A Mobile Manipulator Controller Implemented in the Robot Operating System

Taiser Tadeu Teixeira Barros
SENAI-RS, Santa Cruz, RS, Brazil, taiser.barros@senairs.org.br

Walter Fetter Lages
Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brazil, fetter@ece.ufrgs.br

Abstract

This work presents the development of mathematical model of a mobile manipulator and its use to develop a controller based on backstepping nonlinear control and the computed torque control law. This MIMO nonlinear controller was implemented on the Robot Operating System (ROS) environment. The mobile robot is based on the Barrett WAM (Whole Arm Manipulator) and a custom build mobile platform. The dynamic modeling is used to ensure better performance of the controller.

1 Introduction

A mobile manipulator is an assemble of two mainly components — a manipulator and a mobile base (a mobile robot in fact) — and most of the research in motion control of mobile manipulator consider its kinematics and dynamics [20]. The coupling between the manipulator and the mobile base allows us to benefit from whole structure mobility and therefore operate in a wider workspace if compared with a conventional manipulator. Due to that, mobile manipulators had become an important research topics [32] where the integration between the mobile manipulator and its motion control has the main focus [10] [31].

Recent papers [13] [12] show that mobile manipulation is a wide field of research and one of the principal directions is trying to control all the mobile manipulator as a single device (Whole Body Control), even though considering the manipulator and the mobile robot as separated tasks by ignoring the structure interaction with the environment, enables an easier control development [31].

As presented in [8] the mobile manipulation is a very complex task, the control of a fifty one degrees of freedom robot is proposed considering constraints like self collision avoidance and environment objects collision avoidance. Some control laws proposed to the mobile manipulators can be found as in [21].

1.1 Mobile Platform Mathematical Model

Figure 1 shows the coordinate systems used to describe the mobile platform model, where X_{c_1} and X_{c_2} are the axes of the robot and X_1 and X_2 form the inertial coordinate system.

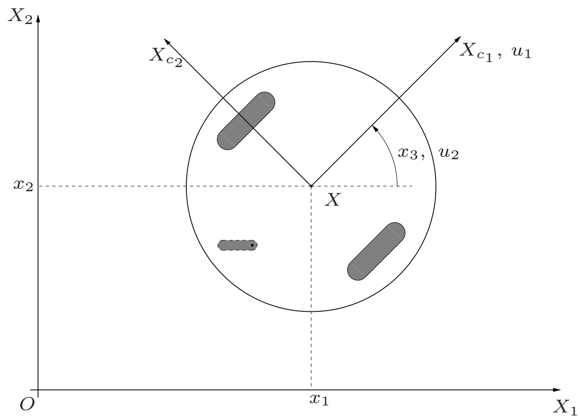

Figure 1: Differential-drive mobile robot coordinates.

The pose (position and orientation) of the platform is represented by ${}^0\xi_c = \begin{bmatrix} x_c & y_c & \theta_c \end{bmatrix}^T$ and is related with the reference frame in the robot by:

$$ {}^c\dot{\xi}_c = {}^cR_0 {}^0\dot{\xi}_c \tag{1}$$

where ${}^cR_0$ is the rotation matrix relating the orientation of the robot and the reference frame given by:

$$ {}^0R_c = \begin{bmatrix} \cos\theta_c & -\sin\theta_c & 0 \\ \sin\theta_c & \cos\theta_c & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The mathematical model for the mobile platform can be obtained based on Lagrange-Euler formulation and it allows the modeling a robot with any number of fixed wheels and castor wheels. Assuming the motion in horizontal plane, the potential energy can be neglected and the kinetic energy of the mobile platform can be ex-

ISBN 978-3-8007-3601-0

pressed by

$$T = \frac{1}{2}{}^0\dot{\xi}_c^{T\,c}R_0^T\left[M\left(\beta_o\right){}^cR_0{}^0\dot{\xi}_c + 2V\left(\beta_o\right)\dot{\beta}_o\right] + \frac{1}{2}\dot{\beta}_o^{\;T}I_\beta\dot{\beta}_o + \frac{1}{2}\dot{\varphi}^TI_\varphi\dot{\varphi} \quad (3)$$

where T is kinetic energy, β_0 is the angle of the castor wheel, $M(\beta_0)$ is the matrix of inertia of the platform, $V(\beta_0)$ is a vector representing the Coriolis and Centrifugal terms, I_β is the inertia moment around the vertical axis of the castor wheel, $\dot{\varphi}$ is vector of the angular speed of the wheels and I_φ is a diagonal matrix with inertia moments of the wheels around their rotating axis.
By applying the Lagrange-Euler formulation:

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\xi}}\right) - \frac{\partial L}{\partial\xi}\right)^T = {}^0R_cJ_1^T\left(\beta_c,\beta_{nc}\right)\lambda + {}^0R_cC_1^T\left(\beta_c,\beta_{nc}\right)\mu$$

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\beta}_{nc}}\right) - \frac{\partial L}{\partial\beta_{nc}}\right)^T = C_2^T\mu + \tau_{nc} \quad (4)$$

$$\left(\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\varphi}}\right) - \frac{\partial L}{\partial\varphi}\right)^T = J_2^T\lambda + \tau_\varphi \quad (5)$$

with $L = T - P$, and $J_1\left(\beta_c,\beta_{nc}\right)$, J_2, $C_1\left(\beta_c,\beta_{nc}\right)$, C_2 satisfying the restrictions of motion of the wheels [5, 27] given by:

$$J_1\left(\beta_c,\beta_{nc}\right){}^cR_0{}^0\dot{\xi}_c + J_2\dot{\varphi} = 0 \quad (6)$$

$$C_1\left(\beta_c,\beta_{nc}\right){}^cR_0{}^0\dot{\xi}_c + C_2\dot{\beta}_{nc} = 0 \quad (7)$$

Note that since the platform is moving in a horizontal plane its potential energy $P = 0$.
By performing the differentials and after some algebraic manipulations on (4-5), it is possible to obtain:

$$\begin{cases}\dot{q} = S\left(q\right)u \\ H\left(\beta\right)\dot{u} + f\left(\beta,u\right) = F\left(\beta\right)\tau\end{cases} \quad (8)$$

where $q = \begin{bmatrix}\xi & \beta_c & \beta_{nc} & \varphi\end{bmatrix}^T$, $u = \begin{bmatrix}v & \omega\end{bmatrix}^T$ is the vector of the linear and angular velocities of the platform, and τ is the input torque on the wheels.
By using the feedback:

$$\tau = F^\dagger\left(\beta\right)\left(H(\beta)v + f(\beta,u)\right) \quad (9)$$

where v is a new input vector, it is possible linearize the second equation of (8) to obtain:

$$\begin{cases}\dot{q} = S\left(q\right)u \\ \dot{u} = v\end{cases} \quad (10)$$

This last of equations represent the configuration dynamic model of the mobile platform. This model include some state variables which are only internal variables and their values are not relevant for the purpose of controlling the mobile platform. Hence, some components of q can be neglected. Hence, the pose dynamic model can be written as:

$$\begin{cases}\dot{x} = B(u) \\ \dot{u} = v\end{cases} \quad (11)$$

where $x = \xi = \begin{bmatrix}x_c & y_c & \theta_c\end{bmatrix}^T$ is the vector of pose coordinates and v is the input vector.
In this paper, the TWIL robot, shown in **Figure 2**, is used as a mobile platform. Twil is a differential driven robot for which:

$$B(x) = \begin{bmatrix}\cos\theta_c & 0 \\ \sin\theta_c & 0 \\ 0 & 1\end{bmatrix} \quad (12)$$

$$H(\beta) = I \quad (13)$$

$$f(\beta,u) = f(u) = -\begin{bmatrix}0 & K_5 \\ K_6 & 0\end{bmatrix}\begin{bmatrix}u_1u_2 \\ u_2^2\end{bmatrix} \quad (14)$$

$$F(\beta) = F = \begin{bmatrix}K_7 & K_7 \\ K_8 & -K_8\end{bmatrix} \quad (15)$$

where K_5, K_6, K_7 and K_8 are constants depending only on the geometric and inertia parameters of the platform.

Figure 2: Real Mobile Robot TWIL.

1.2 Manipulator Mathematical Model

The manipulator dynamic model is given by [9]:

$$\tau\left(t\right) = D\left(q\left(t\right)\right)\ddot{q}\left(t\right) + H\left(q\left(t\right),\dot{q}\left(t\right)\right) + G\left(q\left(t\right)\right) \quad (16)$$

where τ is the torque applied to the joints, q is the vector of positions of the joints, $D(q)$ is the inertia matrix, $H(q,\dot{q})$ is the vector of Coriolis and centrifugal forces and $G(q)$ is the vector of gravitational forces.
This model was developed based on the Barret WAM manipulator shown in **Figure 3**. Actually the model that represents the interaction between the mobile robot and the

ISBN 978-3-8007-3601-0

manipulator is in development and a control law using the impedance control concepts will be proposed.

Figure 3: Barrett WAM manipulator.

2 Control of the Mobile Platform

Differential-drive mobile robots are nonholonomic systems [5]. An important general statement on the control of nonholonomic systems has been made by Brockett [3], who has shown that it is not possible to asymptotically stabilize the system at an arbitrary point through a time-invariant, smooth state feedback law. In spite of it, the system is controllable [1].
Ways around Brockett's conditions for asymptotic stability are time-variant control [22, 29, 11, 25], non-smooth control [1, 28, 6] and hybrid control laws [19]. In this paper, we will obtain a set of possible input signals based on non-smooth control law which is obtained by a non-smooth coordinate transform. A general way of designing control laws for nonholonomic systems through non-smooth coordinate transform was presented by [1]. We have considered a mapping from the state space to the input space as presented by [17].

2.1 Offset to Origin

The mappings from the system state to the input space which are used for point stabilization are such that the state space origin is made asymptotically stable. If we represent the mapping as $g : X \to U$, $x \in X$ and $u \in U$, then the autonomous system

$$\dot{x} = f\left(x, g(x)\right) \tag{17}$$

where $f(x,u) = B(x)u$, is asymptotically stable at the origin by making $u = g(x)$.
However, it is of interest to stabilize the robot at any point x_r, which means any given position and orientation $\begin{bmatrix} x_{r_1} & x_{r_2} & x_{r_3} \end{bmatrix}^T$.This can be accomplished by the coordinate change $\bar{x}(x, x_r)$, obtained by setting a new reference frame $X_{r_1}X_{r_2}$ at the reference position $\begin{bmatrix} x_{r_1} & x_{r_2} \end{bmatrix}^T$ with an angle x_{r_3}, according to **Figure 4**. Thus, the coordinate change from X_1X_2 to $X_{r_1}X_{r_2}$ consists of a translation and a rotation of angle x_{r_3}. It is readily verified that $\bar{x}_3 = x_3 - x_{r_3}$. Therefore, the coordinate change $\bar{x}(\cdot,\cdot)$ is obtained by the transform

$$\bar{x} = \begin{bmatrix} R(x_{r_3}) & 0 \\ 0 & 1 \end{bmatrix} (x - x_r) \tag{18}$$

where $R(x_{r_3})$ is a 2-D rotation matrix, that is,

$$R(x_{r_3}) = \begin{bmatrix} \cos x_{r_3} & \sin x_{r_3} \\ -\sin x_{r_3} & \cos x_{r_3} \end{bmatrix}. \tag{19}$$

Hence, if the system $\dot{\bar{x}} = f\left(\bar{x}, g(\bar{x})\right)$ is stable at $\bar{x} = 0$, then $\dot{x} = f\left(x, g(x)\right)$ is stable at $x = 0$. Therefore, in order to stabilize the system at any arbitrary point x_r based on a control law g that leads the state to the origin, it suffices to use $g(\bar{x})$.

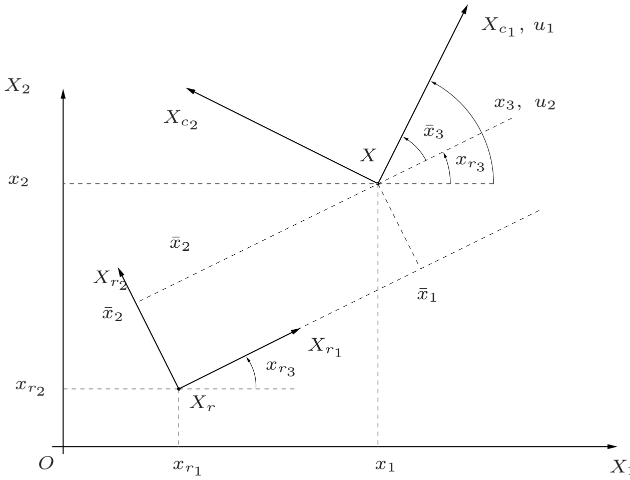

Figure 4: Robot coordinates with respect to the reference frame.

2.2 Non-Smooth Control

By considering a coordinate change [2, 17],

$$e = \sqrt{\bar{x}_1^2 + \bar{x}_2^2} \tag{20}$$

$$\psi = \operatorname{atan2}(\bar{x}_2,\, \bar{x}_1) \tag{21}$$

$$\alpha = \bar{x}_3 - \psi \tag{22}$$

$$\eta_1 = u_1 \tag{23}$$

$$\eta_2 = u_2 \tag{24}$$

the first expression of system model (11) can be rewritten as

$$\begin{cases} \dot{e} = \cos\alpha\eta_1 \\ \dot{\psi} = \frac{\sin\alpha}{e}\eta_1 \\ \dot{\alpha} = -\frac{\sin\alpha}{e}\eta_1 + \eta_2. \end{cases} \tag{25}$$

Then, given a candidate to Lyapunov function:

ISBN 978-3-8007-3601-0

$$V = \frac{1}{2}\left(\lambda_1 e^2 + \lambda_2 \alpha^2 + \lambda_3 \psi^2\right), \qquad (26)$$

it can be shown that the input signal

$$\eta_1 = -\gamma_1 e \cos\alpha \qquad (27)$$

$$\eta_2 = -\gamma_2 \alpha - \gamma_1 \cos\alpha \sin\alpha + \gamma_1 \frac{\lambda_3}{\lambda_2} \cos\alpha \frac{\sin\alpha}{\alpha}\psi \qquad (28)$$

with $\lambda_i > 0$, leads to

$$\dot{V} = -\gamma_1 \lambda_i e^2 \cos^2\alpha - \gamma_2 \lambda_2 \alpha^2 \leq 0 \qquad (29)$$

which, proves that V is indeed a Lyapunov function for (25) and the Barbalat's lemma [23], can be used to prove that (25) is asymptotically stable [17]. We note that even though the model (25) is discontinuous at the origin, due to e in the denominator, the closed loop system is not. The term in the denominator is canceled in closed loop because (27) contains e as a factor.

2.3 Backstepping

Although (27-28) are able to stabilize the first equation of (11), they can not stabilize (11) as whole since its input is v and not u. Note, however, that (11) can be seen as a cascade between two subsystems and in this case, it is possible to use a backstepping procedure [16] to obtain an expression for v from u.

By applying the transforms (18,20-22) to (11) it is possible to write:

$$\begin{aligned} \dot{e} &= \cos\alpha u_1 && (30)\\ \dot{\psi} &= \frac{\sin\alpha}{e} u_1 && (31)\\ \dot{\alpha} &= -\frac{\sin\alpha}{e} u_1 + u_2 && (32)\\ \dot{u}_1 &= v_1 && (33)\\ \dot{u}_2 &= v_2 && (34) \end{aligned}$$

Then, by adding $\cos\alpha(\eta_1 - \eta_1)$ to (30), $\frac{\sin\alpha}{e}(\eta_1 - \eta_1)$ to (31) and $-\frac{\sin\alpha}{e}(\eta_1 - \eta_1) + (\eta_2 - \eta_2)$ to (32), nothing is changed:

$$\begin{cases} \dot{e} = \cos\alpha u_1 + \cos\alpha(\eta_1 - \eta_1) \\ \dot{\psi} = \frac{\sin\alpha}{e} u_1 + \frac{\sin\alpha}{e}(\eta_1 - \eta_1) \\ \dot{\alpha} = -\frac{\sin\alpha}{e} u_1 + u_2 - \frac{\sin\alpha}{e}(\eta_1 - \eta_1) + (\eta_2 - \eta_2) \\ \dot{u}_1 = v_1 \\ \dot{u}_2 = v_2 \end{cases} \qquad (35)$$

which can be rearranged as:

$$\begin{cases} \dot{e} = \cos\alpha\eta_1 + \cos\alpha(u_1 - \eta_1) \\ \dot{\psi} = \frac{\sin\alpha}{e}\eta_1 + \frac{\sin\alpha}{e}(u_1 - \eta_1) \\ \dot{\alpha} = -\frac{\sin\alpha}{e}\eta_1 + \eta_2 - \frac{\sin\alpha}{e}(u_1 - \eta_1) + (u_2 - \eta_2) \\ \dot{u}_1 = v_1 \\ \dot{u}_2 = v_2 \end{cases} \qquad (36)$$

and by defining:

$$\begin{aligned} e_1 &\triangleq u_1 - \eta_1 && (37)\\ e_2 &\triangleq u_2 - \eta_2 && (38)\\ \bar{v}_1 &\triangleq v_1 - \dot{\eta}_1 && (39)\\ \bar{v}_2 &\triangleq v_2 - \dot{\eta}_2 && (40) \end{aligned}$$

results in:

$$\begin{cases} \dot{e} = \cos\alpha\eta_1 + \cos\alpha e_1 \\ \dot{\psi} = \frac{\sin\alpha}{e}\eta_1 + \frac{\sin\alpha}{e} e_1 \\ \dot{\alpha} = -\frac{\sin\alpha}{e}\eta_1 + \eta_2 - \frac{\sin\alpha}{e} e_1 + e_2 \\ \dot{e}_1 = \bar{v}_1 \\ \dot{e}_2 = \bar{v}_2 \end{cases} \qquad (41)$$

Then, by replacing η_1 and η_2 from (27) and (28):

$$\begin{cases} \dot{e} = -\gamma_1 e \cos\alpha^2 + \cos\alpha e_1 \\ \dot{\psi} = -\gamma_1 \sin\alpha\cos\alpha + \frac{\sin\alpha}{e} e_1 \\ \dot{\alpha} = -\gamma_2\alpha + \gamma_1 \frac{\lambda_3}{\lambda_2}\cos\alpha\frac{\sin\alpha}{\alpha}\psi - \frac{\sin\alpha}{e} e_1 + e_2 \\ \dot{e}_1 = \bar{v}_1 \\ \dot{e}_2 = \bar{v}_2 \end{cases} \qquad (42)$$

Let the following candidate to Lyapunov function:

$$V_1 = \frac{1}{2}\left(\lambda_1 e^2 + \lambda_2\alpha^2 + \lambda_3\psi^2 + \lambda_4 e_1^2 + \lambda_5 e_2^2\right) \qquad (43)$$

Its time derivative is given by:

$$\dot{V}_1 = \lambda_1 e\dot{e} + \lambda_2\alpha\dot{\alpha} + \lambda_3\psi\dot{\psi} + \lambda_4 e_1\dot{e}_1 + \lambda_5 e_2\dot{e}_2 \qquad (44)$$

which, by replacing the system equations from (42) gives:

$$\begin{aligned} \dot{V}_1 &= -\gamma_1\lambda_1 e^2\cos^2\alpha - \gamma_2\lambda_2\alpha^2 + \lambda_1 e\cos\alpha e_1 \\ &- \lambda_2\alpha\frac{\sin\alpha}{e} e_1 + \lambda_3\psi\frac{\sin\alpha}{e} e_1 \\ &+ \lambda_4 e_1\bar{v}_1 + \lambda_2\alpha e_2 + \lambda_5 e_2\bar{v}_2 \qquad (45) \end{aligned}$$

Then, by choosing:

ISBN 978-3-8007-3601-0

$$\bar{v}_1 \triangleq -\gamma_4 e_1 - \frac{\lambda_1}{\lambda_4}\cos\alpha + \frac{\lambda_2}{\lambda_4}\alpha\frac{\sin\alpha}{e} - \frac{\lambda_3}{\lambda_4}\psi\frac{\sin\alpha}{e} \quad (46)$$

$$\bar{v}_2 \triangleq -\gamma_5 e_2 - \frac{\lambda_2}{\lambda_5}\alpha \quad (47)$$

results in

$$\dot{V}_1 = -\gamma_1\lambda_1 e^2\cos^2\alpha - \gamma_2\lambda_2\alpha^2 - \gamma_4\lambda_4 e_1^2 - \gamma_5\lambda_5 e_2^2 \leq 0 \quad (48)$$

which proves that V_1 is indeed a Lyapunov function for the system (42). Furthermore, since $\dot{V}_1$ is uniformly continuous, it follows from the Barbalat's lemma [23] that $\dot{V}_1 \to 0$, which implies that $e \to 0$, $\alpha \to 0$, $e_1 \to 0$ and $e_2 \to 0$. It remains to prove that ϕ converges to zero. By applying the Barbalat's lemma to $\dot{\alpha}$ it follows that $\dot{\alpha} \to 0$ in (42), which implies that $\psi \to 0$.
The control law for the system (11) is then given by:

$$v_1 = \bar{v}_1 - \dot{\eta}_1 \quad (49)$$

$$v_2 = \bar{v}_2 - \dot{\eta}_2 \quad (50)$$

or

$$\begin{aligned} v_1 = & -\gamma_4 e_1 - \frac{\lambda_1}{\lambda_4}\cos\alpha + \frac{\lambda_2}{\lambda_4}\alpha\frac{\sin\alpha}{e} \\ & - \gamma_1^2 e\cos^3\alpha + \gamma_1\gamma_2 e\alpha\sin\alpha \\ & - \gamma_1^2\frac{\lambda_3}{\lambda_2}e\cos\alpha\frac{\sin^2\alpha}{\alpha}\psi \end{aligned} \quad (51)$$

$$\begin{aligned} v_2 = & -\gamma_5 e_2 - \frac{\lambda_2}{\lambda_5}\alpha \\ & - \gamma_2^2\alpha + \gamma_1\gamma_2\alpha\sin^2\alpha - \gamma_1^2\frac{\lambda_3}{\lambda_2}\cos\alpha\frac{\sin^3\alpha}{\alpha}\psi \\ & - \gamma_1\gamma_2\alpha\cos^2\alpha + \gamma_1^2\frac{\lambda_3}{\lambda_2}\cos^3\alpha\frac{\sin\alpha}{\alpha}\psi \\ & - \gamma_1\gamma_2\frac{\lambda_3}{\lambda_2}\sin^2\alpha\psi + \gamma_1^2\frac{\lambda_3^2}{\lambda_2^2}\cos\alpha\frac{\sin^3\alpha}{\alpha^2}\psi^2 \\ & + \gamma_1\gamma_2\frac{\lambda_3}{\lambda_2}\cos^2\alpha\psi - \gamma_1^2\frac{\lambda_3^2}{\lambda_2^2}\cos^3\alpha\frac{\sin\alpha}{\alpha^2}\psi^2 \\ & + \gamma_1^2\frac{\lambda_3^2}{\lambda_2^2}\cos^2\alpha\frac{\sin^2\alpha}{\alpha^3}\psi^2 \\ & + \gamma_1^2\frac{\lambda_3}{\lambda_2}\cos^2\alpha\frac{\sin^2\alpha}{\alpha} \end{aligned} \quad (52)$$

3 Control of the Manipulator

The classical computed torque control law [9] is used:

$$\begin{aligned} \tau = & D(q)\left[\ddot{q}_r + K_d(\dot{q}_r - \dot{q}) + K_p(q_r - q)\right] \\ & + H(q,\dot{q}) + G(q) \end{aligned} \quad (53)$$

where q_r is the reference vector for joint variables, K_d and K_p are the differential and proportional gains respectively.
By applying the control law (53) to the system (16) and defining $e = q_r - q$, it is possible to write the error equation:

$$\ddot{e} + K_d\dot{e} + K_p e = 0 \quad (54)$$

and it can be seen that by properly choosing the K_d and K_p gains, the error can be driven to zero.

4 Robot Operating System

The Robot Operating System [24] (ROS) is a message based system developed to integrate subsystems in a robotic system. It is composed by reusable C++ and Python libraries. The ROS philosophy is based on UNIX systems where a lot of tools are designed to work together and its origin is due to a partnership between industries and universities [7]. ROS can be understood as a distributed system [30], where nodes can communicate by publishing and/or subscribing to topics to send and/or receive messages.

4.1 Controllers in ROS

Unfortunately, ROS is not well suited for the implementation of advanced low-level controllers, such as proposed in this paper. In principle, ROS is not a real-time system, although it can show some real-time capability when running on a Linux system with the `PREEMPT_RT` kernel patch. Even then, there is a single real-time loop where all controllers are supposed to run under supervision of a controller manager. This real-time loop has a fixed rate of 1KHz, which, of course, is not adequate for all real-time tasks in a complex system. A common alternative for systems requiring many real-time tasks, with different rates, is to use OROCOS [4] as a lower-level layer to implement the real-time portion of the system.
From the control engineer point of view, there are some pitfalls while implementing a controllers in ROS. The first one in nomenclature. What ROS calls a controller is not necessarily what is called a controller in control systems nomenclature. A controller in ROS is a plugin to the controller manager implementing the controller interface and that typically interfaces with the joint command interface and/or the joint state interface. Note that a controller in ROS can perform a function which is not typically a function of a controller in a control system. An example is the `joint_state_controller`, which a control engineer would suppose to be a state space controller for joints, but is just a publisher of the values of positions and velocities of the joints.
Another problem is that it appears that the ROS control infrastructure was thought for single input, single output (SISO) controllers where each controller handles a scalar reference value, a scalar sensor value and generates a scalar output. Furthermore, discussions on some

ISBN 978-3-8007-3601-0

ROS developers forums in Internet reveal that there is an assumption that multi input, multi output (MIMO) controllers can be implemented by composing with SISO controllers. That is not, indeed, the case. Most advanced control laws for robots are intrinsically MIMO and can not be decomposed in a set of SISO control laws, let aside the problem of synchronizing many controllers to behave like a single MIMO controller.

Nonetheless, in this paper a pure ROS implementation of the controllers is attempted, in order to better understand the capabilities and limitations of ROS with regard to low-level controllers. See [14, 18, 26] for similar implementations using OROCOS to provide sophisticated real-time capabilities.

Figure 5 shows the architecture of low-level controllers in ROS. Controllers are plugins loaded by the controller manager which can load, unload activate and deactivate controllers.

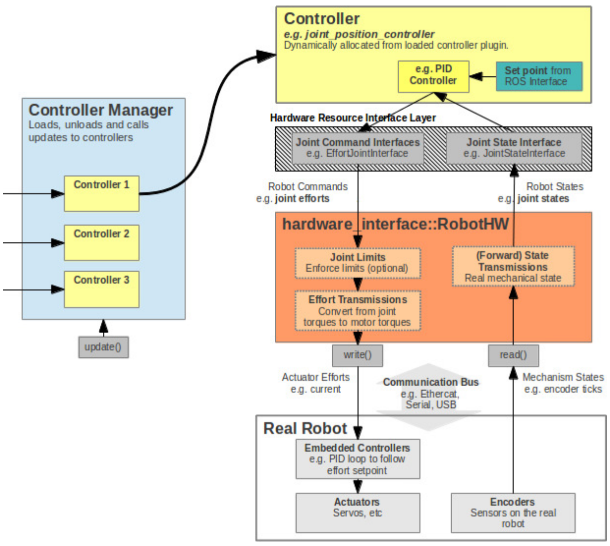

Figure 5: Architecture of low-level controllers in ROS.

Control cycles are performed at 1KHz. In each cycle the `update()` function of all active controllers are called in sequence so that each controller can perform its task for that cycle. If the controller should run at a lower rate, it should implement a sub-sampling by itself. From the digital control point of view, the model is that of a continuous time controller implemented in a digital computer with a sampling rate so fast that the effects of digitalizing can be neglected. Note however, that a sampling of 1KHz may be not enough to make that supposition hold for any robotic system, specially those attempting force or impedance control.

The model of continuous time controller is appropriate for nonlinear controllers, as is the case here. The `update()` function of the controller developed here implements the control law defined by (18), (20-22), (51),(52) e (53).

4.2 Simulation

The mobile manipulator was simulated by using the Gazebo simulator[15], as shown in **Figure 6**. By default, Gazebo implements its own controllers for each joint of the robots it simulates. For each joint it is possible to directly apply an effort (torque or force, depending on the type of the joint), or control velocity or position through PID controllers. There is also a Gazebo plugin which enables the attachment of controllers implemented in ROS. This plugin can be enable by adding the following code in the URDF file describing the robot:

```
<gazebo>
  <plugin name="gazebo_ros_control"
    filename="libgazebo_ros_control.so">
    <robotNamespace>
      /MYROBOT
    </robotNamespace>
  </plugin>
</gazebo>
```

where `/MYROBOT` is the namespace to be used for the robot, in this case `/twil/ros_control` is used.

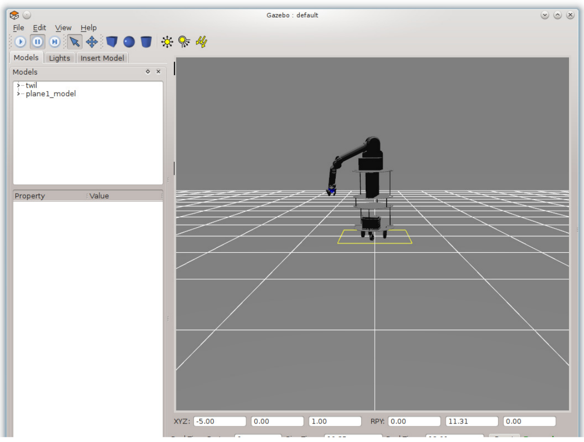

Figure 6: Mobile manipulator model in Gazebo.

Figure 7 shows the ROS nodes an topics created while the simulation is running in ROS. The node named `/gazebo` is the simulator and the topics with names starting with `/gazebo` are used to set or verify simulation parameters. The node named `/twil/ros_control/controller_spawner` is the controller manager. The controller manager loads two plugins: the `cart_linearizing_controller`, which implements the controller proposed in this paper and the `joint_state_controller` which just publishes the values of positions and velocities of the robot joints in the `/joint_states` topic.

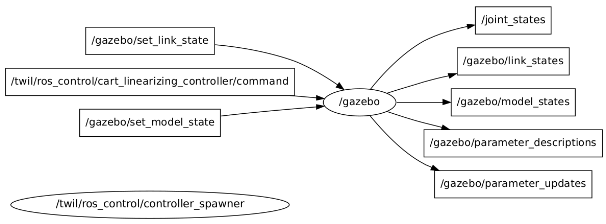

Figure 7: ROS nodes for the simulation of the proposed controller.

The controller implemented in the `cart_linearizing_controller` plugin receives its references from the `/twil/ros_control/cart_linearizing_controller/command` topic. The controller obtains the values of the state variables and sends the control effort to be applied in the robot by calling ROS services, which are not shown in the nodes/topics diagram.
The `/twil/ros_control/cart_linearizing_controller/command` and `/joint_states` topics are connected to the `/gazebo` node because they are connected to plugins, which are not ROS nodes by themselves and are loaded in the `/gazebo` node.

5 Conclusions

This paper presented a proposal for a mathematical model for a mobile manipulator and then proposed a backstepping controller for such a system. That controller was implemented in ROS as a plugin to ROS controller manager and the whole system was simulated by using Gazebo. Contrariwise to most controllers implemented in ROS which are SISO PID controllers, in this paper a MIMO nonlinear controller was implemented, which shows the flexibility of ROS interfaces but a lack in its documentation which only considers SISO PID controllers.

Acknowledgments

Authors would like to thank the financial support from Conselho Nacional de Pesquisa (CNPq), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e Fundação de Apoio à Pesquisa do Estado do Rio Grande do Sul (FAPERGS).

References

[1] A. Astolfi. On the stabilization of nonholonomic systems. In *Proceedings of the 33rd IEEE American Conference on Decision and Control*, pages 3481–3486, Lake Buena Vista, FL, Dez. 1994. Piscataway, NJ, IEEE Press.

[2] Taiser T. T. Barros and Walter Fetter Lages. Development of a firefighting robot for educational competitions. In *Proceedings of the 3rd International Conference on Robotics in Education*, Prague, Czech Republic, 2012.

[3] R. W. Brockett. *New Directions in Applied Mathematics*. Springer-Verlag, New York, 1982.

[4] Herman Bruyninckx. Open robot control software: The orocos project. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 2523–2528, Seoul, South Korea, 2001. IEEE Press.

[5] G. Campion, G. Bastin, and B. D'Andréa-Novel. Structural properties and classification of kinematic and dynamical models of wheeled mobile robots. *IEEE Transactions on Robotics and Automation*, 12(1):47–62, Feb 1996.

[6] C. Canudas de Wit and O. J. Sørdalen. Exponential stabilization of mobile robots with nonholonomic constraints. *IEEE Transactions on Automatic Control*, 37(11):1791–1797, Nov 1992.

[7] Steve Cousins. Ros topics. *IEEE Robotics and Automation Magazine*, pages 13–14, March 2010.

[8] Alexander Dietrich, Thomas Wimböck, Alin Albu-Schäffer, and Gerd Hirzinger. Reactive whole-body control: Dynamic mobile manipulation using a large number of actuated degrees of freedom. *IEEE Robotics and Automation Magazine*, pages 20–33, June 2012.

[9] King Sun Fu, Rafael C. Gonzales, and C. S. George Lee. *Robotics Control, Sensing, Vision and Intelligence*. Industrial Engineering Series. McGraw-Hill, New York, 1987.

[10] Edwardo F. Fukushima, Shigeo Hirose, and Takeo Hayashi. Basic manipulation considerations for the articulated body mobile robot. In *Proceedings of the 1998 JEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, pages 386–393, Victoria, B.C, Canada, October 1998. Piscataway, NJ, IEEE Press.

[11] John-morten Godhavn and Olav Egeland. A lyapunov approach to exponential stabilization of nonholonomic systems in power form. *IEEE Transactions on Automatic Control*, 42(7):1028–1032, Jul. 1997.

[12] Brad Hamner, Seth Koterba, Jane Shi, Reid Simmons, and Sanjiv Singh. An autonomous mobile manipulator for assembly tasks. *Autonomous Robot*, 28:131–149, January 2010.

[13] Satoshi Ide, Tomohito Takubo, Kenichi Ohara, Yasushi Mae, and Tatsuo Arai. Real-time trajectory planning for mobile manipulator using model predictive control with constraints. *8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 244–249, November 2011.

[14] Darlan Ioris, Walter Fetter Lages, and Diego Caberlon Santini. Integrating the OROCOS framework and the barrett WAM robot. In *Proceedings of the 5th Workshop on Applied Robotics and Automation*, Bauru, SP, Brazil, 2012. Sociedade Brasileira de Automática.

[15] Nate Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, volume 3, pages 2149–2154, Sept 2004.

[16] Petar V. Kokotović. Developments in nonholonomic control problems. *IEEE Control Systems Magazine*, 12(3):7–17, Jun 1992.

[17] Walter Fetter Lages and Elder M. Hemerly. Smooth time-invariant control of wheeled mobile robots. In *Proceedings of The XIII International Conference on Systems Science*, Wrocław, Poland, 1998. Technical University of Wrocław.

[18] Walter Fetter Lages, Dalan Ioris, and Diego Santini. An architecture for controlling the barrett wam robot using ros and orocos. In *Proceedings of the Joint 45th International Symposium on Robotics and 8th German Conference on Robotics*, Munich, Germany, 2014.

[19] Pasquale Lucibello and Giuseppe Oriolo. Robust stabilization via iterative state steering with an application to chained-form systems. *Automatica*, 37(1):71–79, Jan. 2001.

[20] Musa Mailah, Endra Pitowarno, and Hishamuddin Jamaluddin. Robust motion control for mobile manipulator using resolved acceleration and proportional-integral active force control. *International Journal of Advanced Robotic Systems*, 2:125–134, December 2005.

[21] Ken'ichiro Nagasaka, Yasunori Kawanami, Satoru Shimizu, Takashi Kito, Toshimitsu Tsuboi, Atsushi Miyamoto, Tetsuharu Fukushima, and Hideki Shimomura. Whole-body cooperative force control for a two-armed and two-wheeled mobile robot using generalized inverse dynamics and idealized joint units. *International Conference on Robotics and Automation*, pages 3377–3383, May 2010.

[22] J. B. Pomet, B. Thuilot, G. Bastin, and G. Campion. A hybrid strategy for the feedback stabilization of nonholonomic mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 129–134, Nice, France, Mai. 1992. IEEE Press.

[23] Vasile Mihai Popov. *Hyperstability of Control Systems*, volume 204 of *Die Grundlehren der matematischen Wissenshaften*. Springer-Verlag, Berlin, Heidelberg, New York, 1973.

[24] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. ROS: an open-source robot operating system. In *Proceedings of the IEEE International Conference on Robotics and Automation, Workshop on Open Source Robotics*, Kobe, Japan, May 2009. IEEE Press.

[25] Fazal-ur Rehman, Muhammad Rafiq, and Qarab Raza. Time-varying stabilizing feedback control for a sub-class of nonholonomic systems. *European Journal of Scientific Research*, 53(3):346–358, May. 2011.

[26] Diego Caberlon Santini and Walter Fetter Lages. An architecture for robot control based on the OROCOS framework. In *Proceedings of the 4th Workshop on Applied Robotics and Automation*, Bauru, SP, Brazil, 2010. Sociedade Brasileira de Automática.

[27] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*. Springer-Verlag, Stanford University, USA, 2008.

[28] O. J. Sørdalen. *Feedback Control of Nonholonomic Mobile Robots*. Thesis (dr. ing.), The Norwegian Institute of Technology, Trondheim, Norway, 1993.

[29] A. R. Teel, R. M. Murray, and G. C. Walsh. Nonholonomic control systems: from steering to stabilization with sinusoids. *International Journal of Control*, 62(4):849–870, 1995.

[30] Bill Wong. Cooperation leads to smarter robots. *Electronic Design*, 59(5):24–30, April 2011.

[31] Yoshio Yamamoto and Xiaoping Yun. Unified analysis on mobility and manipulability of mobile manipulators. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 1200–1206, Detroit, Michigan, May 1999. Piscataway, NJ, IEEE Press.

[32] Qing Yu and I-Ming Chen. A general approach to the dynamics of nonholonomic mobile manipulator systems. *Journal of Dynamic Systems, Measurement, and Control*, 124:512–521, December 2002.

ISBN 978-3-8007-3601-0