Exercise 1

Kinematics of a single robot leg

1. Introduction

In this exercise, we are going to analyze the kinematics of a single robot leg with a point foot. Starting with a set of generalized coordinates, we will elaborate relative rotations, translations, and homogeneous transformations. Subsequently, using foot point Jacobians, we will describe contact constraints and perform a trajectory tracking task using inverse kinematics.

Figure 1 describes a single leg that is attached to a body fixed frame B. This leg has three degrees of freedom consisting of relative rotations about α(alpha) around ${}_Be_x$, about β (beta) around ${}_1e_y$, and about γ(gamma) around ${}_2e_y$. Hence, the generalized coordinates are given by $q = [\alpha\ \beta\ \gamma]^T$.

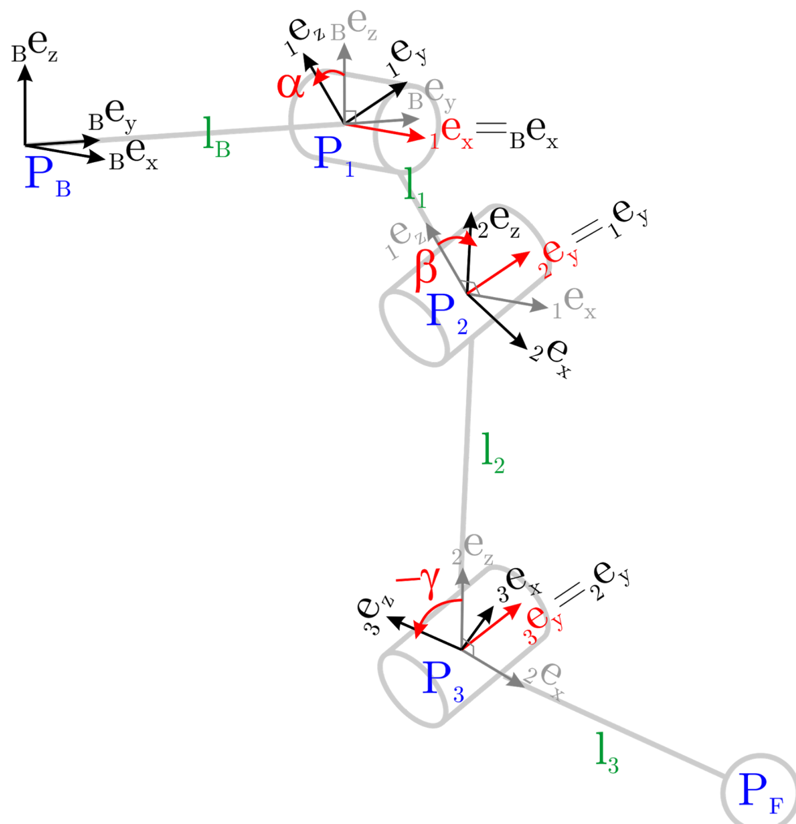

Figure 1: Leg robot frames

2. Relative Rotation Matrices

Given the kinematic description of a single leg with three degrees of freedom (q: $= [\alpha\ \beta\ \gamma]^T$) we determine the relative rotation matrices R_{AC} rotating a vector r from an arbitrary coordinate system *C* to *A*:

$$_A r = R_{AC}\ {}_C r \tag{1}$$

Task: as a function of the generalized coordinates *alpha, beta, gamma*, what are the three relative rotation matrices? Please edit **ex01.m** in such manner that it computes the relative rotation matrices.

3. Homogeneous Transformation

Given the relative rotation matrices from the previous problem and choosing unitary link lengths ($l_B = l_1 = l_2 = l_3 = 1$), we determine the homogeneous transformation H that transforms the footpoint position r_F represented in coordinate frame 3 to coordinate frame B:

$$\begin{pmatrix} {}_B\mathbf{r}_{BF} \\ 1 \end{pmatrix} = \mathbf{H}_{B3} \begin{pmatrix} {}_3\mathbf{r}_{3F} \\ 1 \end{pmatrix} \tag{2}$$

Task: as a function of the generalized coordinates *alpha, beta, gamma*, what are the three relative position vectors and homogeneous transformation matrices? Please edit **ex02.m** with the appropriate answers.

4. Jacobians and Differential Kinematics

Given the end effector position r_{BF} as a function of generalized coordinates q, we will determine the corresponding Jacobian $J_{BF} = \partial r_{BF}/\partial q$ and velocity.

Task: as a function of the generalized coordinates $q = [\alpha\ \beta\ \gamma]^T$, what is the foot point Jacobian and generalized velocity for a desired Cartesian motion? Please edit **ex03.m** with the appropriate answers.

5. Numeric Inverse Kinematics

In this exercise, we will solve the following inverse kinematics problem: given ${}_B r_{BF} = {}_B r_{BF}(q)$, find the according q. Since the function ${}_B r_{BF}$ is often hard to analytically invert, we will implement a numerical approach.

Task: implement the numerical method (Newton method) to determine the goal configuration q_{goal}. First of all open the scene file **Exercise1 Leg.ttt** located in *ex04ex05\scene* using V-REP and edit **ex04.m** located in *ex04ex05\code\common\vrep*. After applying the correct joint angles the foot point must reach the dummy object **rGoal** placed in the scene. Run the matlab code and the application will communicate with V-REP automatically.

6. Trajectory Following Using Inverse Differential Kinematics

In this problem we apply inverse differential kinematics to control a footpoint trajectory. Given a desired goal trajectory $r(t)$ as well as a starting configuration q_0, we have to determine the joint velocities $\dot{q} = [\dot{\alpha}\ \dot{\beta}\ \dot{\gamma}]^T$.

Task: implement a simple proportional controller to determine the desired Cartesian foot point velocity and apply inverse differential kinematics to determine the joint velocities. Still using the same scene file **Exercise1 Leg.ttt** edit **ex05.m** file located in *ex04ex05\code\common\vrep* and complete the variables **v** and **dq** in such manner that the foot point follows the circular trajectory. Compare the desired trajectory and the obtained with your algorithm by exporting the points from V-REP into in Matlab workspace. Plot both of them and discuss the results.

Steps to export graph from V-REP into Matlab:

1. In V-REP select *Graph* located on *Scene hierarchy;*
2. Click on *File>Export>Selected Graph as CSV…* and save the file;
3. In Matlab click on *Import Data* and select the *.csv* file;
4. Select *Numeric Matrix* and click on *Import Selection>Import Data*. After that the variable will be available on matlab workspace.