Control of Industrial and Mobile Robots

Prof. Rocco, Bascetta

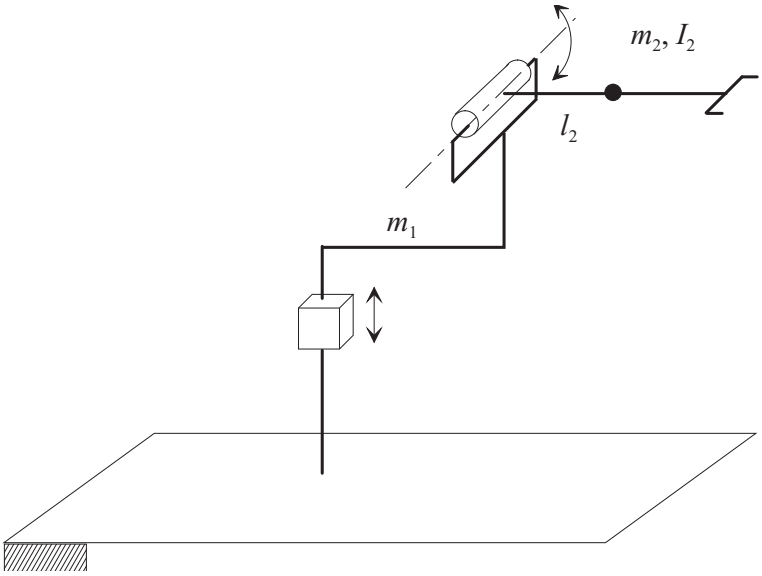
January 14, 2025

SOLUTION

Control of Industrial and Mobile Robots
Prof. Luca Bascetta and Prof. Paolo Rocco

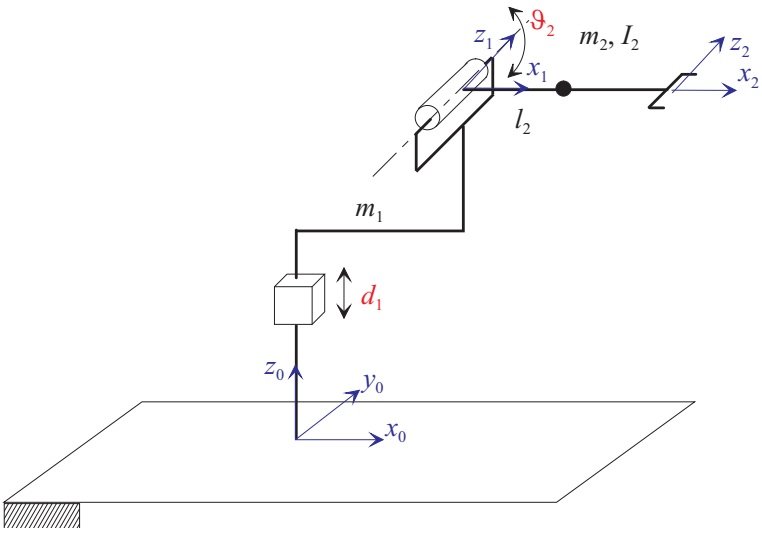
EXERCISE 1

1. Consider the manipulator sketched in the picture:



Find the expression of the inertia matrix $\mathbf{B}(\mathbf{q})$ of the manipulator[1].

Denavit-Hartenberg frames can be defined as sketched in this picture:



[1]The cross product between vector $a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$ and $b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$ is $c = a \times b = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix}$

Computations of the Jacobians:

Link 1

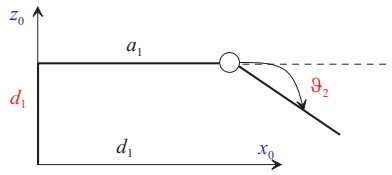
$$\mathbf{J}_P^{(l_1)} = \begin{bmatrix} \mathbf{j}_{P_1}^{(l_1)} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{z}_0 & \mathbf{0} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}$$

Link 2

$$\mathbf{J}_P^{(l_2)} = \begin{bmatrix} \mathbf{j}_{P_1}^{(l_2)} & \mathbf{j}_{P_2}^{(l_2)} \end{bmatrix} = \begin{bmatrix} \mathbf{z}_0 & \mathbf{z}_1 \times (\mathbf{p}_{l_2} - \mathbf{p}_1) \end{bmatrix} = \begin{bmatrix} 0 & -l_2 s_2 \\ 0 & 0 \\ 1 & -l_2 c_2 \end{bmatrix}$$

$$\mathbf{J}_O^{(l_2)} = \begin{bmatrix} \mathbf{j}_{O_1}^{(l_2)} & \mathbf{j}_{O_2}^{(l_2)} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{z}_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

For the above computations, we can make reference to the following picture:



and to the following auxiliary vectors:

$$\mathbf{p}_{l_2} = \begin{bmatrix} a_1 + l_2 c_2 \\ 0 \\ d_1 - l_2 s_2 \end{bmatrix}, \mathbf{p}_1 = \begin{bmatrix} a_1 \\ 0 \\ d_1 \end{bmatrix}, \mathbf{z}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

The inertia matrix can be computed now:

$$\begin{aligned} \mathbf{B}(\mathbf{q}) &= m_1 \mathbf{J}_P^{(l_1)^T} \mathbf{J}_P^{(l_1)} + m_2 \mathbf{J}_P^{(l_2)^T} \mathbf{J}_P^{(l_2)} + I_2 \mathbf{J}_O^{(l_2)^T} \mathbf{J}_O^{(l_2)} \\ &= m_1 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + m_2 \begin{bmatrix} 1 & -l_2 c_2 \\ -l_2 c_2 & l_2^2 \end{bmatrix} + I_2 \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} b_{11} & b_{12} \\ b_{12} & b_{22} \end{bmatrix} \end{aligned}$$

where:

$$\begin{aligned} b_{11} &= m_1 + m_2 \\ b_{12} &= -m_2 l_2 c_2 \\ b_{22} &= I_2 + m_2 l_2^2 \end{aligned}$$

2. Compute the gravitational terms for this robot.

Since the vertical axis is the $\mathbf{z}_0$ axis pointing upwards, the gravity acceleration vector is:

$$\mathbf{g}_0 = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}$$

The gravitational torques are thus:

$$\mathbf{g} = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix},$$

where:

$$g_1 = -m_1 \mathbf{g}_0^T \mathbf{j}_{P_1}^{(l_1)} - m_2 \mathbf{g}_0^T \mathbf{j}_{P_1}^{(l_2)} = -m_1 \mathbf{g}_0^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - m_2 \mathbf{g}_0^T \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = (m_1 + m_2)\, g$$

$$g_2 = -m_1 \mathbf{g}_0^T \mathbf{j}_{P_2}^{(l_1)} - m_2 \mathbf{g}_0^T \mathbf{j}_{P_2}^{(l_2)} = -m_2 \mathbf{g}_0^T \begin{bmatrix} -l_2 s_2 \\ 0 \\ -l_2 c_2 \end{bmatrix} = -m_2 g l_2 c_2$$

3. Ignoring the Coriolis and centrifugal terms, write the dynamic model of the manipulator and show that this model is linear with respect to a certain set of dynamic parameters.

Neglecting Coriolis and centrifugal terms, the dynamic model can be written as:

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau$$

The two equations that form the model are:

$$\begin{aligned} (m_1 + m_2)\, \ddot{d}_1 - m_2 l_2 c_2 \ddot{\vartheta}_2 + (m_1 + m_2)\, g &= \tau_1 \\ -m_2 l_2 c_2 \ddot{d}_1 + \left(m_2 l_2^2 + I_2\right) \ddot{\vartheta}_2 - m_2 g l_2 c_2 &= \tau_2 \end{aligned}$$

The model can be written in the following form which is linear in the dynamic parameters:

$$\mathbf{Y}\left(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}\right) \mathbf{\Pi} = \tau$$

where the vector of dynamic parameters is expressed as:

$$\mathbf{\Pi} = \begin{bmatrix} m_1 + m_2 \\ m_2 l_2 \\ m_2 l_2^2 + I_2 \end{bmatrix}$$

while the regressor matrix is:

$$\mathbf{Y} = \begin{bmatrix} \ddot{d}_1 + g & -c_2 \ddot{\vartheta}_2 & 0 \\ 0 & -c_2 \ddot{d}_1 - g c_2 & \ddot{\vartheta}_2 \end{bmatrix}$$

4. The linearity of the model in a set of dynamic parameters allows to setup experiments for the identification of such parameters. For a generic manipulator, explain what are the variables that need to be recorded during the experiments. With reference to the dynamic model of this exercise, is it possible to experimentally identify the mass of the first link?

   Since at each time instant the regressor matrix has to be computed, we need to record the joint positions, velocities and accelerations, along with the joint torques.
   For this specific manipulator it is not possible to identify the mass of the first link alone, we can only estimate the sum of the masses of the two links.

EXERCISE 2

1. Explain what is the purpose of the kinematic calibration of a robot manipulator and why it is needed.

   The kinematic calibration is a process, based on a series of measurements of the manipulator's end effector, that allows to obtain accurate estimates of the DH parameters. It is needed because of tolerances in mechanical building of components and in the assembly of the links and joints as well as for possible issues in encoder mounting

2. In the kinematic calibration of a robot manipulator the following equation is used:

$$\Delta \mathbf{x} = \mathbf{\Phi} \Delta \zeta$$

   Explain the meaning of each symbol used in such equation, as well as the size of the vectors.

   In this equation:
   $\Delta \mathbf{x} = \mathbf{x}_{act} - \mathbf{x}_{nom}$ are the deviations of the actual pose variables from the nominal ones (a 6×1 vector)
   $\mathbf{\Phi}$ is the calibration matrix, defined as:

$$\mathbf{\Phi} = \begin{bmatrix} \frac{\partial \mathbf{k}}{\partial \mathbf{a}} & \frac{\partial \mathbf{k}}{\partial \alpha} & \frac{\partial \mathbf{k}}{\partial \mathbf{d}} & \frac{\partial \mathbf{k}}{\partial \theta} \end{bmatrix}$$

   where $\mathbf{k}$ is the direct kinematic function, while $\mathbf{a}, \alpha, \mathbf{d}, \theta$ are the vectors of the DH parameters for the n joints. Matrix $\mathbf{\Phi}$ has 6 rows and $4n$ columns.
   $\Delta\zeta$ are the deviations of the DH parameters and are defined as:

$$\Delta \zeta = \begin{bmatrix} \Delta \mathbf{a} \\ \Delta \alpha \\ \Delta \mathbf{d} \\ \Delta \theta \end{bmatrix}$$

   $\Delta\zeta$ is a $4n \times 1$ vector.

3. Based on the above equation, explain how the kinematic calibration can be performed.

   The equation:

$$\Delta\mathbf{x} = \mathbf{\Phi}\Delta\zeta$$

   is a system of 6 equations in $4n$ unknowns. We need to perform a certain number l of experiments, each time changing the pose of the end-effector. Stacking the above equations referred to the various poses, we have:

$$\Delta\bar{\mathbf{x}} = \begin{bmatrix} \Delta\mathbf{x}_1 \\ \vdots \\ \Delta\mathbf{x}_l \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{\Phi}_1 \\ \vdots \\ \Delta\mathbf{\Phi}_l \end{bmatrix} \Delta\zeta = \bar{\mathbf{\Phi}}\Delta\zeta$$

   this equation can be solved for $\Delta\zeta$ in a least squares form:

$$\Delta\zeta = \bar{\mathbf{\Phi}}^{\sharp}\Delta\bar{\mathbf{x}}$$

   where $\bar{\mathbf{\Phi}}^{\sharp}$ is the left pseudoinverse of matrix $\bar{\mathbf{\Phi}}$.

   The deviation $\Delta\zeta$ is then added to the nominal values of the DH parameters ζ. The process can be iterated until convergence under a certain threshold.

4. Consider now a kinematically redundant manipulator. Write the general solution of the inverse kinematics at velocity level. Is the pseudoinverse matrix that appears in this equation the same pseudoinverse of the kinematic calibration problem?

   The forward kinematics at velocity level is written as:

$$\dot{\mathbf{r}} = \mathbf{J}\dot{\mathbf{q}}$$

   where $\dot{\mathbf{q}}$ are joint velocities, $\dot{\mathbf{r}}$ are task velocities and $\mathbf{J}$ is a Jacobian matrix. The solution of the inverse kinematics is written as:

$$\dot{\mathbf{q}} = \mathbf{J}^{\sharp}\dot{\mathbf{x}} + \left(\mathbf{I} - \mathbf{J}^{\sharp}\mathbf{J}\right)\dot{\mathbf{q}}_0$$

   where $\mathbf{J}^{\sharp}$ is the right pseudoinverse of the Jacobian (in the kinematic calibration problem the left pseudoinverse is used).

EXERCISE 3

Consider the following system of kinematic constraints

$$\begin{aligned} 2\dot{q}_1 + q_1\dot{q}_2 - 3\dot{q}_3 &= 0 \\ 2\dot{q}_2 - q_2\dot{q}_3 &= 0 \end{aligned}$$

where $\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}^T$.

1. Using the necessary and sufficient condition, determine if the first constraint, considered as an independent constraint, is holonomic or nonholonomic.

   Considering the first constraint

$$a^T(\mathbf{q})\,\dot{\mathbf{q}} = \begin{bmatrix} 2 & q_1 & -3 & 0 \end{bmatrix}\dot{\mathbf{q}} = 0$$

   as an independent constraint, we can write the following equalities

$$\frac{\partial\left[\alpha(\mathbf{q})\,a_k(\mathbf{q})\right]}{\partial q_j} = \frac{\partial\left[\alpha(\mathbf{q})\,a_j(\mathbf{q})\right]}{\partial q_k}$$

   for $(j,k) \in \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$, obtaining

$$\begin{aligned}
&\frac{\partial\left[\alpha(\mathbf{q})\,q_1\right]}{\partial q_1} = q_1\frac{\partial\alpha(\mathbf{q})}{\partial q_1} + \alpha(\mathbf{q}) = \frac{\partial\left[2\alpha(\mathbf{q})\right]}{\partial q_2}\\
&\frac{\partial\left[-3\alpha(\mathbf{q})\right]}{\partial q_1} = \frac{\partial\left[2\alpha(\mathbf{q})\right]}{\partial q_3}\\
&\frac{\partial\left[2\alpha(\mathbf{q})\right]}{\partial q_4} = 0\\
&\frac{\partial\left[-3\alpha(\mathbf{q})\right]}{\partial q_2} = \frac{\partial\left[\alpha(\mathbf{q})\,q_1\right]}{\partial q_3}\\
&\frac{\partial\left[\alpha(\mathbf{q})\,q_1\right]}{\partial q_4} = 0\\
&\frac{\partial\left[-3\alpha(\mathbf{q})\right]}{\partial q_4} = 0
\end{aligned}$$

   Considering, for example, the last relation it follows that

$$\frac{\partial\alpha(\mathbf{q})}{\partial q_4} = 0$$

   From the second and the fourth it follows that

$$\begin{aligned}
-3\frac{\partial\alpha(\mathbf{q})}{\partial q_1} &= 2\frac{\partial\alpha(\mathbf{q})}{\partial q_3}\\
-3\frac{\partial\alpha(\mathbf{q})}{\partial q_2} &= q_1\frac{\partial\alpha(\mathbf{q})}{\partial q_3}
\end{aligned}$$

   and thus

$$q_1\frac{\partial\alpha(\mathbf{q})}{\partial q_1} = 2\frac{\partial\alpha(\mathbf{q})}{\partial q_2}$$

   Finally, substituting this relation into the first equality one obtains

$$2\frac{\partial\alpha(\mathbf{q})}{\partial q_2} + \alpha(\mathbf{q}) = 2\frac{\partial\alpha(\mathbf{q})}{\partial q_2}$$

   and the only solution is thus $\alpha(\mathbf{q}) = 0$. As a consequence, the first is a nonholonomic constraint.

2. Is the second constraint, considered as an independent constraint, holonomic or nonholonomic?

   For the second constraint one can apply again the necessary and sufficient condition, but there is also a simpler way.
   The constraint can be rewritten as follows

$$2\frac{\dot{q}_2}{q_2} = \dot{q}_3$$

   Integrating each side of the constraint one obtains

$$2\ln q_2 + c_1 = q_3 + c_2$$

   where c_1 and c_2 are arbitrary integration constants.
   The second constraint is integrable and thus holonomic.

3. Assuming that $A^T(\mathbf{q})\,\dot{\mathbf{q}} = 0$ is the Pfaffian form of the system of two constraints, and

$$g_1(\mathbf{q}) = \begin{bmatrix} 6 - q_1 q_2 \\ 2q_2 \\ 4 \\ 0 \end{bmatrix} \qquad g_2(\mathbf{q}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

   are two vectors in the null space of $A^T(\mathbf{q})$, demonstrate that the system of two constraints is holonomic.

   The two constraints can be rewritten in Pfaffian form as

$$A^T(\mathbf{q}) = \begin{bmatrix} 2 & q_1 & -3 & 0 \\ 0 & 2 & -q_2 & 0 \end{bmatrix}$$

   Note that $A^T(\mathbf{q})$ has rank 2, a base of the null space is thus composed of two vectors. $g_1(\mathbf{q})$ and $g_2(\mathbf{q})$ are linearly independent and are thus a base of the null space of $A^T(\mathbf{q})$.
   The procedure to compute the accessibility distribution is initialized with $\Delta_1 = \text{span}\{g_1, g_2\}$.
   A third vector can be generated as

$$g_3(\mathbf{q}) = [g_1, g_2] = \frac{\partial g_2}{\partial \mathbf{q}} g_1 - \frac{\partial g_1}{\partial \mathbf{q}} g_2 = \mathbf{0} - \begin{bmatrix} -q_2 & -q_1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

   No other vector fields can be added, we conclude that the accessibility space has dimension 2, that is equal to $n - k$, and thus the system of constraints is holonomic.

4. Does the following kinematic model

$$\dot{\mathbf{q}} = g_1(\mathbf{q})\,u_1 + g_2(\mathbf{q})\,u_2$$

   where $g_1(\mathbf{q})$ and $g_2(\mathbf{q})$ are the vectors introduced in the previous step, describe the motion of the mobile robot characterized by the system of two constraints?

We already know that $g_1(\mathbf{q})$ and $g_2(\mathbf{q})$ are two linearly independent vectors in the null space of $A^T(\mathbf{q})$. We can thus conclude that the kinematic model describes the motion of the mobile robot characterized by the system of two constraints.

EXERCISE 4

Consider the design of the trajectory tracking controller of a robot described by the rear-wheel-drive bicycle model.

1. Write the relations that allow to transform the bicycle model into the canonical simplified model. Under which assumptions do these relations hold?

   Assuming that the steering rate limit is so high that the steering angle can be changed instantaneously, we can simplify the bicycle model as

   $$\begin{aligned}\dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= v\frac{\tan\phi}{\ell}\end{aligned}$$

   where ϕ is the steering angle and ℓ the length of the bicycle.
   The relations to transform the bicycle into the canonical simplified model are

   $$v = v \qquad \omega = v\frac{\tan\phi}{\ell}$$

2. Considering a point P, located along the linear velocity vector v at a distance ε from the wheel contact point, write the analytical expression of a feedback linearizing controller for the canonical simplified model.

   The equations of the feedback linearizing controller for the canonical simplified model are

   $$\begin{aligned}v &= v_{x_P}\cos\theta + v_{y_P}\sin\theta \\ \omega &= \frac{v_{y_P}\cos\theta - v_{x_P}\sin\theta}{\varepsilon}\end{aligned}$$

3. Using the transformation derived in step 1, write the analytical expression of the feedback linearizing controller for the bicycle model using the relations of the feedback linearizing controller for the canonical simplified model. Write the detailed procedure, not only the results.

   From step 1 it follows that

   $$v = v \qquad \phi = \arctan\left(\frac{\omega\ell}{v}\right)$$

Substituting now the equations of the feedback linearizing controller for the canonical simplified model into the previous relations one obtains

$$v = v_{x_P} \cos\theta + v_{y_P} \sin\theta$$
$$\phi = \arctan\left(\frac{\ell}{\varepsilon} \frac{v_{y_P} \cos\theta - v_{x_P} \sin\theta}{v_{x_P} \cos\theta + v_{y_P} \sin\theta}\right)$$

4. Consider an implementation of the controller as a ROS node. Assuming it receives the robot actual pose measurement as a *Float64MultiArray* message, where the array elements are x, y, θ, and publishes the robot commands as a *Float64MultiArray* message, where the array elements are v and ϕ, complete the code of the callback in order to store the actual pose in the class variables *act_pose_x*, *act_pose_y*, *act_pose_theta*. Write the code to compute the reference velocities, v_{x_P} and v_{y_P}, as unitary steps starting after 5 seconds, and the vehicle commands, using function

```
void canonical_controller::bicycle_feedback_linearization(double vPx,
    double vPy, double& v, double& phi)
```

that is already implemented, and to publish them using an already defined publisher *vehicleCommand_publisher*.

```
void canonical_controller::vehiclePose_MessageCallback(const std_msgs::
    Float64MultiArray::ConstPtr& msg)
{









}

void canonical_controller::PeriodicTask(void)
{
```

```
}
```

The complete code is shown below.

```
void canonical_controller::vehiclePose_MessageCallback(const std_msgs::
    Float64MultiArray::ConstPtr& msg)
{
    act_pose_x = msg->data.at(0);
    act_pose_y = msg->data.at(1);
    act_pose_theta = msg->data.at(2);
}

void canonical_controller::PeriodicTask(void)
{
    /* Generate input commands */
    double vPx, vPy;
    if (ros::Time::now().toSec()<=5.0)
    {
        vPx = 0.0;
        vPy = 0.0;
    }
    else
    {
        vPx = 1.0;
        vPy = 1.0;
    }

    /* Compute the control action */
    double v, phi;
    bicycle_feedback_linearization(vPx, vPy, v, phi);;

    /* Publish vehicle commands */
    std_msgs::Float64MultiArray vehicleCommandMsg;
```

```
        vehicleCommandMsg.data.push_back(v);
        vehicleCommandMsg.data.push_back(phi);
        vehicleCommand_publisher.publish(vehicleCommandMsg);
    }
```