Introdução a Robótica

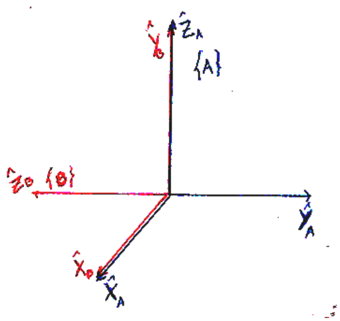
Jhonatan da Silva

Instituto Federal de Santa Catarina. Janeiro 21, 2017

Sumário

1 Matrizes de Rotação e Translação

Quando temos um frame de referência A e outro frame de referência B e queremos saber qual relação por exemplo do eixo x em A, $\hat{X}^A$ com o eixo x em B, $\hat{X}^B$, como na imagem abaixo,



podemos utilizar de matrizes de rotação, geralmente uma matriz de rotação R, tem a seguinte forma geral.

$$R := \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

Assim com matrizes de rotação podemos escrever a relação entre $\hat{X}^A$ e $\hat{X}^B$ em termos de ${}^A_B R$, onde

$${}^A\hat{X}_B \;=\; {}^A_B R \; {}^B\hat{X}_B$$

e ${}^A_B R$ é definido como

$${}^A_B R \;=\; \begin{bmatrix} {}^A\hat{X}_B & {}^A\hat{Y}_B & {}^A\hat{Z}_B \end{bmatrix}$$

Nos resta determinar ${}^A\hat{X}_B \;\; {}^A\hat{Y}_B \;\; {}^A\hat{Z}_B$,

$${}^A\hat{X}_B = \begin{bmatrix} \hat{X}_B\hat{X}_A \\ \hat{X}_B\hat{Y}_A \\ \hat{X}_B\hat{Z}_A \end{bmatrix}$$

1.1 Computação simbólica com python

Como faríamos se quiséssemos definir uma matriz simbólica tal qual igual a R, Em uma linguagem de programação como python ? O python possui uma biblioteca em específico chamada sympy que permite que o usuário compute matemática simbólica, um simples código segue abaixo,

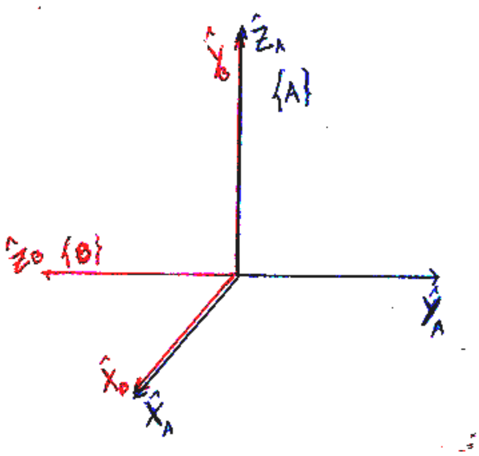
```python
from sympy import symbols,Matrix

(r11,r12,r13,
r21,r22,r23,
r31,r32,r33) = symbols('r11 r12 r13 r21 r22 r23 r31 r32 r33')

R = Matrix([[r11,r12,r13],[r21,r22,r23],[r31,r32,r33]])
```

Que retornará

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

1.2 Definindo a matriz de rotação

Definindo os dois frames de referência pela imagem abaixo



Como poderiamos determinar ${}^A_BR$ tal que $\hat{X}^A \mapsto \hat{X}^B$ Pela definição anterior de que

$$ {}^A_BR \;=\; \begin{bmatrix} {}^A\hat{X}_B & {}^A\hat{Y}_B & {}^A\hat{Z}_B \end{bmatrix} $$

Pela matriz podemos ver que cada coluna na verdade é a cordenada de B em relação a A, como vemos na imagem as coordenadas de $\hat{X}^A$ e $\hat{X}^B$ são as mesmas, já, $\hat{Y}^B$ tem coordenadas em $\hat{Z}^A$ enquanto $\hat{Z}^B$ tem coordenadas em $\hat{Y}^A$. Então podemos escrever ${}^A_BR$ como

$$ {}^A_BR = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} $$

E como R é ortonormal sabemos que ${}^A_BR^T = {}^B_AR = {}^A_BR^{-1}$

1.3 Exemplo de rotação com python

Quando lida-se com computação numérica ao contrário da simbólica usa-se o numpy, usualmente importa-se o numpy como np como se vê no código abaixo

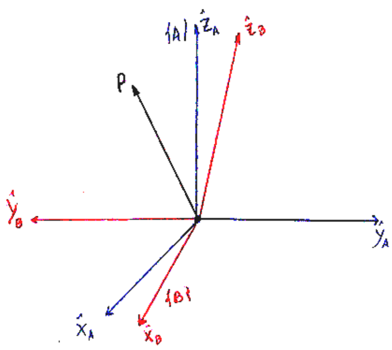
```
import numpy as np

R = np.matrix('1 0 0; 0 0 -1;0 1 0')
```

Que retornará

$$ R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} $$

1.4 Rotação geral

E quando temos um ponto P em termos de B e queremos saber como definir o mesmo em termos de A, por exemplo na imagem

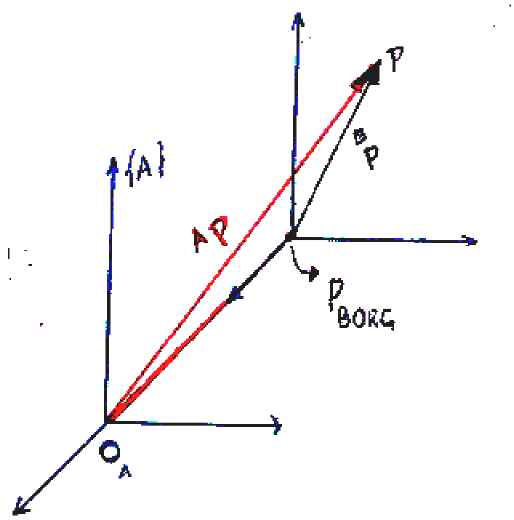


Podemos definir o ponto P em A como ${}^AP$ assim temos

$$ {}^AP = {}^A_BR^BP $$

1.5 Translações

Quando temos um ponto P em um frame A e queremos transladá-lo sem rotacionar ou mudar de frame como mostra a imagem a seguir

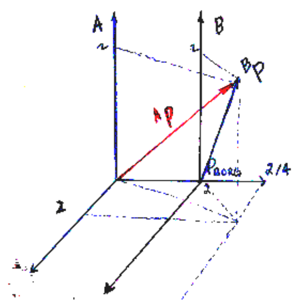


Podemos definir ${}^{A}P$ como

$$ {}^{A}P = {}^{B}P + {}^{A}P_{BORG} $$

1.6 Exemplo de translação em python

Como nenhum exemplo prático foi definido na seção anterior antes de fazer o código precisamos de algumas definições, segundo a imagem



Então temos o primeiro frame de referência A e o segundo B, que está deslocado duas casas no eixo y, então nosso ponto de origem para B é $(0, 2, 0)$, pode-se notar que em relação ao frame B, a posição y é 2 mas em relação a A é 4, agora temos algumas definições

$$ {}^{B}P = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix} $$

$$ P_{BORG} = \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix} $$

$$ {}^{A}P = Pb + Pborg $$

E agora podemos fazer o código para computar ${}^{A}P$

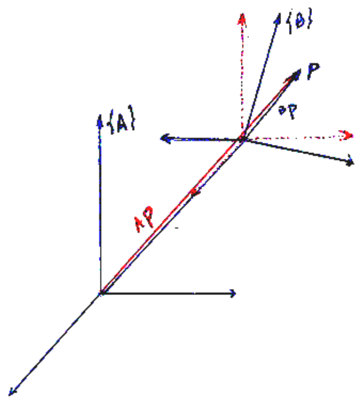
```python
import numpy as np

Pborg = np.matrix('0; 2; 0')
Pb = np.matrix('2; 2; 2')

Pa = Pb + Pborg
```

Que irá retornar

$$^{A}P = \begin{bmatrix} 2 \\ 4 \\ 2 \end{bmatrix}$$

1.7 Transformação Geral

Se temos um caso onde foi feita uma translação e uma rotação em relação a um novo frame, e queremos saber qual é o novo ponto em relação ao frame original, como na imagem abaixo



Primeiro temos que fazer a matriz de rotação ${}^{A}_{B}R$ pois o novo ponto está em um frame diferente do original, então não podemos apenas fazer a soma, depois de fazer ${}^{A}_{B}R$ aplicado em ${}^{B}P$ podemos fazer a translação somando com ${}^{A}P_{BORG}$, em uma forma geral podemos escrever

$$^{A}P = {}^{A}_{B}R\,{}^{B}P + {}^{A}P_{BORG}$$

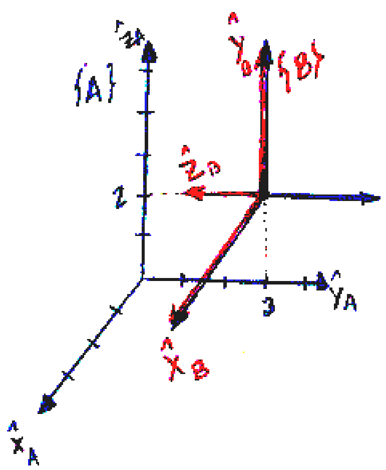
1.7.1 A Transformação Homogênea

Seria interessante ter uma matriz que resolvesse o problema geral, onde temos a rotação e a translação, porque computacionalmente falando, é mais interessante ter uma multiplicação de uma matriz com um vetor que ter várias operações. A transformação homogênea resolve esse problema, expandindo para uma matriz $(4, 4)$ podemos reescrever a expressão ${}^{A}P = {}^{A}_{B}R\,{}^{B}P + {}^{A}P_{BORG}$ em forma matricial

$$\begin{bmatrix} ^{A}P \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{A}_{B}R & {}^{A}P_{BORG} \\ 00 & 0 \end{bmatrix} \begin{bmatrix} ^{B}P \\ 1 \end{bmatrix}$$

1.7.2 Exemplo

Um exemplo prático de como funciona a transformação homogênea, pela imagem abaixo,



Temos um frame de referência A e outro B, pelo que se ve na image o frame de referêcia continuou no mesmo plano xy, deslocou 3 casas em direção ao eixo y e 2 em relação ao eixo z, definindo a translação como q temos que

$$q = \begin{bmatrix} 0 \\ 3 \\ 2 \end{bmatrix}$$

Agora precisamos definir ${}^{A}_{B}R$, que tem a mesma rotação do primeiro exemplo de rotações

$${}^{A}_{B}R = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

Agora podemos definir uma transformação homogênea ${}^{A}_{B}T$ de tal modo

$${}^{A}_{B}T = \begin{bmatrix} {}^{A}_{B}R & {}^{A}P_{BORG} \\ 00 & 0 \end{bmatrix}$$

$${}^{A}_{B}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Se definirmos um ponto ${}^{B}P$

$${}^{B}P = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Onde a quarta cordenada de ${}^{B}P$ não é levada em conta no final. Podemos saber o novo ponto ${}^{A}P$ fazendo

$${}^{A}P = {}^{A}_{B}T\,{}^{B}P$$

$${}^{A}P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad {}^{A}P = \begin{bmatrix} 0 \\ 2 \\ 3 \\ 1 \end{bmatrix}$$

$$\therefore \quad {}^{A}P = \begin{bmatrix} 0 \\ 2 \\ 3 \end{bmatrix}$$

1.8 Exemplo de transformação homogenea com python

Precisamos definir a matriz de transformação ${}^{A}_{B}T$ o ponto ${}^{B}P$ e agora que temos essas duas informações podemos computar ${}^{A}P$ com a multiplicação dos dois com $np.dot({}^{A}_{B}T, {}^{B}P)$, e do resultado final pegar apenas as 3 primeiras coordenadas que resultam no ponto desejado.

```
import numpy as np

T = np.matrix('1 0 0 0; 0 0 -1 3; 0 1 0 2; 0 0 0 1')

Pb = np.matrix('0;1;1;1')

Pa = np.dot(T,Pb)

Pa = Pa[0:3]
```

O código retornará

$${}^{A}P = \begin{bmatrix} 0 \\ 2 \\ 3 \end{bmatrix}$$