Chapter 2 HW Hints

Problem 2.1 Get the rotation matrices from text equation (2.77)-(2.79) and make sure to apply them in the correct order. Note that first rotation should be the first \( R \) matrix to act on position vector \( A_P \).

Problem 2.3 Make use of the fact that the “mapping” matrix that changes descriptions from \( bP \) to \( aP \) is also the matrix that describes frame \( \{B\} \) relative to frame \( \{A\} \). Again use (2.77)-(2.79). NOTE: Since the second rotation is performed about the rotated (frame \( \{B\} \)) axis, consider postmultiplying, since postmultiplication by an operator performs the operation about the frames’s own axes.

Problem 2.5 Consider rotation matrix \( a_B R \) to be an operator which operates on vector \( P \). The eigenvalue problem can then be stated as

\[
\begin{align*}
a_B R P &= \lambda P \\
\lambda &= 1
\end{align*}
\]

For the eigenvector \( \lambda = 1 \) the vector \( P \) is unchanged by the rotation. What vector is unchanged by a rotation? Think of a wheel rotating—what line stays fixed?

Problem 2.12 Velocity vectors are free vectors, and must never be operated on by a \( 4 \times 4 \) \( T \) transformation matrix—only by a \( 3 \times 3 \) \( R \) matrix. Hence extract the \( a_B R \) partition from the given \( a_B T \) matrix and use that. My result—which I obtained using MATLAB—is:

\[
B V = \begin{bmatrix} -1.3400 \\ 22.3200 \\ 30.0000 \end{bmatrix}
\]

Problem 2.13 My solution for transform \( b_C T \) (obtained using the “simplified” frame diagram I’ll show in class and that you should draw) is:

\[
b_C T = \begin{bmatrix} 0.5000 & 0.7500 & 0.4330 & -6.5754 \\ -0.7500 & 0.6250 & -0.2165 & 19.7877 \\ -0.4330 & -0.2165 & 0.8750 & -28.3185 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}
\]

Problem 2.18 Consider a frame \( \{B\} \) initially coincident with frame \( \{A\} \). Perform the following operations in this sequence:

1. Rotate \( \{B\} \) around \( Z_A \) by angle \( \alpha \)
2. Rotate \( \{B\} \) around \( Y_B \) by angle \( -\beta \) (consider postmultiplying here...)
3. Translate along the direction of your choice...

Use full \( T \) matrices for these operations, but only retain the \( 3 \times 1 \) \( 4^{th} \) column of the result. The position vector I obtained is

\[
a_P = \begin{bmatrix} r \cos \alpha \cos \beta \\ r \sin \alpha \cos \beta \\ r \sin \beta \end{bmatrix}
\]
Problem 2.28 By inspection, the transform $\hat{A}C$ can be found to be

$$
\hat{A}C = \begin{bmatrix}
0 & 0.5 & 3 \\
0 & 0.866 & 0.5 \\
0 & 0 & 2
\end{bmatrix}
$$

In addition to finding $\hat{A}C$ I'd like you to also find the Z-Y-Z Euler angles $\alpha$, $\beta$, and $\gamma$ that correspond to $\hat{A}$. A partial result is

$$
\alpha = 30^\circ \quad (-150^\circ)
$$

The angle in parentheses is the result if you use the “-” sign for the square root term in the expression for angle $\beta$. The “-” results are correct but are not “minimal.”

Problem 2.38 This is an interesting problem. Note that the cosine of the angle between the two unit vectors $v_1$ and $v_2$ is given by the dot product between them. Therefore this dot product $v_1 \cdot v_2$ (also expressed as $v_1^T v_2$) is constant, even when transformed by an $R$ matrix. This observation, plus knowledge of basic matrix transpose identities, should allow you to prove the conjecture.

Programming Exercises. You need to write only two MATLAB functions (MATLAB already has an `atan2` and a matrix inverse function), saved as lower case filenames `utoi.m` and `itou.m`—following are the comments I put in my functions (i.e. their descriptions; this is what you see if you type “help utoi”). I capitalized the names of the parameters; this is not necessary. Remember, you email me these two functions; don’t hand them in.

```java
>> help utoi
IFORM = utoi(UFORM). This function converts from USER form, given by vector UFORM = [x y theta]' with x and y in meters, and theta in degrees. Result IFORM is a 3x3 matrix representing position and orientation.

>> help itou
UFORM = itou(IFORM). This function converts from INTERNAL form, given by a 3x3 matrix which represents position and orientation, to USER form, given by vector [x y theta]'. Position is in meters, with angle in degrees.
```

Using these functions in my script `chpt2.m` (available on the class website), I get:

```java
>> chpt2
Final internal form:

$$
T_{CBi} =
\begin{bmatrix}
0.7071 & -0.7071 & -10.8840 \\
0.7071 & 0.7071 & 9.3616 \\
0 & 0 & 1.0000
\end{bmatrix}
$$

Final user form:

$$
T_{CBu} =
\begin{bmatrix}
-10.8840 \\
9.3616 \\
45.0000
\end{bmatrix}
$$
```
MATLAB EXERCISE 2A. Remember, I said to use Z-Y-Z Euler angles (instead of Z-Y-X as in the text)
I wrote two MATLAB functions to do this: euler2R and R2euler. The syntax is:

```
>> help euler2R
    function R = euler2R(e)
    \MATLABs function to satisfy \MATLABs Exercise 2A, part (a); convert Z-Y-Z
    Euler angles to the corresponding rotation matrix R. Vector e should
    contain the three angles (deg).

>> help R2euler
    function [e1,e2] = R2euler(R)
    This function converts the 3x3 R matrix to corresponding Z-Y-Z Euler
    angles. Two sets of angles are generated and returned in e1 and e2.
```

These functions are based on text equations (2.72) and (2.74). Note that in equation (2.74) the only purpose of the
\(\sin \beta\) term in the equations for \(\alpha\) and \(\gamma\) is to determine the sign of \(\sin \beta\); its magnitude is irrelevant. I used the sign
function in MATLAB to obtain this sign. The reason for this is if \(\beta\) is small you don’t typically to divide by a small number.

You can then use these functions from the MATLAB command window to perform the transformations requested in
(a), (b), and (c). We’re not going to use the MATLAB Robotics Toolbox, so ignore part (d).

(a) Note that three of constraints you are asked to demonstrate are that the norm (magnitude) of each column is one.
That is, given two elements of a column \(x\) and \(y\), you could find the third element \(z\) by computing

\[
z = \pm \sqrt{1 - x^2 - y^2}
\]

You can use either the plus or minus sign, but in subsequent columns you must select the sign by requiring the dot
products of each column to be zero. The three dot product constraints and the three magnitude constraints total to
six constraints overall.

Don’t email me these functions, just the Programming Exercises.