ME 449 Robotic Manipulation
Fall 2015
Problem Set 2
Due Tuesday October 27 at beginning of class (turn in on Canvas)

Throughout this quarter you will be developing your own robotics software library. We recommend Matlab or Mathematica for some of the functions and visualization capabilities that they provide natively, but if you are comfortable and confident with how to use graphics in another language, you can use another language.

Your software library should be commented, clearly written, and well tested by examples that you create. Make sure you establish good modularity and a clear programming style right from the beginning, since your later code will build on the earlier code. You will submit your software for testing on Canvas.

All code must be your own work. The point of these assignments is to develop your own skill in programming and your own understanding of the material. It is OK to share your understanding of the syntax and functions of the programming language you are using, but not to share code.

You should not be using built-in matrix inverse functions. Matrix inverse algorithms are slow, generally, and do not take advantage of the structure of the matrices we are using.

As always, where appropriate, show your work and explain your results. If you think a question is unclear, then state your assumptions to clarify it.

1. Write the following functions for your robotics library. (If you happen to be using a programming language that does not have its own matrix algebra library, you will need to create functions that multiply matrices, including vectors, and perform other simple operations like taking a transpose.)

- Helper functions you will find useful, if not already in your programming language: `Magnitude`, which takes a vector and returns its length, and `Normalize`, which takes a vector and scales it to a unit vector.
- `RotInv`: Takes an $R \in SO(3)$ and returns its inverse. Do not use a built-in matrix inverse function; use the transpose. (It is not required, but you should consider checking first if the given matrix is a valid rotation matrix, and having the function return "Error" if not. The matrix is not a valid rotation matrix if $R^T R$ is not sufficiently close to I and if $\det R$ is not sufficiently close to 1. You should also consider error-checking for certain other functions below.)
- `VecToso3`: Takes a 3-vector (representing an angular velocity) and returns the 3×3 skew-symmetric matrix version, an element of $so(3)$.
- `so3ToVec`: Takes a 3×3 skew-symmetric matrix (an element of $so(3)$) and returns the corresponding 3-vector.
- `AxisAng3`: Takes a 3-vector of exponential coordinates for rotation, $r = \hat{\omega}\theta$, and returns the unit rotation axis $\hat{\omega}$ and the rotation amount θ.
- `MatrixExp3`: Takes a 3-vector of exponential coordinates $r = \hat{\omega}\theta$ and returns $R' \in SO(3)$ that is achieved by rotating about $\hat{\omega}$ by θ from an initial orientation $R = I$.
- `MatrixLog3`: Takes an $R \in SO(3)$ and returns the corresponding 3-vector of exponential coordinates $r = \hat{\omega}\theta$.
- `RpToTrans`: Takes an $R' \in SO(3)$ and a position $p \in \mathbb{R}^3$ and returns the corresponding homogeneous transformation matrix $T \in SE(3)$.
- `TransToRp`: Takes a $T \in SE(3)$ and returns the corresponding $R' \in SO(3)$ and $p \in \mathbb{R}^3$.
- `TransInv`: Takes an element of $SE(3)$ and returns the inverse.
- `VecTose3`: Takes a 6-vector (representing a spatial velocity) and returns the corresponding 4×4 matrix, an element of $se(3)$.

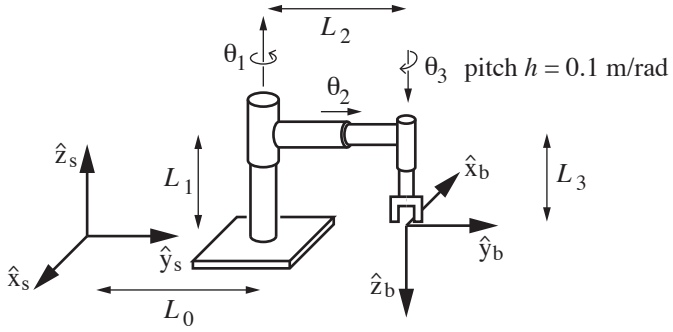

Figure 1: An RPH open chain shown at its home position. All arrows along/about joint axes are drawn in the positive direction (increase of the joint value). The pitch of the helical joint is 0.1 m/rad; it advances linearly by 0.1 m for every radian rotated.

- `se3ToVec`: Takes an element of $se(3)$ and returns the corresponding spatial velocity as a 6-vector.
- `Adjoint`: Takes a $T \in SE(3)$ and returns the 6×6 adjoint representation $[\mathrm{Ad}_T]$.
- `ScrewToAxis`: Takes a point $q \in \mathbb{R}^3$, a unit axis $\hat{s} \in \mathbb{R}^3, \|\hat{s}\| = 1$, and a screw pitch $h \in \mathbb{R}$ and returns the corresponding screw axis $\mathcal{S} = (\omega, v)$, a normalized "unit" spatial velocity. The frame of this spatial velocity is the same frame that q and $\hat{s}$ are given in. (It is OK if this function only handles finite values of h. For finite values of h, the angular portion of $\mathcal{S}$ satisfies $\|\omega\| = 1$.)
- `AxisAng6`: Takes a 6-vector of exponential coordinates for rigid-body motion, $\mathcal{S}\theta$, and returns the screw axis $\mathcal{S}$ and the distance traveled along/about the screw axis θ.
- `MatrixExp6`: Takes a 6-vector of exponential coordinates $\mathcal{S}\theta$ and returns $T' \in SE(3)$ that is achieved by traveling along/about the screw axis $\mathcal{S}$ a distance θ from an initial configuration $T = I$.
- `MatrixLog6`: Takes a $T \in SE(3)$ and returns the corresponding 6-vector of exponential coordinates $\mathcal{S}\theta$.
- `FKinFixed`: Takes $M \in SE(3)$ representing the position and orientation of the end-effector frame when the manipulator is at its home position ($\theta_i = 0$ for all n joints of the robot); a list of screw axes $\mathcal{S}_i$ for the n joints of the robot, expressed in a fixed world frame; and a list of joint coordinates θ_i; and returns $T \in SE(3)$ representing the end-effector frame when the joints are at the angles specified.
- `FKinBody`: Same as above, except now the screw axes are expressed in the end-effector frame as $\mathcal{B}_i$.

Turn in your commented source code for these functions, for testing. Comments for each function should include a sample usage and output.

2. Turn in a log showing the results of the tests below.

(i) Test your `ScrewToAxis` function with $q = (3, 0, 0)^T$, $\hat{s} = (0, 0, 1)^T$, and $h = 2$.

(ii) Test your `MatrixExp6` function with a screw axis $\mathcal{S} = (0, 1/\sqrt{2}, 1/\sqrt{2}, 1, 2, 3)^T$ and a distance $\theta = 1$.

(iii) Test your `MatrixLog6` function with

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

(iv) For the RPH robot shown at its home position in Figure 1, express the configuration of the end-effector M and the screw axes $\mathcal{S}_i$ for the three joints. Now, for the joint variables $\theta = (\pi/2, 3, \pi)$, use your `FKinFixed` function to express the end-effector configuration $T \in SE(3)$.

(v) Same as above, but express the screw axes as $\mathcal{B}_i$ and use your `FKinBody` function instead to find $T \in SE(3)$. (The answer should be the same!)