td>
<td>5</td>
<td>10.5</td>
<td>1000</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>0.4</td>
<td>0.22</td>
<td>3</td>
<td>9</td>
<td>1000</td>
<td>10</td>
<td>0.5</td>
</tr>
<tr>
<td>mean%</td>
<td>-</td>
<td>5.5</td>
<td>10.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

5 CONCLUSION

In this paper, a controller for trajectory following has been presented. The proposed controller presents important features, as the possibility of defining different translation velocities and angular positions to the robot during the trajectory following. Besides, it does not demand a high computational time, which is essential in real time applications and applications of high velocity. The robot model was essential to optimization of the parameters of the controller.
Figure 12: Trajectory "8".

REFERENCES


