Fundamental Robotics
a comprehensive summary of robotics

Nils Funk

Contents

- 1
- 2
- 3

1 Introduction

2 Fundamental Algebra

2.1 Matrix Classes

2.1.1 Rotation Matrix

$R = \begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

- Orthogonal: Matrix times its transpose equals the identity $RR^T = I$
- Special orthogonal: Determinant is +1
- Closed under multiplication: The product of any two rotation matrices is another rotation matrix
- The inverse of a rotation matrix is also a rotation matrix

2.1.2 Matrix Derivatives

$A(t)$

$\dot{A}(t) = \begin{bmatrix} \frac{dR_{11}(t)}{dt} & \frac{dR_{12}(t)}{dt} \\ \frac{dR_{21}(t)}{dt} & \frac{dR_{22}(t)}{dt} \end{bmatrix}$

$\frac{d}{dt}(A \pm B) = \dot{A} \pm \dot{B}$

$\frac{d}{dt}(AB) = \dot{A}B + A\dot{B}$

$\frac{d}{dt}(A(\theta(t))) = \frac{dA}{d\theta}\dot{\theta}$

2.1.3 Jacobian

2.1.4 Hessian

2.1.5 Eigendecomposition

3 Fundamental Analysis

4 Fundamental Optimization

5 Fundamental Probability

6 Fundamental Dynamic Programming

6.1 Graph Search Algorithms

6.1.1 RRT

6.1.2 RRT*

6.1.3 A*

Reference:
Red Blob Games - A*

Red Blob Games - A* Comparison

6.1.4 Dijkstra

6.1.5 Jump Point Search

In computer science, Jump Point Search (JPS) is an optimization to the A* search algorithm for **uniform-cost grids**. It reduces symmetries in the search procedure by means of graph pruning, eliminating certain nodes in the grid based on assumptions that can be made about the current node's neighbors, as long as certain conditions relating to the grid are satisfied. As a result, the algorithm can consider long "jumps" along straight (horizontal, vertical and diagonal) lines in the grid, rather than the small steps from one grid position to the next that ordinary A* considers.

Jump point search preserves A*'s optimality, while potentially reducing its running time by an order of magnitude.

Reference:
Zero Width - Jump point search explained

Hara Blog - Jump point search

Game Dev - Jump point search fast a pathfinding for uniform-cost grids

Game Development - How to speed up a pathfinding with the jump point search algorithm

7 Fundamental Mechanics

7.1 Coordinate Frames

7.1.1 Inertia Frame of Reference

An inertial frame is a frame with constant acceleration i.e. $\mathbf{a} = 0$. A body with zero net force acting upon it is not accelerating within the inertial frame; that is, such a body is at rest or it is moving at a constant speed in a straight line.

7.2 Sources

7.3 Definitions

Transformation from coordinate frame B to frame A

${}^{A}\mathbf{A}_{B}, {}^{A}\mathbf{R}_{B}$

7.3.1 Rigid Body Transformation

- length is preserved $\|g(p) - g(q)\| = \|p - q\|$
- cross product is preserved $g_*(v) \times g_*(w) = g_*(v \times w)$
- inner product is preserved $g_*(v) \cdot g_*(w) = v \cdot w$

7.4 Rotation Group SO(3)

$SO(3) = \{\mathbf{R} \in \mathbb{R} | \mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}, det\mathbf{R} = 1\}$

7.4.1 Rotation Matrices

7.4.2 Euler Angles

7.4.3 Axis Angle Parametrization

Rodrigues' formula

7.4.4 Quaternions

7.5 Time Derivatives of Rotations

$R^T(t)R(t) = I \frac{d}{dt}(.) \dot{R}^T R + R^T \dot{R} = 0$

$R(t)R^T(t) = I \frac{d}{dt}(.) R\dot{R}^T + \dot{R}R^T = 0$

$R^T\dot{R}$ and $\dot{R}R^T$ are skew symmetric

7.6 Skew Symmetric

7.6.1 Transformation vs Displacement

7.7 Momentum

7.7.1 Linear Momentum

7.7.2 Angular Momentum

${}^A\mathbf{H}_C^B = \mathbf{I}_C \cdot^A \omega^B$

Angular momentum $\mathbf{H}$ with inertia tensor $\mathbf{I}$ with C as the origin and angular velocity ω of the body B in coordinate frame A.

7.7.3 Rate of Change of Angular Momentum

The rate of change of angular momentum of the rigid body B relative to C in A is equal to the resultant moment of all external forces $\mathbf{M}$ acting on the body B relative to C.

$\frac{{}^A d^A \mathbf{H}_C^B}{dt} = \mathbf{M}_C^B$

Simplification

$\frac{{}^B d^A \mathbf{H}_C^B}{dt} +^A \omega^B \times \mathbf{H}_C = \mathbf{M}_C^B$

$$\begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix} \begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix} + \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} M_{C,1} \\ M_{C,2} \\ M_{C,3} \end{bmatrix}$$

The second term on the left hand side, ${}^{A}\omega^{B} \times \mathbf{H}_C$, is zero when angular velocity ${}^{A}\omega^{B}$ is perpendicular to $\mathbf{H}_C$. This is the case if the body rotates around a **principal axis of inertia**.

7.8 Moment of Inertia

The moment of inertia $\mathbf{I}$ is a tensor that determines the torque needed for a desired angular acceleration about a rotational axis; similar to how mass determines the force needed for a desired acceleration. It depends on the body's mass distribution and the axis chosen, with larger moments requiring more torque to change the body's rotation. It is an **additive** property: for a point mass the moment of inertia is just the mass m times the square of the distance r to the rotation axis.

$$\mathbf{I} = mr^2$$

For bodies free to rotate in 3D, their moments can be described bz a szmmetric 3x3 matrix, with a set of mutually perpendicular **principal axes** for which this matrix is diagonal and torques around the axes act **independently** of each other.

7.8.1 Principal Axis of Inertia

Measured in the body frame the inertia matrix is a constant real symmetric matrix. A real symmetric matrix has the eigendecomposition into the product of a rotation matrix $\mathbf{Q}$ and a diagonal matrix $\mathbf{\Lambda}$.

$$\mathbf{I}^B_{\mathbf{C}} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathsf{T}$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{bmatrix}$$

The columns of the rotation matrix $\mathbf{Q}$ define the directions of the principal axes of the body, and the constant I_1, I_2 and I_3 are called the **principal moments of inertia**.

8 Fundamental Control

9 Fundamental Machine Learning

10 Fundamental Computer Vision