Introduction to Rigid-Body Motion

Yuzhe Qin
Physics-based Learning System
Toward Better Modeling

How to represent the spatial relationship of robot and objects?

Wang et al., “DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion”, CVPR 2019
How to command the robot to execute the desired motion?
Topics

• Rotation and SO(3)

• Rotation Parameterizations

• Learning to Predict Rotation by NN

• Homogenous Transformation and SE(3)
Rigid-Body in 3D Space

Point: Position 3-Dimensional

Rigid-Body: Position + Orientation 6-Dimensional
Represent Orientation

• The relationship of two frames define orientation

• Orientation is defined via two frames:
  
  • Space Frame: \( \{s\} = \{\hat{x}_s, \hat{y}_s, \hat{z}_s\} \)
  
  • Body Frame: \( \{b\} = \{\hat{x}_b, \hat{y}_b, \hat{z}_b\} \)
  
  • \( R_{sb} = [\hat{x}_{sb}, \hat{y}_{sb}, \hat{z}_{sb}] \) is called a rotation matrix
Property of Rotation Matrix

• Special Orthogonal Group $SO(n)$ for $R^n$ space:

  - $SO(n) = \{ R \in R^{n\times n} : \det(R) = 1, RR^T = I \}$

• Degree of freedom of $SO(n)$ is $\frac{n(n-1)}{2}$

• Standard property of Group:
  - Associativity, closure, identity element, inverse element

• Change reference frame:
  - $p_s = R_{sb} p_b$, where $p \in R^3$
Topics

- Rotation and SO(3)
- Rotation Parameterizations
- Learning to Predict Rotation by NN
- Homogenous Transformation and SE(3)
Representation I: Euler Angle

• Euler Angle Definition:
  1. Body frame \(\{b\}\) coincident with space frame \(\{s\}\) on the beginning
  2. Rotate \(\{b\}\) about one axis
  3. Rotate \(\{b\}\) about another axis, different from first one
  4. Finally rotate \(\{b\}\) about another axis in step 2

• E.g. ZYZ Euler Angle
Euler Angle Parameterization

• Use three angles for rotation about principal axes:
  • Euler Angle: rotate about one axis, then another and then the first
  • Tait-Bryan Angle: rotate about all three axes

• Rotation about fixed axis or body axis?
  • Extrinsic axes: the principal axes is global: denote \(s\) (static)
  • Intrinsic axes: the principal move with rotation: denote \(r\) (rotating)

• E.g. axes=‘szyx’:
  • First global z, then global y and then global x axis
Euler Angle to Rotation Matrix

• Rotation about principal axis is represented as:

\[ R_x(\theta) \triangleq \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \]

\[ R_y(\theta) \triangleq \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \]

\[ R_z(\theta) \triangleq \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

• E.g. axes='rxyz':
  • \( R_{sb} = R_x(\theta_1)R_y(\theta_2)R_z(\theta_3) \)
Euler Angle Singularity

- Singularity: for a $R \in SO(3)$, there is more than one Euler angle to represent.

- Euler Angle is not unique for some rotation matrix:

\[
R = R_z(90^\circ) R_y(90^\circ) R_x(90^\circ) = \begin{bmatrix}
0.000 & 0.000 & 1.000 \\
0.000 & 1.000 & 0.000 \\
-1.000 & 0.000 & 0.000 \\
\end{bmatrix}
\]

\[
R = R_z(45^\circ) R_y(90^\circ) R_x(45^\circ) = \begin{bmatrix}
0.000 & 0.000 & 1.000 \\
0.000 & 1.000 & 0.000 \\
-1.000 & 0.000 & 0.000 \\
\end{bmatrix}
\]
Gimbal Lock

• Singularity cause discontinuity:
  • Whether two Euler Angle are same?
  • Is difference between two Euler Angle meaningful?
  • Unsatisfied for numerical approach

• Gimbal Lock (±90 degree pitch singularity):
  • Roll and pitch are not stable values
  • Only a problem if recover Euler Angle from rotation matrix
Representation II: Axis-Angle

• Euler Theorem:
  • Any rotation in $R^3$ is equivalent to rotation about a fixed axis $\hat{\omega} \in R^3$ through an positive angle $\theta$
  • $\hat{\omega}$: unit vector of rotation axis
  • $\theta$: angle of rotation
  • $R \in SO(3) \triangleq Rot(\hat{\omega}, \theta)$
• Consider a point \( q \) in body frame. At time \( t = 0 \), the position is \( q_0 \)

• Rotate \( q \) with unit angular velocity around axis \( \hat{\omega} \):
  • \( v = \hat{\omega} \times r \)
  • \( \dot{q}(t) = \hat{\omega} \times q(t) = [\hat{\omega}]q(t) \)
Skew-Symmetric Matrix

• $A$ is skew-symmetric $A = -A^T$

• Skew-symmetric matrix operator:

\[
\omega = \begin{bmatrix} \omega_1 \\ \omega_2 \\ -\omega_3 \end{bmatrix}, [\omega] \triangleq \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}
\]

• Cross product can be a linear transformation:
  • $a \times b = [a]b$

• Lie Algebra of 3d rotation: $so(3) \triangleq \{S \in R^{3\times3} : S^T = -S\}$
Axis-Angle Conversion

• Rotate $p$ with unit angular velocity around axis $\hat{\omega}$:
  • $\hat{v} = \hat{\omega} \times r$
  • $\dot{p}(t) = \hat{\omega} \times p(t) = [\hat{\omega}]p(t)$

• Solution of this ODE: $p(t) = e^{[\hat{\omega}]t}p_0$

• For unit angular velocity, $\theta(t) = t$, $\theta$ is the total rotation angle
  • So $p(\theta) = e^{[\hat{\omega}]\theta}p_0$

• It means that $Rot(\hat{\omega}, \theta) = e^{[\hat{\omega}]\theta}$ (exponential map)

• $\bar{\omega} = \hat{\omega} \theta$ also called rotation vector or exponential coordinate
Axis-Angle to Rotation

- Definition of Matrix Exponential:

  \[ e^{[\mathbf{\omega}]\theta} = I + \theta \mathbf{\omega} + \frac{\theta^2}{2!} [\mathbf{\omega}]^2 + \frac{\theta^3}{3!} [\mathbf{\omega}]^3 + \cdots + \frac{\theta^n}{n!} [\mathbf{\omega}]^n, n \to +\infty \]

- Sum of infinite series? Rodrigues Formula

  - Can prove \([\mathbf{\omega}]^3 = -[\mathbf{\omega}]\]

  - Then use Taylor expansion of \(\sin\) and \(\cos\)

    \[ e^{[\mathbf{\omega}]\theta} = I + [\mathbf{\omega}] \sin(\theta) + [\mathbf{\omega}]^2 (1 - \cos(\theta)) \]

Exponential: \([\mathbf{\omega}]\theta \in \text{so}(3) \quad \rightarrow \quad R \in \text{SO}(3)\)
Rotation to Axis-Angle

• Is Exponential Coordinate unique?
  • \( \theta \in [0, \pi) \), otherwise simple reverse the axis and angle
  • \( R \neq I \), otherwise \( \theta = 0 \) and \( \hat{\omega} \) can be any direction
  • \( tr(R) \neq -1 \), otherwise \( \theta = \pi \) and \( \hat{\omega} \) have three possible solutions

• Then, a unique inverse mapping exist:
  \[
  \theta = \cos^{-1}\left(\frac{1}{2}(tr(R) - 1)\right)
  \]
  \[
  [\hat{\omega}] = \frac{1}{2 \sin(\theta)}(R - R^T)
  \]

Logarithm: \( R \in SO(3) \) \( \rightarrow \) \([\hat{\omega}]\theta \in so(3)\)
Representation III: Quaternion

• Recall the complex number $a + bi$

• Quaternion is more generalized complex number:
  \[ q = w + xi + yj + zk \]

  • $w$ is the real part and $\tilde{q} = (x, y, z)$ is the imaginary part

• Imaginary: $i^2 = j^2 = k^2 = ijk = -1$

• Anticommutative: $ij = k = -ji, jk = i = -kj, ki = j = -ik$
Quaternion Property

• In vector-form, product of two quaternion:

\[ \begin{align*}
q_1 &= (w_1, \vec{q}_1), q_2 = (w_2, \vec{q}_2) \\
q_1q_2 &= (w_1w_2 - \vec{q}_1^T\vec{q}_2, \omega_1\vec{q}_2 + \omega_2\vec{q}_1 + \vec{q}_1 \times \vec{q}_2)
\end{align*} \]

• Norm:

\[ \|q\| = w^2 + \vec{q}^T\vec{q} = qq^* = q^*q \]

• Conjugate:

\[ q^* = (w, -\vec{q}) \]

• Quaternion Unit:

\[ q_0 = (1,0,0,0) \]

• Inverse:

\[ q^{-1} \triangleq \frac{q^*}{\|q\|^2} \]
Quaternion Conversions

• A **unit** quaternion \(||q|| = 1\) can represent rotation

• Exponential coordinates \(\vec{\omega}\theta\) to unit quaternion:
  \[
  q = \left[\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right) \hat{\omega}\right]
  \]

• Unit quaternion to axis-angle representation:
  \[
  \theta = 2 \cos^{-1}(\omega), \quad \hat{\omega} = \begin{cases} \frac{1}{\sin\left(\frac{\theta}{2}\right)} \vec{q}, & \theta \neq 0 \\ 0, & \theta = 0 \end{cases}
  \]

• Obtain rotation matrix from unit quaternion:
  \[
  R(q) = E(q)G(q)^T, \quad E(q) = [-\vec{q}, wI + [\vec{q}]], \quad G(q) = [-\vec{q}, wI - [\vec{q}]]
  \]
Quatetion as Rotation

• Represent orientation:

• Change reference frame for $p_b \in R^3$ in body frame:
  1. Construct a purely imaginary quaternion $p = (0, \vec{p})$
  2. Quaternion form body to world: $q_{sb} = (\omega, \vec{q})$
  3. $\vec{P}_s = Im(q_{sb}pq_{sb}^*) = (\omega^2 - ||\vec{q}||^2)\vec{p} + 2(\vec{q}^T\vec{p})\vec{q} + 2\omega(\vec{q} \times \vec{v})$

• Composing rotation:
  • Similar as rotation matrix, whichever quaternion is on the right is the rotation that is performed first:

    \[ q_{ab}q_{bc}pq_{c}^*q_{ab}^*q_{bc} = p_a \]
More About Quaternion

• **No singularity** with $SO(3)$
  • No singularity if we require the real part to be positive
  • Embeds a 3-D space into a 4-D space with unit norm constraint

• **Singularity** with axis-angle at $\theta = 0$
  • Many-to-one from axis-angle to quaternion

• Quaternion is computationally cheap:
  • Internal representation of Physical Engine and Robot

• Pay attention to convention $(w, x, y, z)$ or $(x, y, z, w)$?
  • $(w, x, y, z)$: SAPIEN, transforms3d, Eigen, blender, MuJoCo, V-Rep
  • $(x, y, z, w)$: ROS, Physx, PyBullet
## Rotation Representation Sum.

<table>
<thead>
<tr>
<th></th>
<th>Form</th>
<th>Change Frame?</th>
<th>Inverse?</th>
<th>Composing?</th>
<th>No Singularity?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation Matrix</td>
<td>$SO(3)$</td>
<td></td>
<td></td>
<td></td>
<td>N/A</td>
</tr>
<tr>
<td>Euler Angle</td>
<td>$R^3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rotation Vector</td>
<td>$R^3$</td>
<td></td>
<td></td>
<td></td>
<td>?</td>
</tr>
<tr>
<td>Skew-symmetrical Matrix</td>
<td>$so(3)$</td>
<td></td>
<td></td>
<td></td>
<td>?</td>
</tr>
<tr>
<td>Quaternion</td>
<td>$R^4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• ? means no singularity with single exceptions
Topics

- Rotation and SO(3)
- Rotation Parameterizations
- Learning to Predict Rotation by NN
- Homogenous Transformation and SE(3)
Predict Rotation via Neural Network

Learn to predict rotation via supervised learning?
  • 3D Shape Pose Estimation\(^1\)
  • Human Pose Estimation\(^2\)
  • Grasping Pose Prediction\(^3\)

Yuan et al., “BigHand2.2M Benchmark: Hand Pose Dataset and State of the Art Analysis”, CVPR 2017
Pas et al., “DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion”, IJRR 2017
Rotation Regression: Loss and Representation

• We need to choose the representation and loss

• This is not a trivial task, simply combining loss and representation will fail

• E.g., Euler Angle + L2 Loss is not a good choice
  • Prediction: \( \left( \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2} \right) \), Ground-Truth: \( \left( \frac{\pi}{4}, \frac{\pi}{2}, \frac{\pi}{4} \right) \), but they are the same

• Representation should be chosen based on loss function
Distance Metric on $SO(3)$

• How far it is between two rotations?
  • L2 distance of rotation representation?

• Remember metric in mathematics:
  • $d(x, y) > 0$
  • $d(x, y) = 0 \iff x = y$
  • $d(x, y) = d(y, x)$
  • $d(x, y) < d(x, z) + d(z, y)$

• Can we find a metric to define the distance in $SO(3)$
Loss: Distance Metric on SO(3)

• Key idea: use the relative transformation of two rotation

• How much angle from $R_{pred}$ to $R_{gt}$?
  • First compute the relative rotation, $R_{loss} = R_{gt}^T R_{pred}$
  • Then compute the axis-angle representation for rotation matrix, only the angle is used in loss term

• $\theta_{loss} = \log(R_{gt}^T R_{pred}) = \cos^{-1}\left(\frac{\text{tr}(R_{loss})-1}{2}\right)$,

• It’s easy to prove that this quantity is a metric
An Empirically-good Representation/Loss

• Representations for the 3D rotations are **discontinuous** in 4 or fewer dimensions

• Should use redundant representation for rotation

• E.g. 5D/6D continuous representation for 3D rotations:
  • The first two column (for 6D case) of rotation matrix

\[
\begin{bmatrix}
r_{11} & r_{12} & r_{13} \\
r_{21} & r_{22} & r_{23} \\
r_{31} & r_{32} & r_{33}
\end{bmatrix}
\quad \text{or} \quad
\begin{bmatrix}
r_{11} & r_{12} & r_{13} \\
r_{21} & r_{22} & r_{23} \\
r_{31} & r_{32} & r_{33}
\end{bmatrix}
\]

• Loss for this representation: just L2 loss

Zhou et al., “On the Continuity of Rotation Representation in Neural Networks”, ICCV 2017
Training Rotation Regression Network

- Regress 6D rotation representation on the last layer
  - $r_{11}, r_{21}, r_{31}, r_{12}, r_{22}, r_{32}$
- Denote $a_1 = [r_{11}, r_{21}, r_{31}]^T, a_2 = [r_{12}, r_{22}, r_{32}]^T, B = [b_1, b_2, b_3] \in SO(3)$

\[
\begin{pmatrix}
\vdots & \vdots & \vdots \\
 a_1 & a_2 & \\
\vdots & \vdots & \vdots
\end{pmatrix}
\text{operation}
\rightarrow
\begin{pmatrix}
\vdots & \vdots & \vdots \\
 b_1 & b_2 & b_3 \\
\vdots & \vdots & \vdots
\end{pmatrix}
\]

1. Normalize the first column, denote as $N(a_1) = b_1$
2. Gram-Schmidt Orthogonalization: $N(a_2 - (b_1 \cdot a_2)b_1) = b_2$
3. Cross product for third column: $b_1 \times b_2 = b_3$
4. Calculate loss based on the L2 loss of 6D representation
5. Update network parameters
Classification in Rotation Prediction

• Classification can always be a simpler choice
• Fine-grained space with discretized angles\(^1\)
  • E.g., divide rotation into 360 section
  • Using Bin classification for rotation prediction
• Classification method not suffer from loss issues

Topic

• Rotation and SO(3)
• Rotation Parameterizations
• Learning to Predict Rotation by NN
• Homogenous Transformation and SE(3)
Rigid-Body Configuration and SE(3)

• General rigid-body configuration includes both position $p \in R^3$ and rotation $R \in SO(3)$.

• Special Euclidean Group for 3D Space:
  • $SE(3) \triangleq \{T = \begin{bmatrix} R & p \\ 0_{1 \times 3} & 1 \end{bmatrix}, R \in SO(3), p \in R^3\}$

• $T \in SE(3)$ called homogenous transformation matrix. Similar to rotation, $T$ can represent pose of a rigid-body and change reference frame of point $p \in R^3$

• Composing: $T_{ab}T_{bc} = T_{ac}$
Homogenous Coordinates

• Homogeneous Coordinate for 3D Space:
  • \( \tilde{p} \triangleq \begin{bmatrix} p \\ 1 \end{bmatrix} \in \mathbb{R}^4 \)

• Homogeneous coordinates make it easier when changing reference frame:
  • \( p_s = R_{sb} p_b + p_{sb} \), where \( p_{sb} \) is the position of frame \( \{b\} \) origin in frame \( \{s\} \). The formula below is equivalent:

\[
\tilde{p}_s = T_{sb} \tilde{p}_b
\]
Geometrical Interpretation of Rigid-body Motion

- Screw motion:
  - Any rigid body motion is equivalent to rotating about one axis while also translating along axis
  - The axis may not pass the origin
Geometrical Interpretation of Rigid-body Motion

- How to find a screw motion for $T \in SE(3)$
  1. Find the axis and rotation angle based on rotation $R \in SO(3)$
  2. Now, we can only change the position of this axis and translation along this axis
  3. Translation: Along axis (1-DoF) and on rotation plane(2-DoF)
Exponential Map of Rigid-body Motion

- Differential equation of rigid-body motion:
  - \( \dot{p}(t) = \omega \times (p(t) - q) + v = [\omega]p(t) - \omega \times q + v \)

- \( A \triangleq \begin{bmatrix} [\omega] & -[\omega]q + v \\ 0 & 0 \end{bmatrix}, \tilde{p}(t) = \begin{bmatrix} p(t) \end{bmatrix} \)

- \( \dot{\tilde{p}}(t) = \begin{bmatrix} \dot{p}(t) \end{bmatrix} = A \begin{bmatrix} p(t) \end{bmatrix} = A\tilde{p}(t) \)

- \( \tilde{p}(t) = e^{At}\tilde{p}(0) \)
Mid Term Project

- Task: Stack box in SAPIEN

- Objective: Pick up the three box and stack them on the cyan plane. The elevation of three box should be: blue > green > red

- You will need to implement each subtask to achieve the final objective. The main structure of the project code is provided, you only need to implement the functions marked as “unimplemented”.
Mid Term Project

• You can work on this project in a team of two people

• Form the team by yourself and inform TA via piazza

• Submit you project as a group with two of your names

• You will also present the paper with your team member later in this quarter

• Deadline: May. 10\textsuperscript{th}, 12:00pm
This project covers the content from lecture 4 to lecture 8. It does not include reinforcement learning and physical simulation.

This project includes two parts. You need to submit the two parts before the deadline:

1. In the first part, you are asked to move your robot hand to pick the box (unphysically, do not consider robot dynamics) based on camera observation. It covers from lecture 5 to lecture 6.

2. In the second part, you are asked to move your hand to pick the box (physically). You need to implement a motion planning algorithm with a PID controller. Then you will need to stack the box to achieve the desired configuration.
First Part of the Project

- You should write the code in “env/hw1_env.py” following the instructions on comments, including the function input and output type.

- Run “hw1.py” to evaluate your code. No need to change this file.