法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

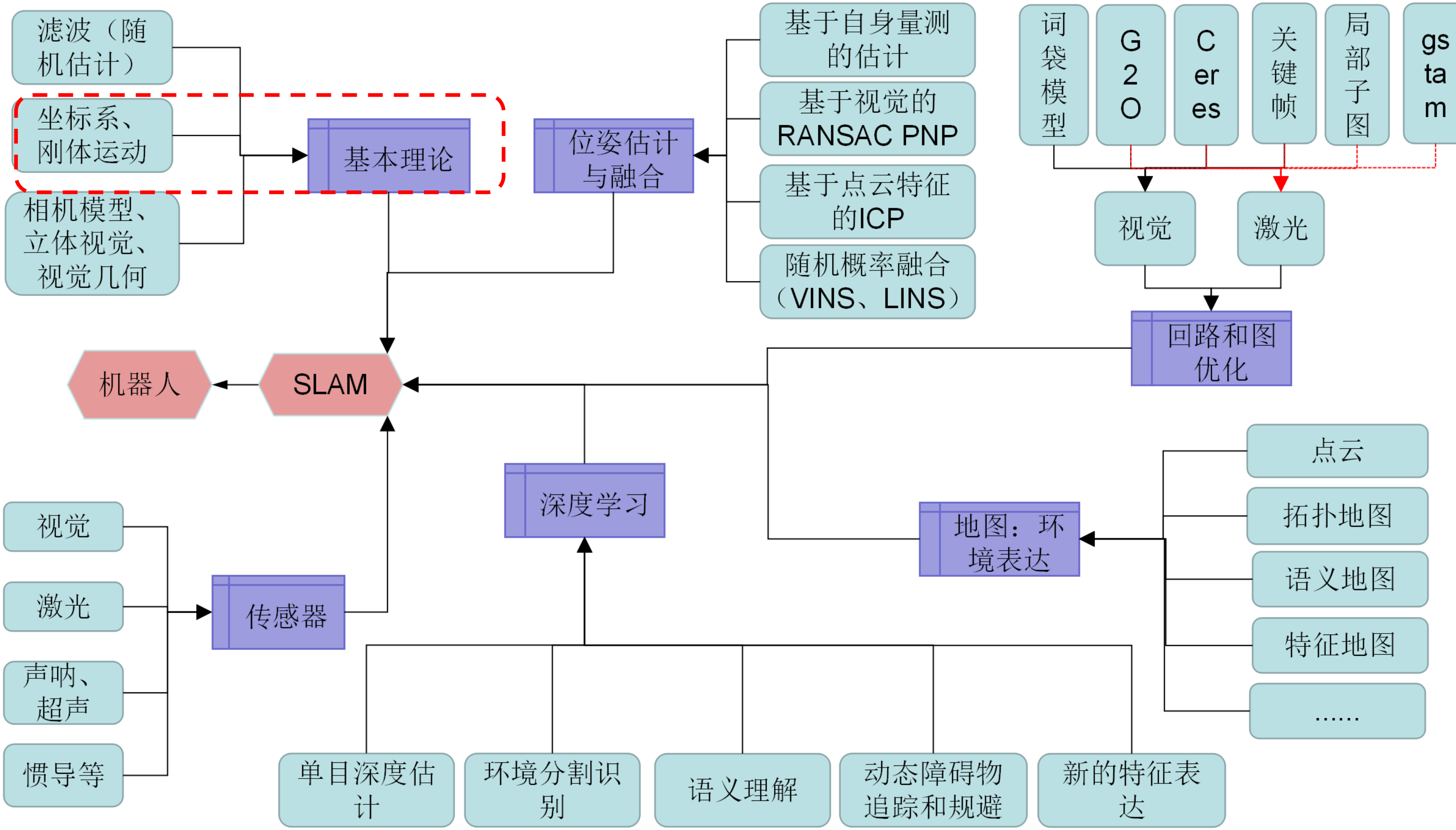
- 课程详情请咨询
  - 微信公众号：小象学院
  - 新浪微博：小象AI学院

第二讲：

SLAM基本理论一：坐标系、刚体运动和李群

主讲：杨亮

总结

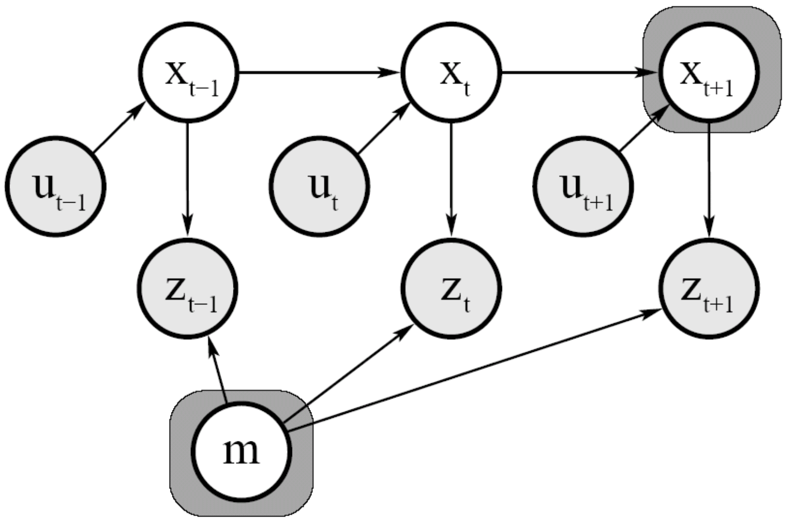


滤波（随机估计）
坐标系、刚体运动
相机模型、立体视觉、视觉几何
基本理论
位姿估计与融合
基于自身量测的估计
基于视觉的RANSAC PNP
基于点云特征的ICP
随机概率融合（VINS、LINS）
词袋模型
G2O
Ceres
关键帧
局部子图
gstam
视觉
激光
回路和图优化
机器人
SLAM
深度学习
地图：环境表达
点云
拓扑地图
语义地图
特征地图
......
视觉
激光
声呐、超声
惯导等
传感器
单目深度估计
环境分割识别
语义理解
动态障碍物追踪和规避
新的特征表达

提纲

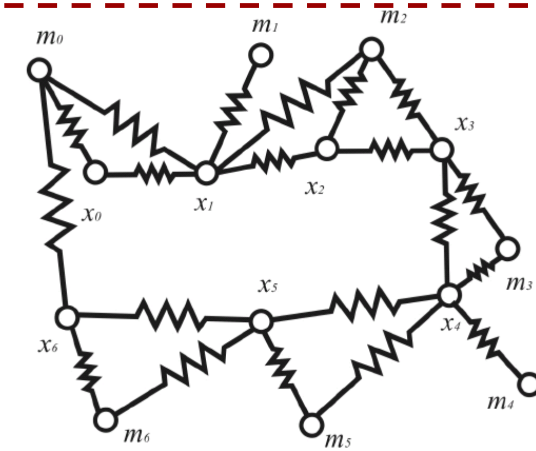
- SLAM的数学表达
- 欧式坐标系和刚体姿态表示
- 李群和李代数
- 实例：Eigen和Sophus在滤波器上的应用

SLAM的数学表达



$$p(x_t, m \mid z_{1:t}, u_{1:t}) =$$
$$\int\int \cdots \int p(x_{1:t}, m \mid z_{1:t}, u_{1:t})\, dx_1\, dx_2 \ldots dx_{t-1}$$

+



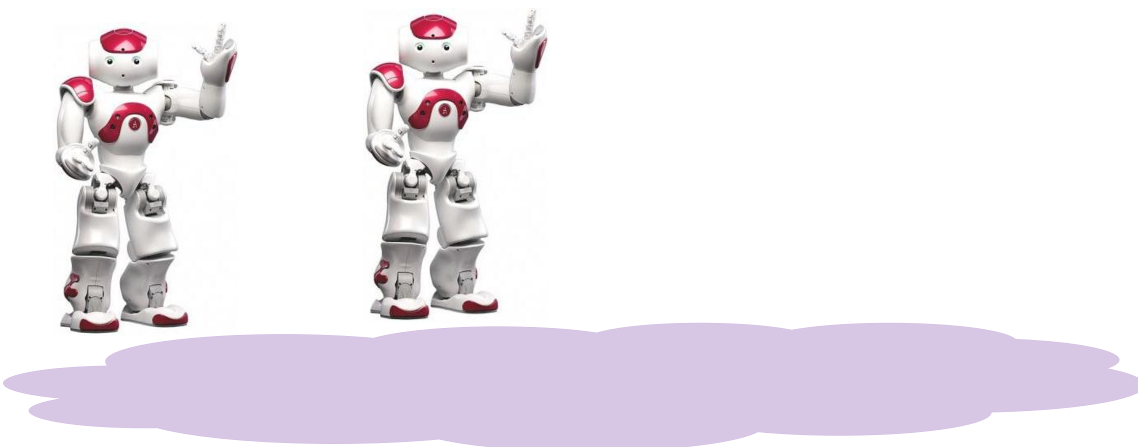
$$\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$
$$\mathbf{F}(\mathbf{x}) = \sum_{\langle i,j \rangle \in \mathcal{C}} \underbrace{\mathbf{e}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}}_{\mathbf{F}_{ij}}$$
$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}}\, \mathbf{F}(\mathbf{x})$$

滤波器：无限增加的状态

姿态图：有限的图链接

SLAM的数学表达

聊聊滤波器：



起点　　下一步没有观测

状态 x_k　　状态 x_{k+1}

传感器 u_k　　传感器 u_{k+1}

单步估计？

Given

- The robot's controls
  $u_{1:T} = \{u_1, u_2, u_3 \ldots, u_T\}$
- Observations
  $z_{1:T} = \{z_1, z_2, z_3 \ldots, z_T\}$

Wanted

- Map of the environment
  m
- Path of the robot
  $x_{0:T} = \{x_0, x_1, x_2 \ldots, x_T\}$

SLAM的数学表达

Evolving State Of The EKF:

$$X_{IMU}=[{}^{W}_{B}P \quad {}^{W}_{B}V \quad {}^{W}_{B}a \quad {}^{B}_{W}q \quad w_B \quad b_a \quad b_g]$$

Simplified

$$X_{IMU}=[{}^{W}_{B}P \quad {}^{W}_{B}V \quad b_a \quad {}^{B}_{W}q \quad b_g]$$

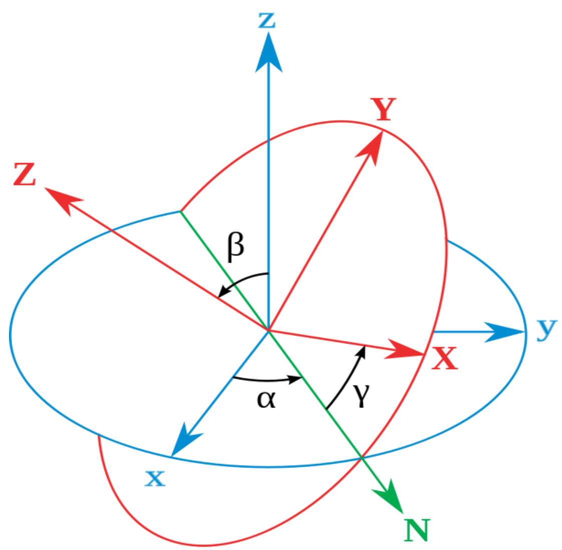
位姿：位置、姿态

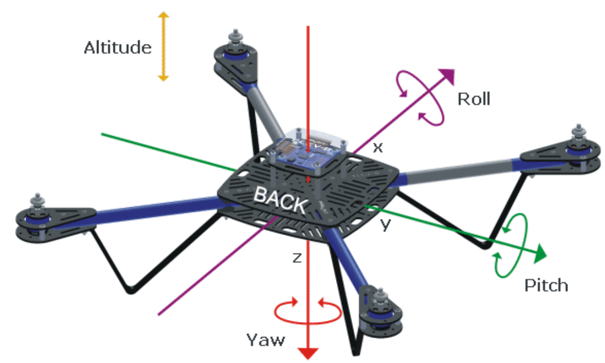
$$P=(x,y,z)$$

$$R=(\alpha,\beta,\gamma)$$

四元数

$$\mathrm{q}=a+bi+cj+dk$$





SLAM的数学表达

Evolving State Of The EKF:

$$X_{IMU} = [{}^{W}_{B}P \quad {}^{W}_{B}V \quad {}^{W}_{B}a \quad {}^{B}_{W}q \quad w_B \quad b_a \quad b_g]$$

Simplified

$$X_{IMU} = [{}^{W}_{B}P \quad {}^{W}_{B}V \quad b_a \quad {}^{B}_{W}q \quad b_g]$$

$$\begin{cases} \dot{P} = v \\ \dot{V} = R(a_m - b_a) + g \\ \dot{a} = w \times a \\ \dot{q} = \frac{1}{2} q \otimes (w_m - b_g) \\ \dot{w} = 0 \\ \dot{b}_a = 0 \\ \dot{b}_g = 0 \end{cases}$$

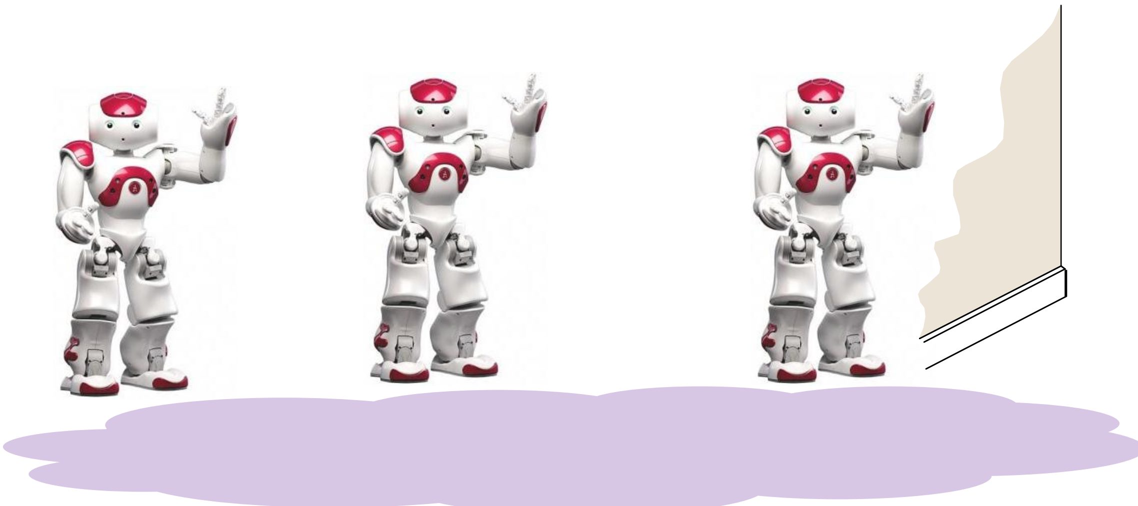
State Transition Matrix

F

$$\begin{cases} P_{k+1} = P_k + \Delta T \cdot v_k + \frac{1}{2} \cdot \Delta T^2 \cdot a_k \\ V_{k+1} = V_k + \Delta T \cdot a_k \\ a_{k+1} = a_k + \Delta T \cdot (w_k \cdot a_k) \\ q_{K+1} = \delta q(\theta) \otimes q_k \\ w_{k+1} = w_k + \Delta T \cdot \alpha_k \\ b_{a\,k+1} = b_{a\,k} \\ b_{g\,k+1} = b_{g\,k} \end{cases}$$

SLAM的数学表达

聊聊滤波器：



起点　　下一步没有观测　　视觉有反馈

Given

- The robot’s controls

$$u_{1:T} = \{u_1, u_2, u_3 \ldots, u_T\}$$

- Observations

$$z_{1:T} = \{z_1, z_2, z_3 \ldots, z_T\}$$

Wanted

- Map of the environment

$$m$$

- Path of the robot

$$x_{0:T} = \{x_0, x_1, x_2 \ldots, x_T\}$$

状态 x_k

传感器 u_k

自身估计状态 $\bar{x}_{k+1}$

传感器 u_{k+1}

基于地图观测 $Z_{k+1} = f(\bar{x}_{k+1})$

SLAM的数学表达

估计：

$$X_{\mathrm{k}} = F \cdot X_{k-1} + G \cdot w$$

$$\overline{bel}(x_{\mathrm{k}}) = p(x_k \mid u_k, x_{k\text{-}1}) bel(x_{k\text{-}1})$$

观测：

$$Z_k = H \cdot X_k$$

$$P(z_k \mid x_k)$$

增益：

$$K_{k+1} = P_{k+1|k} H_{k+1}^T (H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1})^{-1}$$

更新状态:

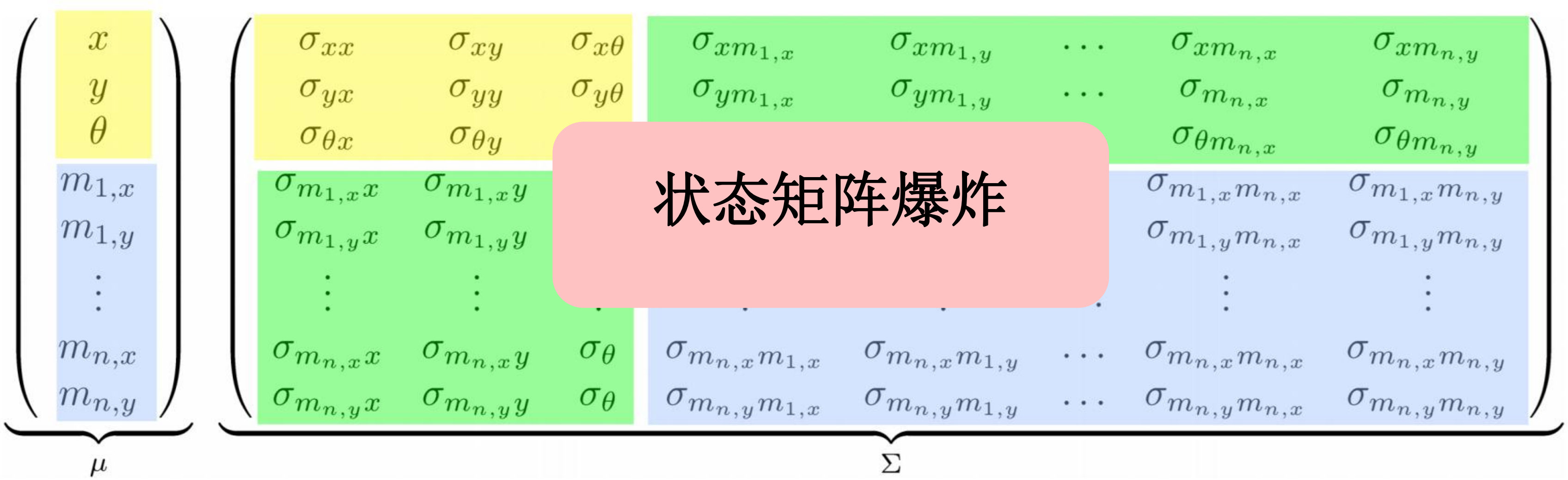
$$X_{K|real} = X_k + K_k (Z_K - H_k X_k)$$

SLAM的数学表达

机器人的状态：

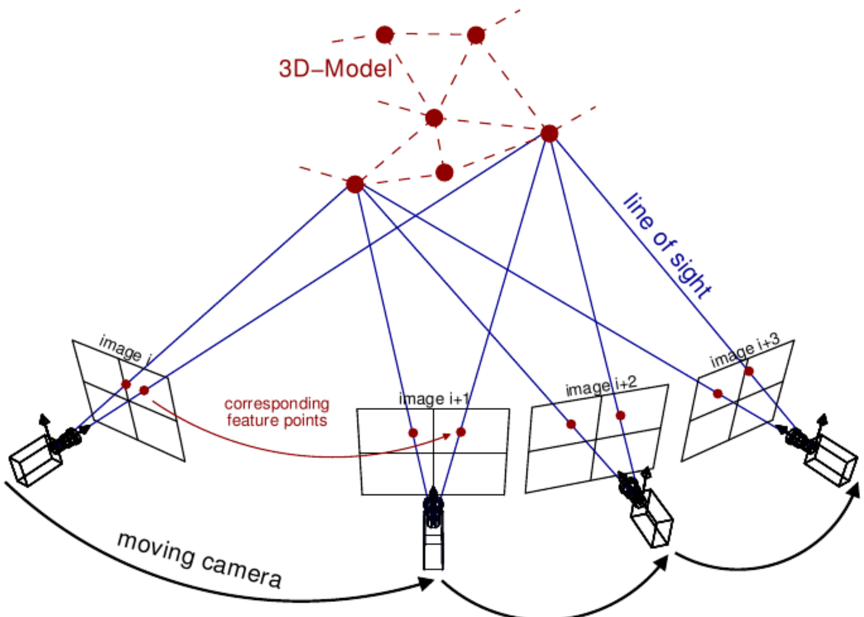
$$x_t = (\underbrace{x, y, \theta}_{\text{robot's pose}}, \underbrace{m_{1,x}, m_{1,y}}_{\text{landmark 1}}, \ldots, \underbrace{m_{n,x}, m_{n,y}}_{\text{landmark n}})^T$$

状态的方差矩阵（置信）：



SLAM的数学表达

聊聊图：



$$[R,t]=$$

$$[\alpha,\beta,\gamma,x,y,z]$$

初始化：

状态 x_k

传感器 u_k

帧到帧估计：

状态 $x_{k+1}=[R,t]x_k$

局部捆绑约束（Bundle Adjustment）：

$$\min\sum_{e_{ij}}\| x_k - T_{eik} x_i \|^2$$

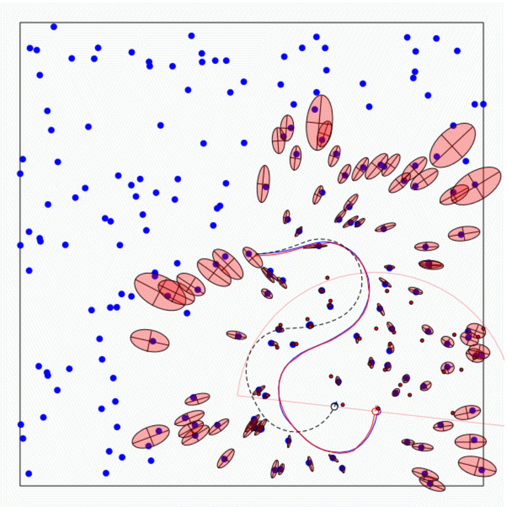
$$x_{k|real}=x_k+\delta x$$

SLAM的数学表达

滤波器类：

$$P(x \mid u) = \int P(x \mid u, x')P(x')dx'$$

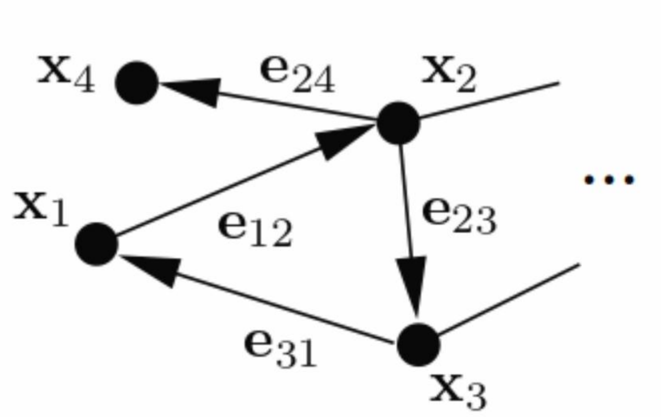
$$Bel(x_t) = \eta\ P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1})\ Bel(x_{t-1})\ dx_{t-1}$$



回归优化类：

$$\min \sum_{e_{ij}} \| x_k - T_{eik} x_i \|^2$$

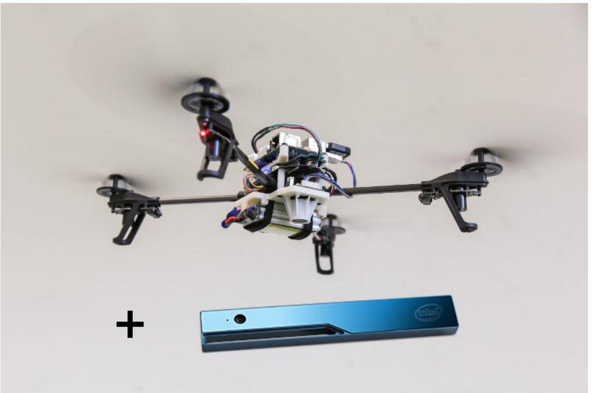
$$x_{k|real} = x_k + \delta x$$



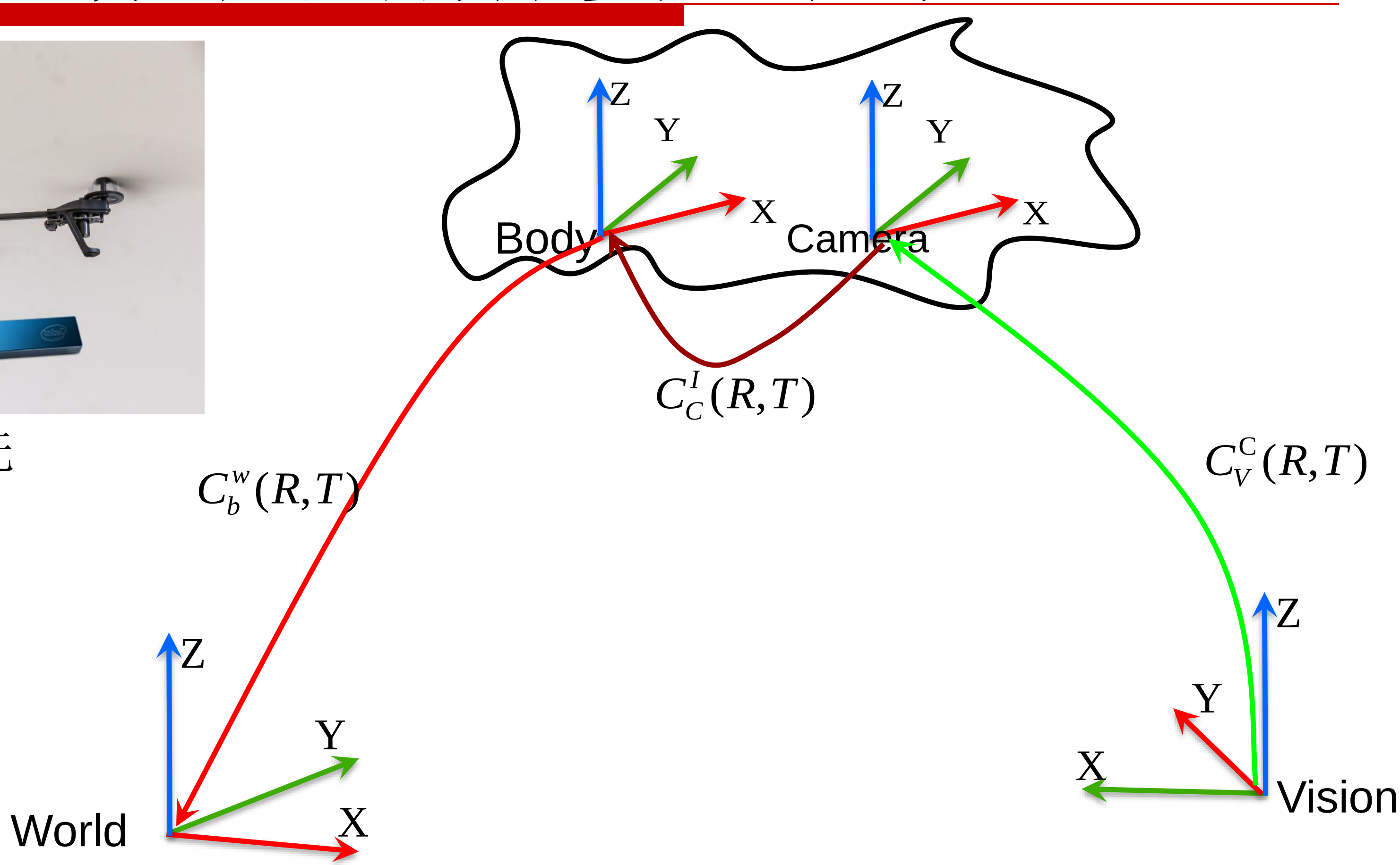
提纲

- SLAM的数学表达
- 欧式坐标系和刚体姿态表示
- 李群和李代数
- 实例：Eigen和Sophus在滤波器上的应用

欧式坐标系和刚体姿态表示



视觉SLAM无人机导航



$C_b^w(R,T)$：本体到世界

$C_C^I(R,T)$ ：相机到本体

$C_V^C(R,T)$ ：视野到相机本体（投影）

欧式坐标系和刚体姿态表示

Evolving State Of IMU:

$$X_{IMU} = [\,{}^{W}_{B}P \quad {}^{W}_{B}V \quad {}^{W}_{B}a \quad {}^{B}_{W}q \quad w_B \quad b_a \quad b_g\,]$$

位姿：位置、姿态

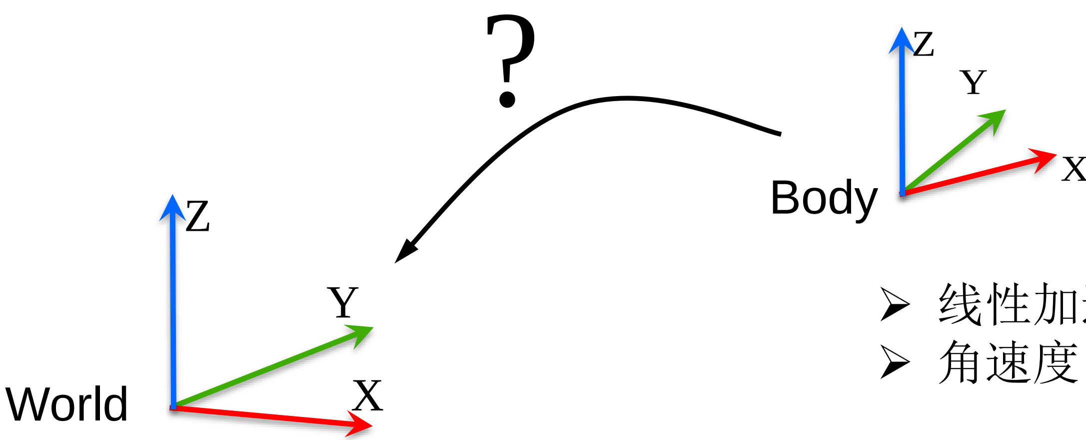
$$P = (x, y, z)$$

$$R = (\alpha, \beta, \gamma)$$

$$\Longleftrightarrow$$

四元数

$$q = a + bi + cj + dk$$

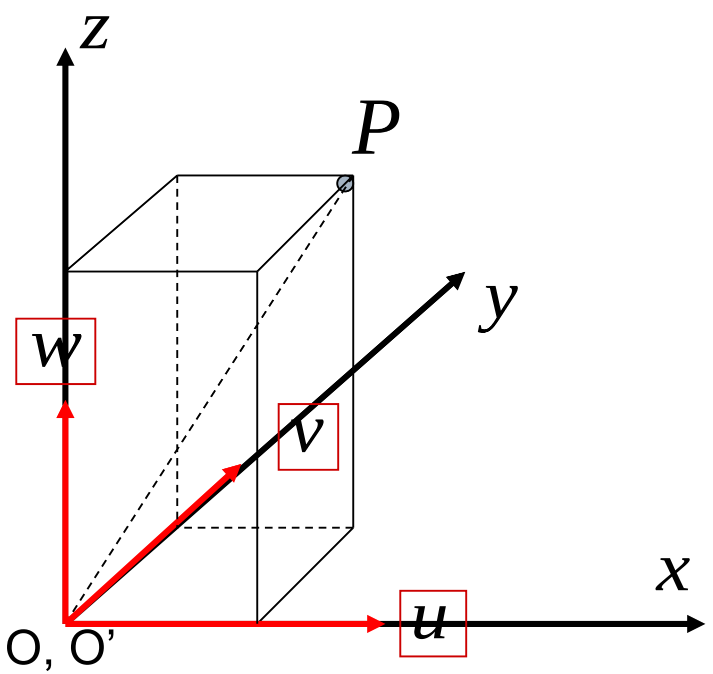


- 线性加速度
- 角速度

欧式坐标系和刚体姿态表示

我们来看看最基本的坐标系两个坐标系：

- OXYZ
- Ouvw



1）在OXYZ坐标系下：

$$P_{xyz} = [p_x, p_y, p_z]^T$$

$$P_{xyz} = p_x \mathrm{i}_\mathrm{x} + p_y \mathrm{j}_\mathrm{y} + p_z \mathrm{k}_\mathrm{z}$$

2）在Ouvw坐标系下：

$$P_{uvw} = p_u \mathrm{i}_\mathrm{u} + p_v \mathrm{j}_\mathrm{v} + p_w \mathrm{k}_\mathrm{w}$$

现在OXYZ和Ouvw重合：

$$P_{xyz} = p_x \mathrm{i}_\mathrm{x} + p_y \mathrm{j}_\mathrm{y} + p_z \mathrm{k}_\mathrm{z} = P_{uvw} = p_u \mathrm{i}_\mathrm{u} + p_v \mathrm{j}_\mathrm{v} + p_w \mathrm{k}_\mathrm{w}$$

基本运算：点乘

x 和 y 是在 R^3 中的任意两个向量，并且θ 是两个向量的夹角：

$$x \cdot y = |x||y| \cos\theta$$

垂直的坐标系的性质

- 相互垂直

$$i \cdot j = 0$$

$$i \cdot k = 0$$

$$k \cdot j = 0$$

- 单位向量

$$|i| = 1$$

$$|j| = 1$$

$$|k| = 1$$

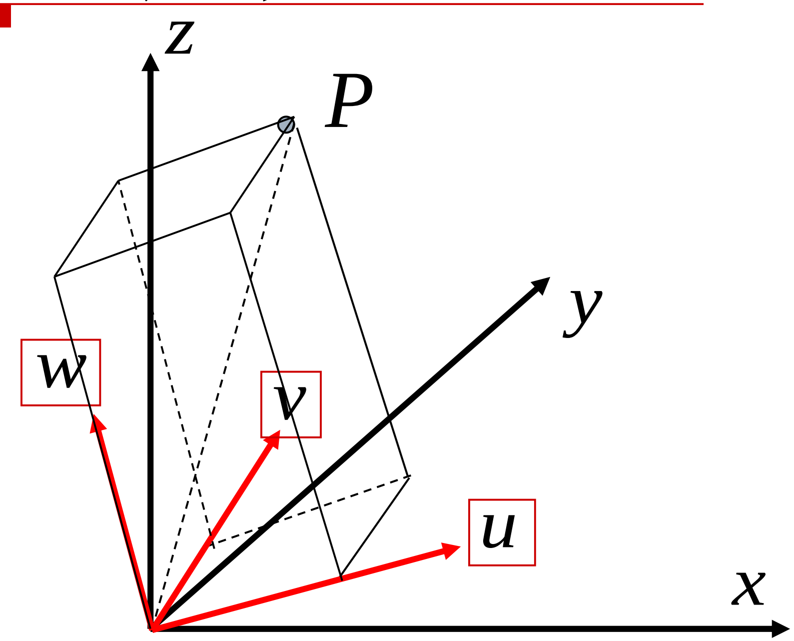
欧式坐标系和刚体姿态表示

假设发生旋转：

$$P_{xyz} = p_x \mathrm{i}_\mathrm{x} + p_y \mathrm{j}_\mathrm{y} + p_z \mathrm{k}_\mathrm{z}$$

$$P_{uvw} = p_u \mathrm{i}_\mathrm{u} + p_v \mathrm{j}_\mathrm{v} + p_w \mathrm{k}_\mathrm{w}$$

$$P_{xyz} = RP_{uvw}$$



如果知道了关系：

$p_x \leftrightarrow p_u$

$p_x \leftrightarrow p_u$

$p_y \leftrightarrow p_v$

$p_z \leftrightarrow p_w$

$$p_x = \mathrm{i}_\mathrm{x} \cdot P = \mathrm{i}_\mathrm{x} \cdot \mathrm{i}_\mathrm{u} p_u + \mathrm{i}_\mathrm{x} \cdot \mathrm{j}_\mathrm{v} p_v + \mathrm{i}_\mathrm{x} \cdot \mathrm{k}_\mathrm{w} p_w$$

$$p_y = \mathrm{j}_\mathrm{y} \cdot P = \mathrm{j}_\mathrm{y} \cdot \mathrm{i}_\mathrm{u} p_u + \mathrm{j}_\mathrm{y} \cdot \mathrm{j}_\mathrm{v} p_v + \mathrm{j}_\mathrm{y} \cdot \mathrm{k}_\mathrm{w} p_w$$

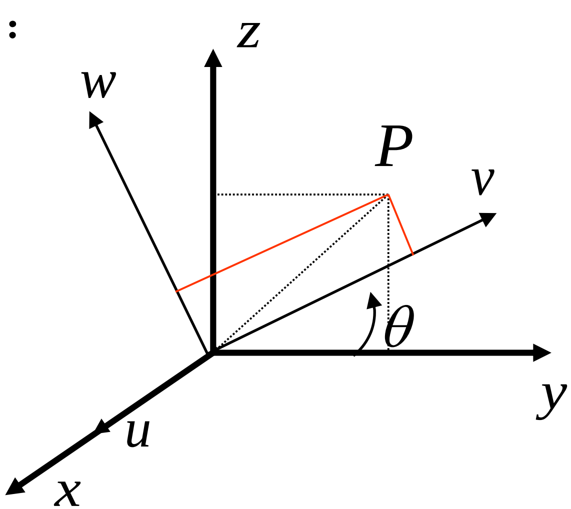
$$p_z = \mathrm{k}_\mathrm{z} \cdot P = \mathrm{k}_\mathrm{z} \cdot \mathrm{i}_\mathrm{u} p_u + \mathrm{k}_\mathrm{z} \cdot \mathrm{j}_\mathrm{v} p_v + \mathrm{k}_\mathrm{z} \cdot \mathrm{k}_\mathrm{w} p_w$$

欧式坐标系和刚体姿态表示

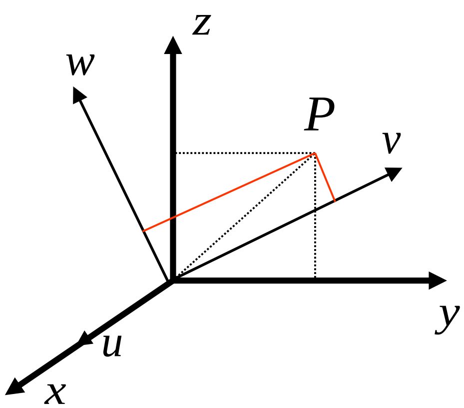
基本的3维旋转表示：

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} \mathrm{i_x} \cdot \mathrm{i_u} & \mathrm{i_x} \cdot \mathrm{j_v} & \mathrm{i_x} \cdot \mathrm{k_w} \\ \mathrm{j_y} \cdot \mathrm{i_u} & \mathrm{j_y} \cdot \mathrm{j_v} & \mathrm{j_y} \cdot \mathrm{k_w} \\ \mathrm{k_z} \cdot \mathrm{i_u} & \mathrm{k_z} \cdot \mathrm{j_v} & \mathrm{k_z} \cdot \mathrm{k_w} \end{bmatrix} \begin{bmatrix} p_u \\ p_v \\ p_w \end{bmatrix}$$

围绕X轴旋转 θ ：



欧式坐标系和刚体姿态表示



$$p_x = p_u$$
$$p_y = p_v \cos\theta - p_w \sin\theta$$
$$p_z = p_v \sin\theta + p_w \cos\theta$$

$$\Leftrightarrow \quad Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\theta & -S\theta \\ 0 & S\theta & C\theta \end{bmatrix}$$

$$Rot(x,\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\theta & -S\theta \\ 0 & S\theta & C\theta \end{bmatrix}$$

$$Rot(z,\theta) = \begin{bmatrix} C\theta & -S\theta & 0 \\ S\theta & C\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Rot(y,\theta) = \begin{bmatrix} C\theta & 0 & S\theta \\ 0 & 1 & 0 \\ -S\theta & 0 & C\theta \end{bmatrix}$$

三轴

$$R = Rot(y,\phi) I_3 Rot(w,\theta) Rot(u,\alpha)$$

$$= \begin{bmatrix} C\phi & 0 & S\phi \\ 0 & 1 & 0 \\ -S\phi & 0 & C\phi \end{bmatrix} \begin{bmatrix} C\theta & -S\theta & 0 \\ S\theta & C\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\alpha & -S\alpha \\ 0 & S\alpha & C\alpha \end{bmatrix}$$

$$= \begin{bmatrix} C\phi C\theta & S\phi S\alpha - C\phi S\theta C\alpha & C\phi S\theta S\alpha + S\phi C\alpha \\ S\theta & C\theta C\alpha & -C\theta S\alpha \\ -S\phi C\theta & S\phi S\theta C\alpha + C\phi S\alpha & C\phi C\alpha - S\phi S\theta S\alpha \end{bmatrix}$$

欧式坐标系和刚体姿态表示

别忘了，平移！！

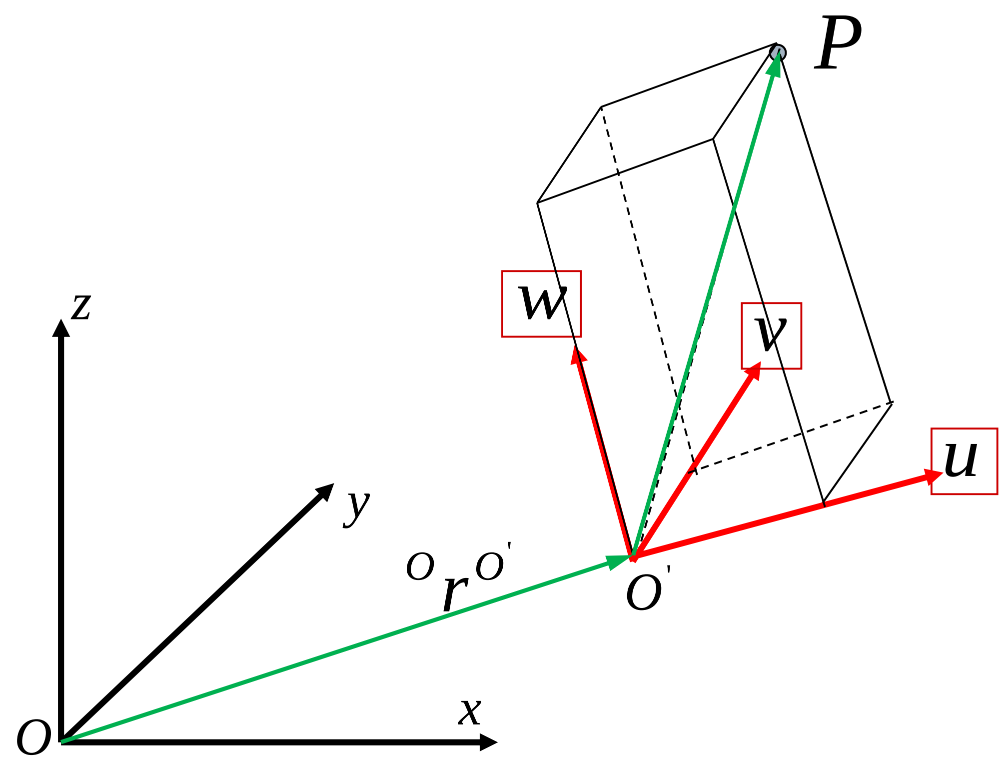
$$^{O}r^{P} = {}^{O}R_{O'}\,{}^{'}r^{P} + {}^{O}r^{o'}$$

$^{O}r^{P}$ ：P在世界坐标系下的位置

$^{O}R_{O'}$ ：本体坐标系在世界坐标系下的旋转

$^{O}r^{o'}$ ：本体坐标系在世界坐标系下的位置

$^{'}r^{P}$ ：P在本体坐标系下的位置

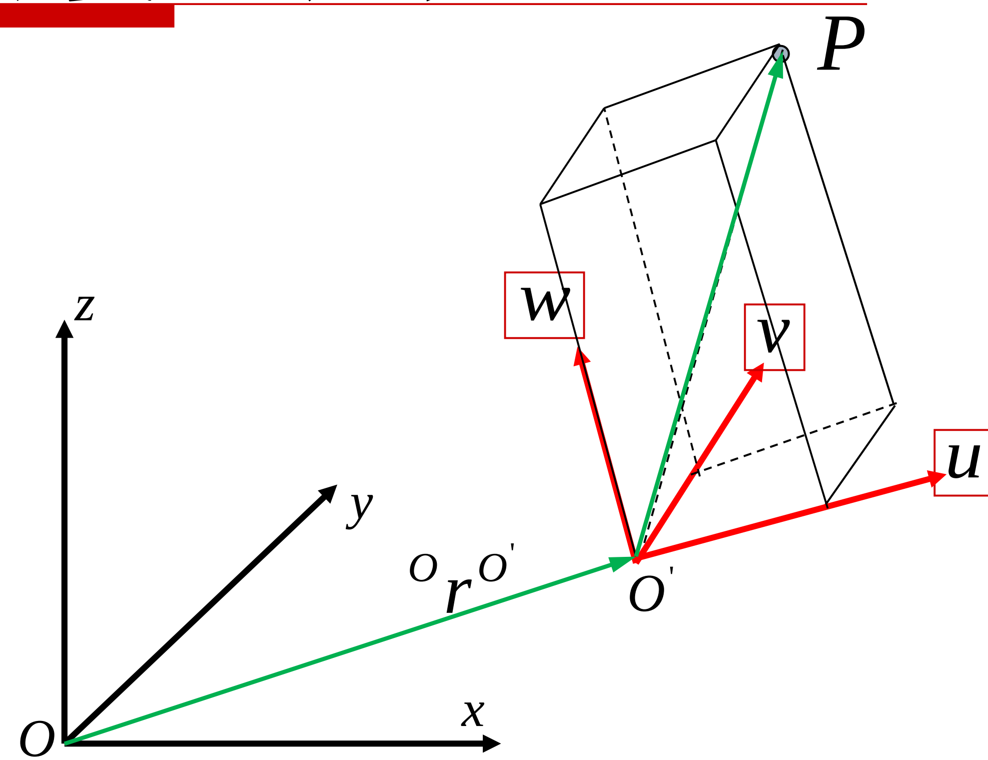


欧式坐标系和刚体姿态表示

欧式空间刚体的旋转、平移表达：

$$ {}^{O}r^{P} = {}^{O}R_{O'}\,{}^{O'}r^{P} + {}^{O}r^{O'} $$

$$ \begin{bmatrix} {}^{O}r^{P} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{O}R_{O'} & {}^{O}r^{O'} \\ 0_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} {}^{O'}r^{P} \\ 1 \end{bmatrix} $$



齐次转换矩阵表达：

$$ {}^{O}T_{O'} = \begin{bmatrix} {}^{O}R_{O'} & {}^{O}r^{O'} \\ 0_{1\times3} & 1 \end{bmatrix} = \begin{bmatrix} \boxed{R_{3\times3}} & \boxed{P_{3\times1}} \\ 0 & \boxed{1} \end{bmatrix} $$

旋转矩阵

位置向量

尺度

欧式坐标系和刚体姿态表示

$$^{O}T_{O'} = \begin{bmatrix} ^{O}R_{O'} & ^{O}r^{o'} \\ 0_{1\times3} & 1 \end{bmatrix} = \begin{bmatrix} \boxed{R_{3\times3}} & \boxed{P_{3\times1}} \\ 0 & \boxed{1} \end{bmatrix}$$

特殊情况：

1）只有旋转

$$^{O}T_{O'} = \begin{bmatrix} R_{3\times3} & O_{3\times1} \\ 0 & 1 \end{bmatrix}$$

2）只有平移

$$^{O}T_{O'} = \begin{bmatrix} I_{3\times3} & P_{3\times1} \\ 0 & 1 \end{bmatrix}$$

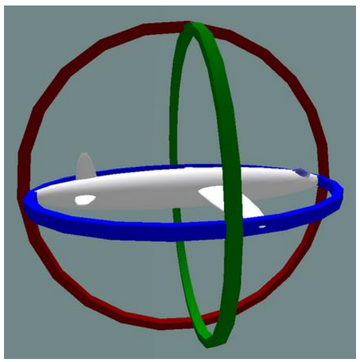
旋转平移构成了一个特殊的欧式群：

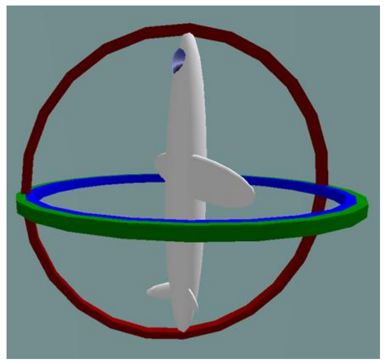
$$\{T \mid T = \begin{bmatrix} R & r \\ 0_{1\times3} & 1 \end{bmatrix}, R \in R^{3\times3}, r \in R^{3}, R^{T}R = RR^{T} = I, |R| = 1\} = \mathrm{SE}(3)$$

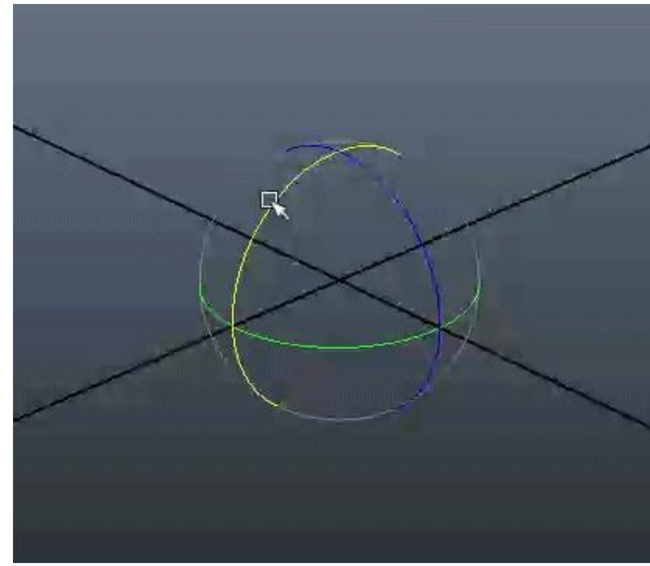
欧式坐标系和刚体姿态表示

Gimbal Lock:

由于两个坐标轴旋转到重合，导致3个自由度退化成2个自由度。

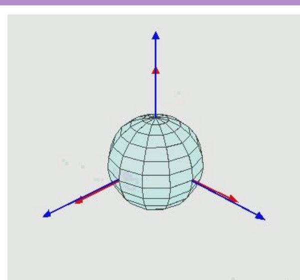






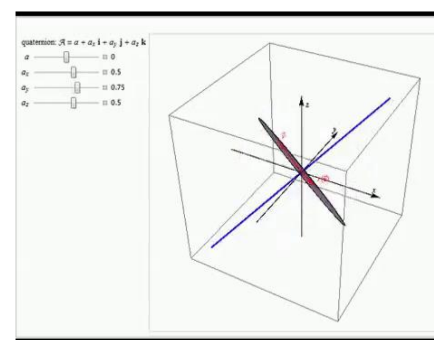
提出了四元数：Quaternion

欧式旋转



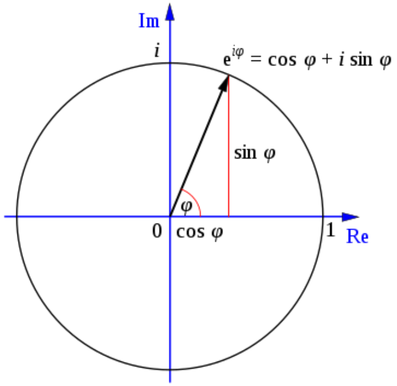
$$q = a + bi + cj + dk$$

$$i^2 = j^2 = k^2 = ijk = -1$$

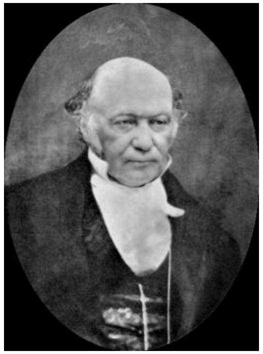


欧式坐标系和刚体姿态表示

欧拉旋转表达式：

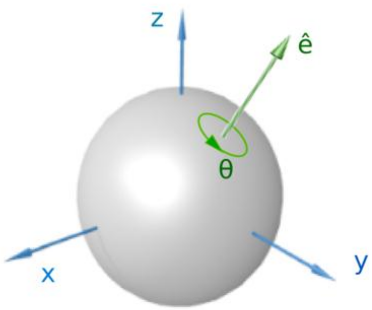


$$e^{i\varphi} = \cos\varphi + i\sin\varphi$$



1843： William Rowan Hamilton

这个表示方法好用



单位向量为：

$$\hat{e} = e_x i + e_y j + e_z k$$

旋转表示：

$$q = e^{\frac{\theta}{2}(e_x i + e_y j + e_z k)} = \cos\frac{\theta}{2} + (e_x i + e_y j + e_z k)\sin\frac{\theta}{2}$$

$$= \cos\frac{\theta}{2} + \hat{e}\sin\frac{\theta}{2}$$

一个向量绕单位向量旋转 θ

欧式坐标系和刚体姿态表示

Evolving State Of IMU:

$$X_{IMU} = [{}^W_B P \quad {}^W_B V \quad {}^W_B a \quad {}^B_W q \quad w_B \quad b_a \quad b_g]$$

输出

角速度 $[w_x, w_y, w_z]$

加速度 $[a_x, a_y, a_z]$

我们表示方向：

四元数 $\mathrm{q_k} = a_k + b_k i + c_k j + d_k k$

$$\dot{q} = \frac{1}{2}\begin{bmatrix} 0 & -w_x & -w_y & -w_z \\ w_x & 0 & w_z & -w_y \\ w_y & -w_z & 0 & w_x \\ w_z & w_y & -w_x & 0 \end{bmatrix} \otimes q$$

$$q_{k+1} = \exp(w\frac{\Delta t}{2}) \otimes q_k$$

欧式坐标系和刚体姿态表示

四元数旋转：$p' = Rp$

$$\mathbf{R} = \begin{bmatrix} 1-2s(q_j^2+q_k^2) & 2s(q_iq_j-q_kq_r) & 2s(q_iq_k+q_jq_r) \\ 2s(q_iq_j+q_kq_r) & 1-2s(q_i^2+q_k^2) & 2s(q_jq_k-q_iq_r) \\ 2s(q_iq_k-q_jq_r) & 2s(q_jq_k+q_iq_r) & 1-2s(q_i^2+q_j^2) \end{bmatrix}$$

欧拉旋转：

$$R = Rot(y,\phi)I_3Rot(w,\theta)Rot(u,\alpha)$$

$$= \begin{bmatrix} C\phi C\theta & S\phi S\alpha - C\phi S\theta C\alpha & C\phi S\theta S\alpha + S\phi C\alpha \\ S\theta & C\theta C\alpha & -C\theta S\alpha \\ -S\phi C\theta & S\phi S\theta C\alpha + C\phi S\alpha & C\phi C\alpha - S\phi S\theta S\alpha \end{bmatrix}$$

变换

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \mathrm{atan2}(2(q_0q_1+q_2q_3), 1-2(q_1^2+q_2^2)) \\ \mathrm{asin}(2(q_0q_2-q_3q_1)) \\ \mathrm{atan2}(2(q_0q_3+q_1q_2), 1-2(q_2^2+q_3^2)) \end{bmatrix}$$

提纲

- SLAM的数学表达
- 欧式坐标系和刚体姿态表示
- 李群和李代数
- 实例：Eigen和Sophus在滤波器上的应用

李群和李代数

旋转平移构成了一个特殊的欧式群：

$$\{T \mid T = \begin{bmatrix} R & r \\ 0_{1\times3} & 1 \end{bmatrix}, R \in R^{3\times3}, r \in R^3, R^T R = RR^T = I, |R| = 1\} = \mathrm{SE}(3)$$

旋转构成了一个特殊的正交群SO3：

$$\{R \mid R \in R^{3\times3}, R^T R = RR^T = I, |R| = 1\} = \mathrm{SO}(3)$$

我们只讨论旋转和平移这两个群

李群

记李群为 G， 满足如下性质：

- Closure　　　：$如果 A,B\in G, 那样 A\otimes B\in G$
- Associativity：$如果 A,B,C\in G, 那样 (A\otimes B)\otimes C = A\otimes(B\otimes C)$
- Identity　　　：$存在 1_G\in G, 使得 1_G\otimes A = A\otimes 1_G = A$
- Inverse　　　：$存在 B\in G, 使得 B\otimes A = A\otimes B = 1_G$

回到四元数和欧拉角对增量的定义

$$\Delta q = \begin{bmatrix} 1 \\ \\ \frac{1}{2}\Delta\theta \end{bmatrix} \Rightarrow \dot{q} = \frac{1}{2} q \otimes \begin{bmatrix} 0 & -w_x & -w_y & -w_z \\ w_x & 0 & w_z & -w_y \\ w_y & -w_z & 0 & w_x \\ w_z & w_y & -w_x & 0 \end{bmatrix}$$

$$\dot{R} = R \otimes \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & w_x \\ w_y & w_x & 0 \end{bmatrix}$$

李群和李代数

$$\dot{R} = R \otimes \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & w_x \\ w_y & w_x & 0 \end{bmatrix} \xrightarrow{\text{令}} [w]_\times = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & w_x \\ w_y & w_x & 0 \end{bmatrix}$$

$$\dot{R} = R[w]_\times \longrightarrow R = \exp([w]_\times t)R$$

$$R(w) = \exp([w]_\times)$$

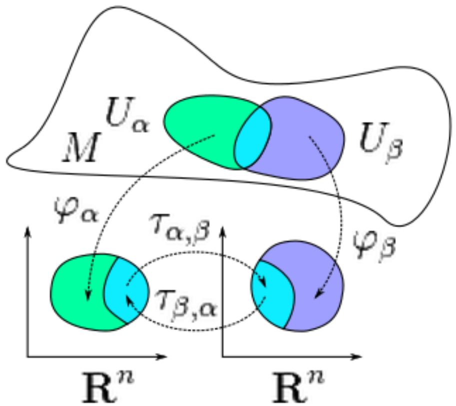
$$= \mathrm{I} + [w]_\times + \frac{1}{2!}[w]_\times^2 + \frac{1}{3!}[w]_\times^3 + \ldots.$$

通过这种方式，找寻增量累计的关系

在滤波器、单步估计上都需要

李群和李代数

跳开所有的步骤，为什么要研究这个：



我们期待的以及所假设的是平滑连续运动

研究运动旋转、平移

$$\{T \mid T = \begin{bmatrix} R & r \\ 0_{1\times3} & 1 \end{bmatrix}, R \in R^{3\times3}, r \in R^3, R^T R = RR^T = I, |R| = 1\} = \mathrm{SE}(3)$$

- 李群（Lie Group）：

- 具有连续（光滑）性质的群。
- 既是群也是流形。
- 直观上看，一个刚体能够连续地在空间中运动，故 SO(3) 和 SE(3) 都是李群。

提纲

- SLAM的数学表达
- 欧式坐标系和刚体姿态表示
- 李群和李代数
- 实例：Eigen和Sophus在滤波器上的应用

Eigen和Sophus在滤波器上的应用

http://eigen.tuxfamily.org/index.php?title=Main_Page

Installation:

sudo apt-get install libeigen3-dev

手动：

git clone https://bitbucket.org/eigen/eigen/
mkdir build
cd build
cmake ..
make
sudo make install

Eigen和Sophus在滤波器上的应用

Module	Header file	Contents
Core	`#include <Eigen/Core>`	Matrix and Array classes, basic linear algebra (including triangular and selfadjoint products), array manipulation
Geometry	`#include <Eigen/Geometry>`	Transform, Translation, Scaling, Rotation2D and 3D rotations (Quaternion, AngleAxis)
LU	`#include <Eigen/LU>`	Inverse, determinant, LU decompositions with solver (FullPivLU, PartialPivLU)
Cholesky	`#include <Eigen/Cholesky>`	LLT and LDLT Cholesky factorization with solver
Householder	`#include <Eigen/Householder>`	Householder transformations; this module is used by several linear algebra modules
SVD	`#include <Eigen/SVD>`	SVD decompositions with least-squares solver (JacobiSVD, BDCSVD)
QR	`#include <Eigen/QR>`	QR decomposition with solver (HouseholderQR, ColPivHouseholderQR, FullPivHouseholderQR)
Eigenvalues	`#include <Eigen/Eigenvalues>`	Eigenvalue, eigenvector decompositions (EigenSolver, SelfAdjointEigenSolver, ComplexEigenSolver)
Sparse	`#include <Eigen/Sparse>`	Sparse matrix storage and related basic linear algebra (SparseMatrix, SparseVector) (see Quick reference guide for sparse matrices for details on sparse modules)
	`#include <Eigen/Dense>`	Includes Core, Geometry, LU, Cholesky, SVD, QR, and Eigenvalues header files
	`#include <Eigen/Eigen>`	Includes Dense and Sparse header files (the whole Eigen library)

Eigen和Sophus在滤波器上的应用

http://eigen.tuxfamily.org/dox/group__QuickRefPage.html

add subtract	`mat3 = mat1 + mat2;  mat3 += mat1;` `mat3 = mat1 - mat2;  mat3 -= mat1;`
scalar product	`mat3 = mat1 * s1;  mat3 *= s1;  mat3 = s1 * mat1;` `mat3 = mat1 / s1;  mat3 /= s1;`

matrix/vector
products *
	`col2 = mat1 * col1;`
	`row2 = row1 * mat1;  row1 *= mat1;`
	`mat3 = mat1 * mat2;  mat3 *= mat1;`

transposition adjoint *	`mat1 = mat2.transpose();  mat1.transposeInPlace();` `mat1 = mat2.adjoint();  mat1.adjointInPlace();`

dot product
inner product *
	`scalar = vec1.dot(vec2);`
	`scalar = col1.adjoint() * col2;`
	`scalar = (col1.adjoint() * col2).value();`

outer product *	`mat = col1 * col2.transpose();`
norm **normalization** *	`scalar = vec1.norm();  scalar = vec1.squaredNorm()` `vec2 = vec1.normalized();  vec1.normalize(); // inplace`
cross product *	`#include <Eigen/Geometry>` `vec3 = vec1.cross(vec2);`

Eigen和Sophus在滤波器上的应用

https://github.com/strasdat/Sophus

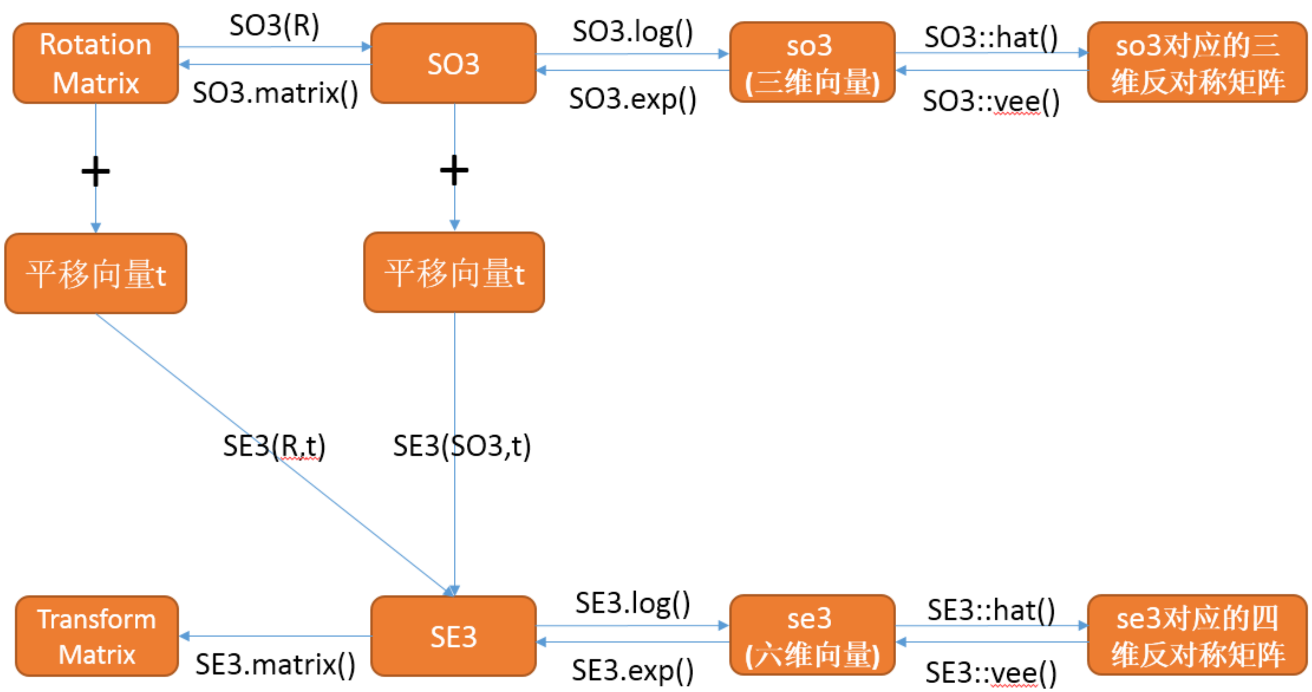
Installation:

sudo apt-get install ros-kinetic-sophus

或者:

```
git clone https://github.com/strasdat/Sophus.git
$ cd Sophus
$ mkdir build
$ cd build
$ cmake ..
$ make
```

Eigen和Sophus在滤波器上的应用



S03, so3, SE3和se3的相互转换关系

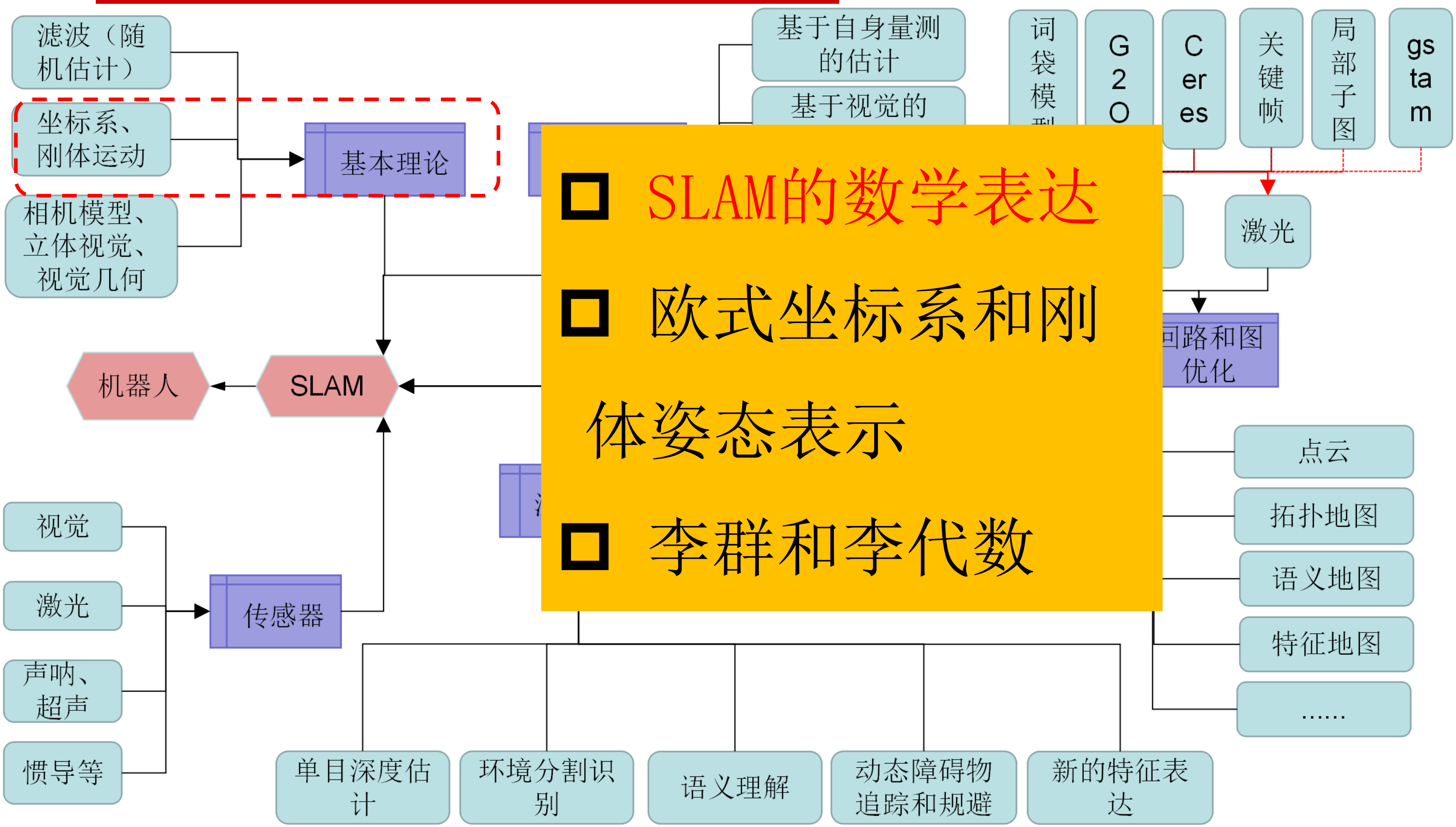
实用参考：http://blog.csdn.net/u011092188/article/details/77833022

Eigen和Sophus在滤波器上的应用

–	功能	函数
1	adjoint Transform	Adj()
2	inverse	inverse()
3	to lie algebra	so3()
4	log map	log()
5	exp map	exp()
6	归一化so3元素	normalize()
7	hat	hat()
8	李括号	lieBranket()

实例

总结



- SLAM的数学表达
- 欧式坐标系和刚体姿态表示
- 李群和李代数

Q&A