Introduction to Robotics

Manipulation and Programming

Unit 2: Kinematics

DYNAMICS – DIFFERENTIAL KINEMATICS

DR. ERIC CHOU

IEEE SENIOR MEMBER

Objectives

- Robot Motion Modelling
- Differential kinematics relates the velocities of the manipulator components.
- Differential forms of the homogeneous transformation can be used to examine the pose velocities of frames. This method will be compared to a conventional dynamics vector approach.
- The Jacobian matrix is used to map motion between join and Cartesian space, an essential operation when curvilinear robot motion is required in applications such as welding or assembly.

Differential Kinematics

SECTION 1

Factors to Determine the Differential Kinematics

- Frame to frame transformation
- Angular velocity
- Displacement velocity

Differential Transformations

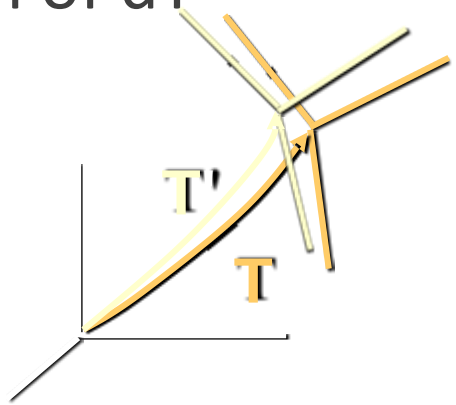
T frame transformation

- Now if we have a frame T relating a set of axes (primed axes) to global or base axes, then a small differential displacement of these axes (dp, $dq = dq\ k$) relative to the base axes results in a new frame $T' = T + dT = H(dq, dp)\ T$.

- Although finite rotations can't be considered vectors, differential (infinitesimal) rotations can, thus $dq = dq\ k$. Thus, the form of dT can be expressed as

$$dT = (H(dq, dp) - I)\ T = D\ T$$

(in base frame)

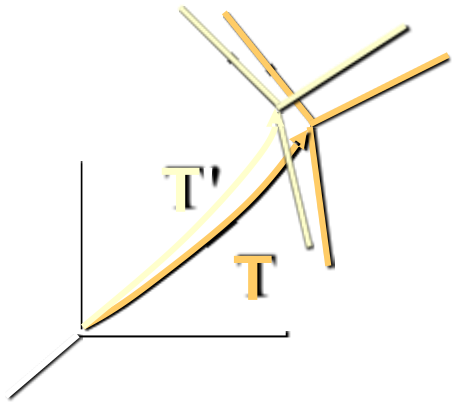


Differential Transformations

- Displacements made relative to the primed axes (dp, dq expressed in primed axes) change the form of dT to

$$dT = T\,(H(dq, dp) - I) = T^T\ D$$

(in primed frame)



Differential Transformations

- Since $dT = DT = T^T D$, we can relate the differential transformation in either frame, given the other, as

$$D = \boldsymbol{T^T}\ D\ \boldsymbol{T^{-1}} \quad \textbf{or} \quad {}^{T}D = T^{-1}\ D\ T$$

- Letting $sin(dq)$ be dq, $cos(dq)\ be\ 1$, the form of $D\ (or\ D\ )$ can be shown to be

$$\Delta(d\theta, dp) = \begin{bmatrix} 0 & -k_z d\theta & k_y d\theta & dp_x \\ k_z d\theta & 0 & -k_x d\theta & dp_y \\ -k_y d\theta & k_x d\theta & 0 & dp_z \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Velocity Matrix

SECTION 1

Screw Velocity Matrix

- In the case of pure rotation, the differential transformation, depicted by D (or ${}^{T}D$), reduces to a screw rotation about the unit vector k. Taking the derivative of the matrix we get the screw (angular) velocity matrix:

$$\Omega = \begin{bmatrix} 0 & -\omega_z & \omega_y & 0 \\ \omega_z & 0 & -\omega_x & 0 \\ -\omega_y & \omega_x & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Differential Transformations

- Given the relations

$$D = \boldsymbol{T}^{\boldsymbol{T}} \; D \; \boldsymbol{T}^{\boldsymbol{-1}} \quad \textbf{or} \quad {}^{T}D = T^{-1} \, D \, T$$

- we can relate the differential transformations relative to the base frame or relative to the offset frame T where T is known as

$$T = \begin{bmatrix} \boldsymbol{a} & \boldsymbol{b} & \boldsymbol{c} & \boldsymbol{p} \\ a_x & b_x & c_x & p_x \\ a_y & b_y & c_y & p_y \\ a_z & b_z & c_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Differential Transformations

- Define the differential rotations and translations relative to the base frame by

$$d = d_x \boldsymbol{I} + d_y \boldsymbol{J} + d_z \boldsymbol{K}$$

$$\boldsymbol{d} = \boldsymbol{d}_x \boldsymbol{I} + \boldsymbol{d}_y \boldsymbol{J} + \boldsymbol{d}_z \boldsymbol{K}$$

where

$$d_x = \boldsymbol{d}q_x \qquad \boldsymbol{d}_x = \boldsymbol{d}p_x$$

$$d_y = \boldsymbol{d}q_y \qquad \boldsymbol{d}_y = \boldsymbol{d}p_y \quad \textbf{...giving ...}$$

$$d_z = \boldsymbol{d}q_z \qquad \boldsymbol{d}_z = \boldsymbol{d}p_z$$

$$D = \begin{bmatrix} 0 & -d_z & d_y & d_x \\ d_z & 0 & -d_x & d_y \\ -d_y & d_x & 0 & d_z \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Differential Transformations

Define the differential rotations and translations relative to the base frame by

$$\delta = \delta_x \mathbf{I} + \delta_y \mathbf{J} + \delta_z \mathbf{K} \qquad \Delta = \quad \delta$$

$$\mathbf{d} = \mathbf{d}_x \mathbf{I} + \mathbf{d}_y \mathbf{J} + \mathbf{d}_z \mathbf{K}$$

where

$$\delta_x = d\theta_x \qquad \mathbf{d}_x = \mathbf{dp}_x$$

$$\delta_y = d\theta_y \qquad \mathbf{d}_y = \mathbf{dp}_y \text{ ...giving ...}$$

$$\delta_z = d\theta_z \qquad \mathbf{d}_z = \mathbf{dp}_z$$

$$\Delta = \begin{bmatrix} 0 & -\delta_z & \delta_y & d_x \\ \delta_z & 0 & -\delta_x & d_y \\ -\delta_y & \delta_x & 0 & d_z \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Differential Transformations

- Define the differential rotations and translations relative to the offset frame by

$$d' = d_x, \boldsymbol{i} + d_y, \boldsymbol{j} + d_z, \boldsymbol{k} \quad \textbf{(Projection: Inner Product)}$$

$$\boldsymbol{d'} = \boldsymbol{d_x}, \boldsymbol{i} + \boldsymbol{d_y}, \boldsymbol{j} + \boldsymbol{d_z}, \boldsymbol{k} \quad \textbf{(Leverage: Cross Product)}$$

- It can be shown (see notes) that the differential displacements in the offset frame can be related to those in the base frame by

$$\begin{aligned} d_x, &= d \cdot \boldsymbol{a} = d^T \boldsymbol{a} \\ d_y, &= d \cdot \boldsymbol{b} = d^T \boldsymbol{b} \\ d_z, &= d \cdot \boldsymbol{c} = d^T \boldsymbol{z} \end{aligned} \qquad \begin{aligned} \boldsymbol{d_x}, &= d \cdot (\boldsymbol{p} \times \boldsymbol{a}) + \boldsymbol{d} \cdot \boldsymbol{a} \\ \boldsymbol{d_y}, &= d \cdot (\boldsymbol{p} \times \boldsymbol{b}) + \boldsymbol{d} \cdot \boldsymbol{b} \\ \boldsymbol{d_z}, &= d \cdot (\boldsymbol{p} \times \boldsymbol{c}) + \boldsymbol{d} \cdot \boldsymbol{c} \end{aligned}$$

Velocities

- Vector w can be resolved into components p in the base frame by the equation $p = T\,w$. Now under a small displacement, frame T can be expressed by the new frame

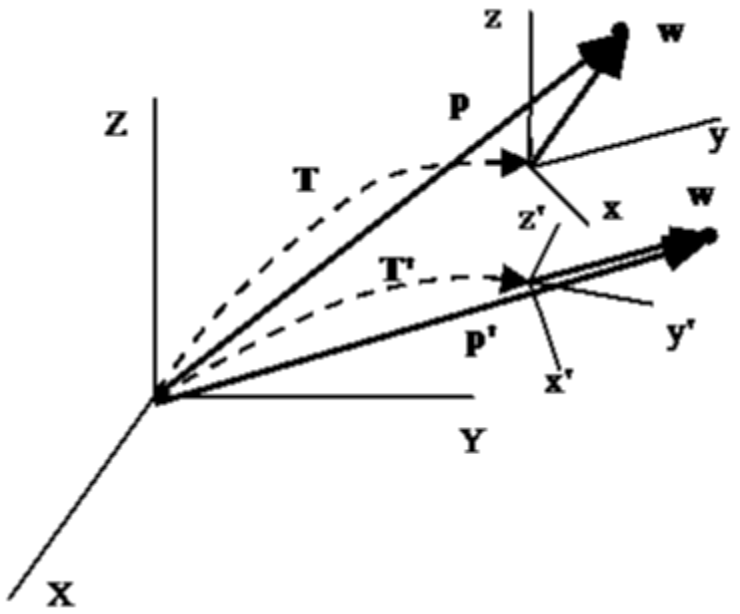
$$\boldsymbol{T' = T + dT = T + \Delta T}$$

- A vector w in frame T, once perturbed, can be located globally by p' such that

$$\boldsymbol{p' = (T + DT)\,w}$$

- The delta move, p' - p, becomes

$$\boldsymbol{p' - p = (T + DT)\,w - T\,w = DT\,w}$$



Velocities

Dividing by Δt and taking the limit as Δt→ 0 (take derivative), we determine the velocity in the base frame:

$$V = \Delta^{\&} T\, w$$

where

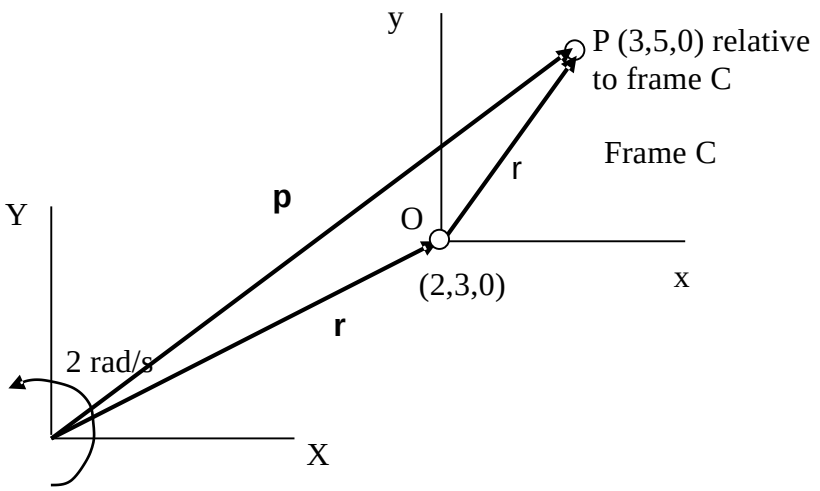
$$\Delta^{\&} = \begin{bmatrix} 0 & -k_z d\theta & k_y d\theta & dp_x \\ k_z d\theta & 0 & -k_x d\theta & dp_y \\ -k_y d\theta & k_x d\theta & 0 & dp_z \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\omega_z & \omega_y & v_x \\ \omega_z & 0 & -\omega_x & v_y \\ -\omega_y & \omega_x & 0 & v_z \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Case Study

SECTION 1

Example

- A point P is located by vector $\boldsymbol{r}$ in an offset frame C as shown. At this moment, the frame C origin is being translated by the velocity $\boldsymbol{v}$ = 2 i + 2 j m/s relative to the base frame. The frame is also being rotated relative to the base frame by the angular velocity $\boldsymbol{\omega}$ = 2 rad/s K where i, j, k are the unit vectors for the base frame.
- Determine the instantaneous velocity of point P using both the conventional vector dynamics approach and the differential methods in this chapter.



Example: Conventional Solution

$$v = v_0 + \omega \times \rho$$

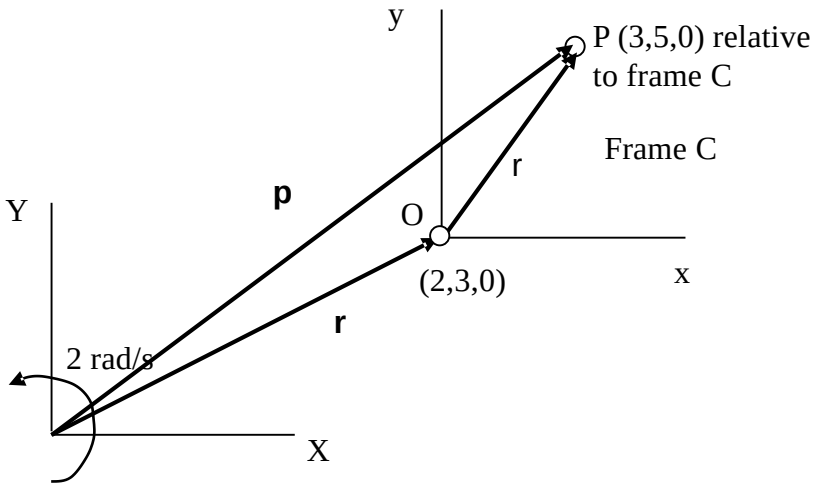
- where $v_0 = v_t + \omega \times r$. At this instant note that frame **C** is aligned with the base frame so that the unit vectors are parallel.
- It can be shown that $\omega \times r = -6i + 4j$ so that

$$v = 2i + 2j - 6i + 4j$$

$$= -4i + 6j \text{ m/s}$$

- It can be shown that $\omega x \rho = -10i + 6j$, so that

$$v = -4i + 6j - 10i + 6j = -14i + 12j \text{ m/s} \quad \text{(answer)}$$



Example: differential solution

- Apply $v = \Delta T w$
- where w = ρ and $\Delta = \begin{bmatrix} 0 & -2 & 0 & 2 \\ 2 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$, $T = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
- to give $v = \begin{bmatrix} -14 \\ 12 \\ 0 \\ 0 \end{bmatrix}$ m/s
- Note that the same values are obtained!

Python Coding

SECTION 1

These Matrix are used as HTM

Demo Program: differential.py

- Δ is the angular velocity matrix
- T is the c frame orientation matrix
- w is the point in c frame (in HTM format)

$$v = \Delta\, T\, w$$

```
import numpy as np
E = 10    # round of 10 decimal points

D = [[0, -2, 0, 2],
     [2,  0, 0, 2],
     [0,  0, 0, 0],
     [0,  0, 0, 0],
     ]

D = np.array(D)
D = D.reshape(4, 4)
D = np.ndarray.round(D, E)

T = [[1,  0, 0, 2],
     [0,  1, 0, 3],
     [0,  0, 1, 0],
     [0,  0, 0, 1],
     ]
T = np.array(T)
T = T.reshape(4, 4)
T = np.ndarray.round(T, E)
```

```
DT = np.dot(D, T)
DT = np.ndarray.round(DT, E)

w = [3, 5, 0, 1]
w = np.array(w)
w = w.reshape(4, 1)

Tw  = np.dot(T, w)
DTw = np.dot(D, Tw)
DTw = np.ndarray.round(DTw, E)
print("DTw:")
print(DTw)
print()
```