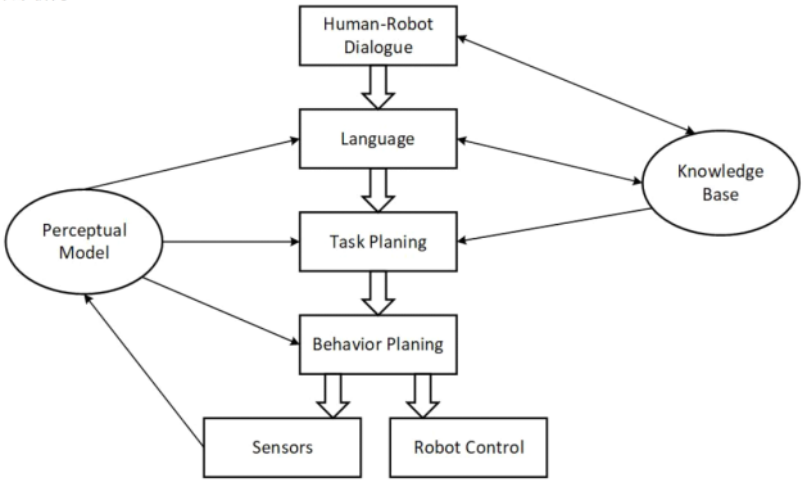
机器人总复习

2019年1月15日 3:15

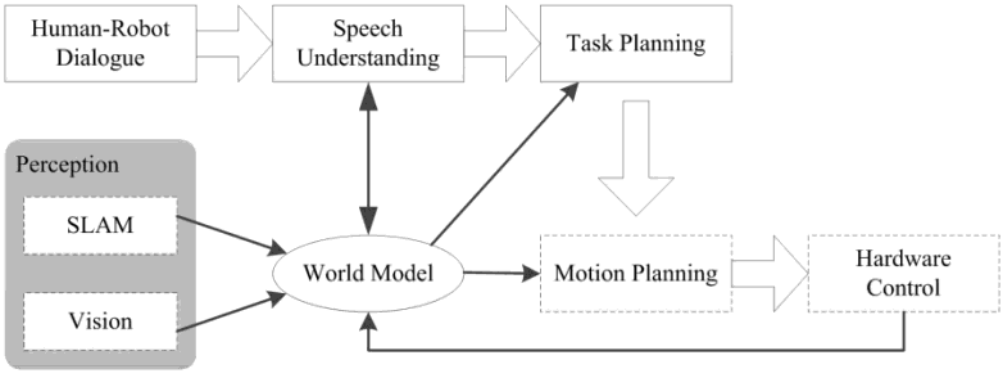
第一章 概述

可佳机器人

基本结构



软件架构



硬件架构

- 底盘双轮驱动
- 机械臂及手爪，能抓握物体
- 升降机构、云台
- Kinect摄像头，高分辨率RBG摄像头
- 激光传感器
- 电池
- 计算单元

地图构建

1. 激光测距仪构建2D网格图
2. 八叉树表示三维环境

视觉

通过**Kinect+CCD相机**获得RGB-D+高质量RGB图像，校准并建立变换关系

路径规划

1. 初始的寻找路径的算法是A*算法。
2. 计算从机器人->目标的路径。
3. 生成一系列有序的路径点，将路径点传给本地规划器。
4. 如果没有找到合适的路径，则将本地地图继续扩大。
5. 如果还是失败，则向人类求助
6. 以上全都失败，重新全局路径规划

物体抓取

通过深度学习和点云图，点云图是立体的，深度学习是神经网络来识别的。

机器人设计三原则

1. 阿西莫夫，一个科幻小说作家
2. 机器人必须不危害人类，也不允许它眼看人将受害而袖手旁观。
3. 机器人必须绝对服从于人类，除非这种服从有害于人类。
4. 机器人必须保护自身不受伤害，除非为了保护人类或者是人类命令它作出牺牲。

第二章

坐标系

世界坐标系

又叫做全局坐标系，建立了描述其它坐标系所需要的参考框架。是我们关注的最大坐标系。

相机坐标系

摄像坐标系是和观察者密切相关的坐标系。该坐标系定义在摄像机的屏幕可视区域。

空间坐标变换

齐次坐标

点

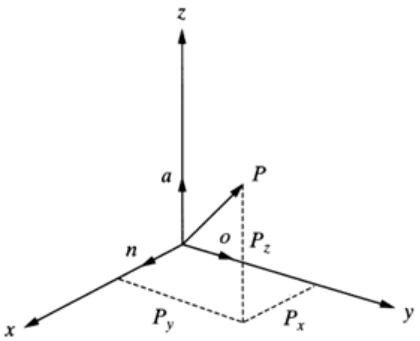
在一个三维向量的末尾添加1，就变成了四维向量，称之为齐次坐标。添加了一维，即多了一个自由度，但是允许把变换写成线性的形式。在齐次坐标中，某个点 x 的每个分量同乘一个非零常数 k 后， 仍然表示的是同一个点。

坐标轴

$$X = [1 \quad 0 \quad 0 \quad 0]^T$$
$$Y = [0 \quad 1 \quad 0 \quad 0]^T$$
$$Z = [0 \quad 0 \quad 1 \quad 0]^T$$

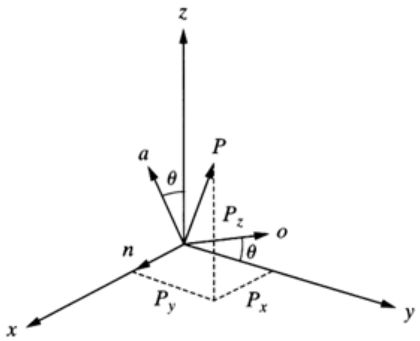
旋转矩阵

两个坐标系原点相同，绕某一个坐标轴进行旋转



旋转前

(a)



旋转后

(b)

绕x，y，z轴旋转θ角的变换矩阵是

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

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

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

$$\begin{bmatrix}p_x\\ p_y\\ p_z\end{bmatrix}=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\theta & -\sin\theta\\ 0 & \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}p_n\\ p_o\\ p_a\end{bmatrix}$$

期中考试考过

性质

1. 是一个行列式为1的正交矩阵。
2. 单位向量之间互相垂直，正交。

齐次坐标变换矩阵

用一个矩阵描述旋转和平移关系，使整个变换关系变成了线性关系。这个矩阵就是齐次坐标变换矩阵。

结构

左上角为旋转矩阵，右侧为平移向量，左下角为0向量，右下角为1.

$$SE(3)=\left\{\boldsymbol{T}=\begin{bmatrix}\boldsymbol{R} & \boldsymbol{t}\\ \mathbf{0}^T & 1\end{bmatrix}\in\mathbb{R}^{4\times4}\middle|\boldsymbol{R}\in SO(3),\boldsymbol{t}\in\mathbb{R}^3\right\}.$$

$$F=\begin{bmatrix}n_x & o_x & \alpha_x & p_x\\ n_y & o_y & \alpha_y & p_y\\ n_z & o_z & \alpha_z & p_z\\ 0 & 0 & 0 & 1\end{bmatrix}$$

逆变换

$$\boldsymbol{T}^{-1} = \begin{bmatrix} \boldsymbol{R}^T & -\boldsymbol{R}^T\boldsymbol{t} \\ \boldsymbol{0}^T & 1 \end{bmatrix}.$$

表示一个反向的变换。

旋转向量

方向为旋转轴，长度为旋转的角度。

旋转向量与矩阵的不同：1. 仅有三个量 2. 无约束 3. 更直观

欧拉角

将旋转分解为三个方向上的转动。

轴可以是定轴或者动轴，顺序亦可以不同。

常见的有yaw-pitch-roll，偏航-俯仰-滚转，东北天。

期中考过

万向锁

欧拉角的奇异性问题，在特定值时，旋转自由度减1。

万向锁的情况：在旋转是动态的时候，即绕旋转之后的旋转轴旋转的。在三个旋转次序间，只有中间的那个旋转次序，旋转角是±90°时，才会发生万向锁。此时第一个旋转轴和第三个旋转轴处于同一个平面，丢失一个维度。

讲解视频

四元数

四元数既是紧凑的，也没有奇异性。

一个四元数 q 拥有一个实部和三个虚部。$\mathbf{q}=q_0+q_1i+q_2j+q_3k$

其中i,j,k 为四元数的三个虚部。这三个虚部满足关系式：

$$\begin{cases} i^2 = j^2 = k^2 = -1 \\ ij = k, ji = -k \\ jk = i, kj = -i \\ ki = j, ik = -j \end{cases}$$

能用单位四元数表示三维空间中任意一个旋转。

假设某个旋转是绕**单位向量** $\mathbf{n}=[n_x,n_y,n_z]^T$进行了角度为 θ 的旋转，那么这个旋转的四元数形式为

$$q = [cos\frac{\theta}{2}, n_x sin\frac{\theta}{2}, n_y\frac{\theta}{2}, n_z\frac{\theta}{2}]^T$$

四元数计算乘法：

$$q_b q_a q_b' = [s_b, \boldsymbol{b}][0, \boldsymbol{a}][s_b, -\boldsymbol{b}]$$

为了将实部变为0，需要再乘以四元数的共轭。

例题

给定两个坐标系，算齐次坐标变换矩阵

第三章

图像处理的流程

图像采集、图像增强、图像复原、形态学处理、分割、表示&描述、对象识别

可能有的：彩色图像处理、图像压缩

颜色空间

RGB-红绿蓝

HSV-色度，饱和度，亮度

为什么要有这些颜色空间

以RGB和HSV为例，RGB是面向硬件的，HSV是面向用户的，更符合人的生理特征。

卷积

二维连续函数的卷积

$$h(x,y)=f(x,y)*g(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(x',y')g(x-x',y-y')dx'dy'$$

图像离散系统可写成：

$$h[i,j]=f[i,j]*g[i,j]=\sum_{k=0}^{n-1}\sum_{l=0}^{m-1}f[k,l]g[i-k,j-l]$$

卷积核

卷积核就是图像处理时，给定输入图像，在输出图像中每一个像素是输入图像中一个小区域中像素的加权平均，其中权值由一个函数定义，这个函数称为卷积核。

图像预处理

1. 方法
    1. 空间域法和频率域法
        1. 空间域法是直接在空间域对图像像素运算处理；
        2. 频率域法是先对图像做某种变换（DFT、DCT、DWT、K-L），然后在变换域对图像的变换值进行运算，最后将计算后的图像逆变换到空间域。
    2. 全局运算和局部运算
    3. 灰度图像和彩色图像
2. 空间域
    1. *噪声*
        1. 白噪声：均值为0，分布随机：均值滤波器
        2. 椒盐噪声：含有随机出现的黑白亮度值。中值滤波器，边缘保持滤波器。
        3. 高斯噪声：含有亮度服从高斯或正态分布的噪声。高斯噪声是很多传感器噪声的模型，如摄像机的电子干扰噪声。高斯滤波器
    2. *滤波器*

        滤波器的本质是一个卷积运算

        1. 均值滤波器：线性

            最简单均值滤波器是局部均值运算，即每一个像素只用其局部邻域内所有值的平均值来置换：

            $$h[i,j]=\frac{1}{M}\sum_{(k,l)\in N}f[k,l]$$

            也可以加权，一种只有一个峰值，并且在水平和垂直方向上对称的典型权值模板：

            $$\begin{bmatrix}\frac{1}{16}&\frac{1}{8}&\frac{1}{16}\\\frac{1}{8}&\frac{1}{4}&\frac{1}{8}\\\frac{1}{16}&\frac{1}{8}&\frac{1}{16}\end{bmatrix}$$

2. 高斯平滑滤波器：线性
    a) 高斯平滑滤波器是一类根据高斯函数的形状来选择权值的线性滤波器。
    b) 高斯平滑滤波器对去除服从正态分布的噪声是很有效的。
    c) 思想：随着距离的增大，对中心像素的影响减小。

$$g[i,j]=e^{-\frac{(i^2+j^2)}{2\sigma^2}}$$

3. 中值滤波器：非线性
    1. 按亮度值大小排列像素点
    2. 选择排序像素集的中间值作为点[i，j]的新值
    3. 一般采用奇数点的邻域计算中值，如果像素点数为偶数，则中值就取排序像素中间两点的平均值。

    中值滤波在一定条件下可以克服线性滤波器所造成的图像细节模糊，而对滤除脉冲干扰很有效。

3. 寻找边缘

1. 一阶边缘检测算子

1. Roberts交叉算子

$$G[i,j]=\left|f[i,j]-f[i+1,j+1]\right|+\left|f[i+1,j]-f[i,j+1]\right|$$

模板卷积：

$$G[i,j]=\left|G_x\right|+\left|G_y\right|$$

Gx和Gy的模板是

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

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

2. Sobel算子

Sobel算子为梯度幅度值计算

$$M=\sqrt{s_x^2+s_y^2}$$

$$s_x=\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix}\quad s_y=\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}$$

3. Prewitt算子

Prewitt算子和Sobel算子方程一样

$$M=\sqrt{s_x^2+s_y^2}$$

$$s_x=\begin{bmatrix}-1 & 0 & 1\\ -1 & 0 & 1\\ -1 & 0 & 1\end{bmatrix}\quad s_y=\begin{bmatrix}1 & 1 & 1\\ 0 & 0 & 0\\ -1 & -1 & -1\end{bmatrix}$$

2. 二阶边缘检测算子

1. *拉普拉斯算子*

拉普拉斯算子是二阶导数的二维等效式

$$\nabla^2 f(x,y) = L(x,y)$$
$$= \{[f(x+1,y) - f(x,y)] - [f(x,y) - f(x-1,y)]\}$$
$$+ \{[f(x,y+1) - f(x,y)] - [f(x,y) - f(x,y-1)]\}$$
$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

$$\nabla^2 \approx \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

有时候为了突出中心点，可能会采用更大的权值。

2. *Canny边缘检测（期中考了）*

1. 用高斯平滑滤波器平滑图像
2. 用一阶偏导数的有限差分来计算图像的梯度的幅度和方向
3. 对梯度幅度值进行非极大值抑制（NMS）
4. 用双阈值算法检测和连接边缘

3. 灰度

1. 阶跃不连续：图像亮度在不连续处两边的像素灰度值有着显著的差异。
2. 线条不连续：图像亮度突然从一个值变化到另一个值，保持一个较小的行程后又返回到原来的值。
3. 老师说的灰度检测边缘，就是指把图像转为灰度，然后直观的去看就好了

4. 边缘检测算子的用法

1. 使用如上的卷积模板对图像进行卷积操作。
2. 边缘就会更突出。展示即可。

轮廓表示

链码：沿着轮廓记录边缘表的一种表示方法。

链码：规定了边缘表中每一个边缘点所对应的轮廓方向，轮廓方向被量化为4邻点链码或8邻点链码

一般以右下为起点。

差分链码具有旋转不变性，可用于手写数字识别。

例题

曲线链码A：

6 0 2 2 2 2 2 0 2 1 0 1 3 4 4 4 4 4 4 5 4 5 7 7 0 1 2

差分链码A：

2 2 0 0 0 0 6 2 7 7 1 2 1 0 0 0 0 0 1 7 1 2 0 1 1 1

曲线链码B：

0 2 4 4 4 4 4 2 4 3 2 3 5 6 6 6 6 6 6 7 6 7 1 1 2 3 4

差分链码B:

2 2 0 0 0 0 6 2 7 7 1 2 1 0 0 0 0 0 1 7 1 2 0 1 1 1

第四章：视觉里程计

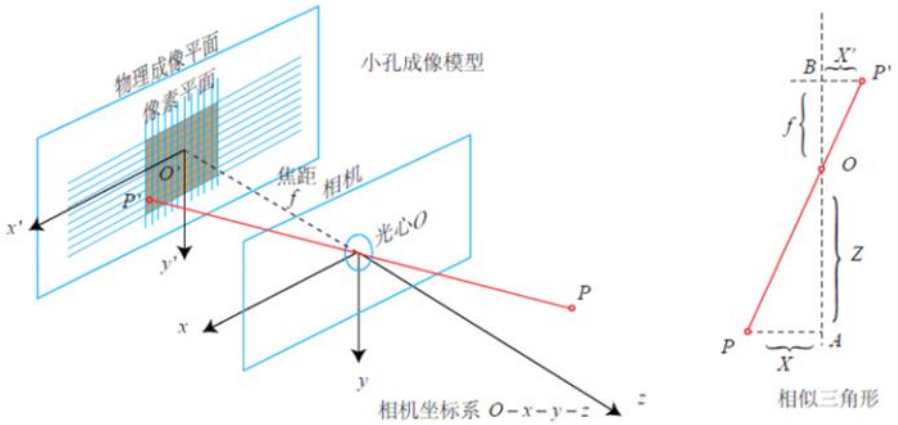
相机模型

要求掌握针孔相机模型，双目相机模型，深度相机模型。

针孔相机模型

原理：小孔成像。

内参



由相似三角形，可以由相机坐标系->物理成像平面的变换。Z是物体距离小孔的距离，即景深。

$$X' = f\frac{X}{Z}$$
$$Y' = f\frac{Y}{Z}$$

物理成像平面坐标系的原点在正对着针孔处。

像素平面的原点在左上角。

由物理成像平面到像素平面，有一个缩放（米->像素）和平移（中间->左上角）的过程。

缩放：在u轴上缩放了α倍，在v轴上缩放了β倍。则有：

$$\begin{cases} u = \alpha X' + c_x \\ v = \beta Y' + c_y \end{cases}$$

和上面的X'联立，有

$$\begin{cases} u = f_x\frac{X}{Z} + c_x \\ v = f_y\frac{Y}{Z} + c_y \end{cases}$$

fx，fy，cx，cy叫内参。

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{Z}\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \triangleq \frac{1}{Z}KP.$$

中间的叫内参矩阵。（不必记形式，考试中如果涉及，会给）

外参

由世界坐标系->相机坐标系的变换。

坐标系变换需要1. 旋转矩阵。 2. 平移向量。

因此，旋转矩阵R和平移向量t组成的T是相机当前的姿态，称为外参。

畸变

两种畸变：径向畸变，切向畸变。

径向畸变

原因：针孔前引入了一个镜头。

失真：桶形失真，枕形失真。

切向畸变

原因：组装过程中，成像平面与透镜面不平行。

解决办法

5个畸变参数。

小结

1. 首先，世界坐标系下有一个固定的点 P，世界坐标为 $\boldsymbol{P}_w$；
2. 由于相机在运动，它的运动由 $\boldsymbol{R},\boldsymbol{t}$ 或变换矩阵 $\boldsymbol{T}\in SE(3)$ 描述。P 的相机坐标为：$\tilde{\boldsymbol{P}}_c=\boldsymbol{R}\boldsymbol{P}_w+\boldsymbol{t}$。
3. 这时的 $\tilde{\boldsymbol{P}}_c$ 仍有 X,Y,Z 三个量，把它们投影到归一化平面 $Z=1$ 上，得到 P 的归一化相机坐标：$\boldsymbol{P}_c=[X/Z,Y/Z,1]^T$①。
4. 最后，P 的归一化坐标经过内参后，对应到它的像素坐标：$\boldsymbol{P}_{uv}=\boldsymbol{K}\boldsymbol{P}_c$。

双目相机模型

双目相机一般由左眼和右眼两个水平放置的相机组成。

需要掌握：双目相机测距原理。

深度相机模型

有一个深度相机。

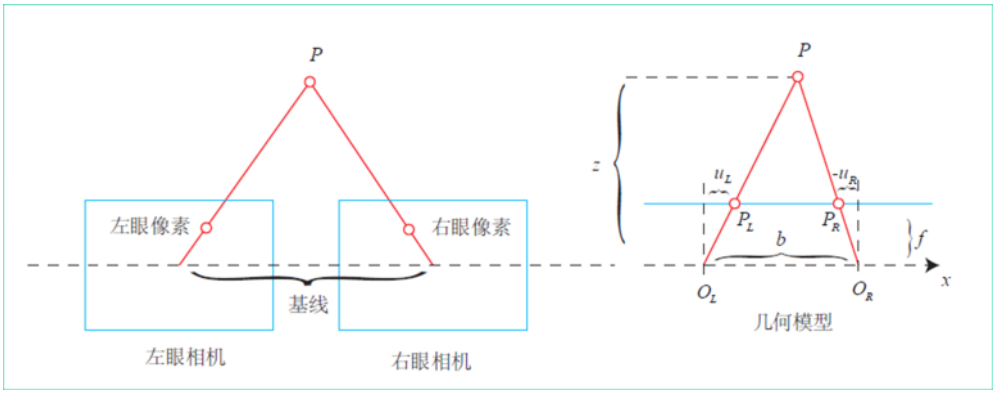
需要掌握：深度相机测距方法。

深度测量

散焦深度法

通过改变针孔光圈的大小，改变不同距离的物体，成像的锐利程度，以获得深度。物体接近透镜时，散焦变化明显，有景深限制。

双目视觉法



根据左右相机的视差来求景深。

公式

$$z=\frac{fb}{d},\quad d=u_L-u_R.$$

O_L，O_R为左右光圈中心，f为焦距，u_L 和 u_R 为成像平面的坐标。注意u_R应该为负数。 d 为左右图的横坐标之差，称为视差（Disparity），

两个相机的光圈中心的距离叫基线，记为b，baseline，基线不是一条线，是

一个距离。

深度相机法

1. 红外结构光法：相机根据返回的结构光图案，计算物体离自身的距离。
   Kinect 1 代
2. 飞行时间法：相机向目标发射脉冲光，然后根据发送到返回之间的光束飞行时间，确定距离。

特征点法视觉里程计

特征点：图像当中具有代表性的部分。

可重复性：在不同图像中能重现
可区别性：不同的点有不同的表达
高效：特征点数量应远小于像素的数量
本地：特征仅与一小片图像区域相关

特征点的信息

位置、大小、方向、评分等——关键点(Key point)
特征点周围的图像信息——描述子（Descriptor）

例子：SIFT/SUF/ORB

Harris Detector暂时不考

Harris角点检测：具有平移不变性和旋转不变性，但是对尺度很敏感。

FAST角点

是一种关键点，没有描述子信息。

如果一个像素与淋雨的像素差别较大（过亮或者过暗），则他有可能是个角点。

以p为中心，3为半径的圆上的16个像素点，有N个符合判据，则p为特征点，N为12,11,9.

FAST算法

1. 在图像中选取像素 p，假设它的亮度为 I_p。
2. 设置一个阈值 T(比如 I_p 的 20%)。
3. 以像素 p 为中心, 选取半径为 3 的圆上的 16 个像素点。
4. 假如选取的圆上，有连续的 N 个点的亮度大于 $I_p + T$ 或小于 $I_p - T$，那么像素 p 可以被认为是特征点 (N 通常取 12，即为 FAST-12。其它常用的 N 取值为 9 和 11，他们分别被称为 FAST-9，FAST-11)。
5. 循环以上四步，对每一个像素执行相同的操作。

ORB特征

关键点：Oriented FAST

描述子：BRIEF

Oriented FAST

Oriented FAST在尺度上引入图像金字塔，对图像进行不同层次的降采样，如果都是关键点的话，则认为是关键点。

在FAST基础上计算旋转信息：灰度质心法。

灰度质心法

1. 在一个小的图像块 B 中，定义图像块的矩为：

$$m_{pq} = \sum_{x,y \in B} x^p y^q I(x,y), \quad p,q = \{0,1\}.$$

2. 通过矩可以找到图像块的质心：

$$C = (\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}).$$

3. 连接图像块的几何中心 O 与质心 C，得到一个方向向量 $\overrightarrow{OC}$，于是特征点的方向可以定义为：

$$\theta = \arctan(m_{01}/m_{10}).$$

BRIEF

BRIEF 是一种二进制描述子，它的描述向量由许多个 0 和 1 组成，这里的 0 和 1 编码了关键点附近两个像素（比如说 p 和 q）的大小关系：如果 p 比 q 大，则取 1，反之就取 0。

特征匹配

最简单的：暴力匹配。速度慢。

改进：FLANN快速近似最近邻。

增量式运动估计（只考ICP，以ICP为例）

ICP中只考SVD方法。

1. 计算两组点的质心位置 $\boldsymbol{p}, \boldsymbol{p}'$，然后计算每个点的**去质心坐标**：

$$\boldsymbol{q}_i = \boldsymbol{p}_i - \boldsymbol{p}, \quad \boldsymbol{q}'_i = \boldsymbol{p}'_i - \boldsymbol{p}'.$$

2. 根据以下优化问题计算旋转矩阵：

$$\boldsymbol{R}^* = \arg\min_{\boldsymbol{R}} \frac{1}{2} \sum_{i=1}^{n} \|\boldsymbol{q}_i - \boldsymbol{R}\boldsymbol{q}'_i\|^2.$$

3. 根据第二步的 $\boldsymbol{R}$，计算 $\boldsymbol{t}$：

$$\boldsymbol{t}^* = \boldsymbol{p} - \boldsymbol{R}\boldsymbol{p}'.$$

小结

特征点法流程：

在图像中提取特征点并计算特征描述

在不同图像中寻找特征匹配

利用匹配点信息计算相机位姿

Input

左图像、右图像序列、左右相机的内参、外参

Output

图像对之间的位姿关系、相机轨迹(也可以生成基于特征点的稀疏3d地图，optional)

特征点法视觉里程计工作流程（与前面的类似，理解就好）：

获得左右图像、图像校正

提取图像特征点

左右图像特征匹配，获取特征点坐标

前后图像特征匹配

增量式的运动估计

光流法视觉里程计

LK光流法

概述：计算两帧在时间t 到t+δt之间每个每个像素点位置的移动。

光流：追踪源图像某个点在其他图像中的运动

三个假设条件

亮度恒定：就是同一点随着时间的变化，其亮度不会发生改变，用于得到光流法基本方程；

小运动：这个必须满足，就是时间的变化不会引起位置的剧烈变化，这样灰度才能对位置求偏导（换句话说，小运动情况下我们才能用前后帧之间单位位置变化引起的灰度变化去近似灰度对位置的偏导数）；特征点在前一帧和后一帧还大量存在。

空间一致：一个场景上邻近的点投影到图像上也是邻近点，且邻近点速度一致。**这是Lucas-Kanade光流法特有的假定。**

计算&&推导

设 t 时刻位于 x, y 处像素点的灰度值为 $\boldsymbol{I}(x,y,t)$.

在 t+dt 时刻，该像素运动到了 $\boldsymbol{I}(x+\mathrm{d}x, y+\mathrm{d}y, t+\mathrm{d}t)$

希望计算运动 dx/dt, dy/dt

灰度不变假设：$\boldsymbol{I}(x+\mathrm{d}x, y+\mathrm{d}y, t+\mathrm{d}t) = \boldsymbol{I}(x,y,t)$.

注意：灰度不变是一种理想的假设，实际当中由于高光/阴影/材质/曝光等不同，很可能不成立。

对左边进行泰勒展开，保留一阶项，得：

$$\boldsymbol{I}(x+\mathrm{d}x, y+\mathrm{d}y, t+\mathrm{d}t) \approx \boldsymbol{I}(x,y,t) + \frac{\partial \boldsymbol{I}}{\partial x}\mathrm{d}x + \frac{\partial \boldsymbol{I}}{\partial y}\mathrm{d}y + \frac{\partial \boldsymbol{I}}{\partial t}\mathrm{d}t.$$

因为我们假设了灰度不变，于是下一个时刻的灰度等于之前的灰度，从而

$$\frac{\partial \boldsymbol{I}}{\partial x}\mathrm{d}x + \frac{\partial \boldsymbol{I}}{\partial y}\mathrm{d}y + \frac{\partial \boldsymbol{I}}{\partial t}\mathrm{d}t = 0.$$

两边除以 $\mathrm{d}t$，得：

$$\frac{\partial \boldsymbol{I}}{\partial x}\frac{\mathrm{d}x}{\mathrm{d}t} + \frac{\partial \boldsymbol{I}}{\partial y}\frac{\mathrm{d}y}{\mathrm{d}t} = -\frac{\partial \boldsymbol{I}}{\partial t}.$$

其中 dx/dt 为像素在 x 轴上运动速度，而 dy/dt 为 y 轴速度，把它们记为 u; v。

由**空间一致**假设，在一个窗口内的像素具有相同的运动

考虑一个大小为 $w \times w$ 大小的窗口，它含有 w^2 数量的像素。由于该窗口内像素具有同样的运动，因此我们共有 w^2 个方程：

$$\begin{bmatrix} \boldsymbol{I}_x & \boldsymbol{I}_y \end{bmatrix}_k \begin{bmatrix} u \\ v \end{bmatrix} = -\boldsymbol{I}_{tk}, \quad k = 1, \ldots, w^2. \tag{8.6}$$

记：

$$\boldsymbol{A} = \begin{bmatrix} [\boldsymbol{I}_x, \boldsymbol{I}_y]_1 \\ \vdots \\ [\boldsymbol{I}_x, \boldsymbol{I}_y]_k \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} \boldsymbol{I}_{t1} \\ \vdots \\ \boldsymbol{I}_{tk} \end{bmatrix}. \tag{8.7}$$

于是整个方程为：

$$\boldsymbol{A} \begin{bmatrix} u \\ v \end{bmatrix} = -\boldsymbol{b}. \tag{8.8}$$

这是一个关于 u, v 的超定线性方程，传统解法是求最小二乘解。最小二乘在很多时候都用到过：

$$\begin{bmatrix} u \\ v \end{bmatrix}^* = -\left(\boldsymbol{A}^T \boldsymbol{A}\right)^{-1} \boldsymbol{A}^T \boldsymbol{b}. \tag{8.9}$$

这个结果背也得背下来，考试要考！！

超定方程组

概述

对于形如Am×nX=b的方程，考虑测量数据和所需解的参数之间的关系，该方程的解可以分为以下几种情况：

1. 如果$m < n$，未知数大于方程数。那么解不唯一，存在一个解矢量空间。
2. 如果$m = n$，那么只要A可逆（非奇异，也就是满秩）就有唯一解，解为$\mathbf{x} = A^{-1}\mathbf{b}$。
3. 如果$m > n$，方程数大于未知数。方程一般没有解，除非$\mathbf{b}$属于A的列向量组成的子空间。

考虑m≥n并且R(A)=n的情况，如果解不存在，但找一个最接近Am×nX=b的解矢量仍然有意义，这个方程成为超定方程（方程大于未知数）。也就是说，我们寻找一个向量x使得 ||Ax−b||最小，这里的||·||表示矢量范数。这样的xx称为该超定方程组的最小二乘解。

最小二乘法

目标：求误差的最小平方和。

如果矩阵是可逆的，线性模型最小二乘的解是closed-form（闭式解）即

$$\theta = (X^T X)^{-1} X^T Y$$

定理

x*是Ax=b的最小二乘解的充要条件为x*是 $A^TAx=A^Tb$ 的解。

证明（不考）

证　充分性　若存在 n 维向量 x^* 使 $A^TAx^*=A^Tb$，

任取一 n 维向量 $\bar{x} \neq x^*$，令 $y=\bar{x}-x^*$，则 $y\neq 0$，且

$$\begin{aligned}\|b-A\bar{x}\|_2^2 &= \|b-Ax^*-Ay\|_2^2 \\ &= (b-Ax^*-Ay, b-Ax^*-Ay) \\ &= (b-Ax^*, b-Ax^*)-2(Ay, b-Ax^*)+(Ay, Ay) \\ &= \|b-Ax^*\|_2^2-2y^TA^T(b-Ax^*)+\|Ay\|_2^2 \\ &= \|b-Ax^*\|_2^2+\|Ay\|_2^2 \geq \|b-Ax^*\|_2^2\end{aligned}$$

所以 x^* 是 $Ax=b$ 的最小二乘解.

必要性　误差向量 $r=b-Ax$ 的第 i 个分量为

$$r_i = b_i - \sum_{k=1}^{n} a_{ik}x_k \quad (i=1,2,\ldots,m),$$

记　$$I = I(x_1,x_2,\ldots,x_n) = \|r\|_2^2 = \sum_{i=1}^{m}(b_i-\sum_{k=1}^{n}a_{ik}x_k)^2$$

误差最小，由多元函数求极值的必要条件，可得

$$\frac{\partial I}{\partial x_j} = -2\sum_{i=1}^{m}(b_i-\sum_{k=1}^{n}a_{ik}x_k)a_{ij} = 0 \quad (j=1,2,\ldots,n)$$

设解为　$x^* = (x_1^*, x_2^*, \ldots, x_n^*)^T$

例题

例　求超定方程组

$$\begin{cases} 2x_1+4x_2=11 \\ 3x_1-5x_2=3 \\ x_1+2x_2=6 \\ 2x_1+x_2=7 \end{cases}$$

的最小二乘解，并求误差平方和.

解　方程组写成矩阵形式为

$$Ax = \begin{bmatrix} 2 & 4 \\ 3 & -5 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 11 \\ 3 \\ 6 \\ 7 \end{bmatrix} = b$$

$$\begin{bmatrix} 2 & 4 \\ 3 & -5 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 11 \\ 3 \\ 6 \\ 7 \end{bmatrix}$$

正规方程组为 $A^TAx=A^Tb$，即

$$\begin{bmatrix} 2 & 3 & 1 & 2 \\ 4 & -5 & 2 & 1 \end{bmatrix}\begin{bmatrix} 2 & 4 \\ 3 & -5 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 2 \\ 4 & -5 & 2 & 1 \end{bmatrix}\begin{bmatrix} 11 \\ 3 \\ 6 \\ 7 \end{bmatrix}$$

即有 $\begin{bmatrix} 18 & -3 \\ -3 & 46 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 51 \\ 48 \end{bmatrix}$ 解得最小二乘解为 $\begin{cases} x_1 = 3.0403 \\ x_2 = 1.2418 \end{cases}$

此时有 $\begin{cases} 2x_1 + 4x_2 = 11.0478 = b_1^* \\ 3x_1 - 5x_2 = 2.9119 = b_2^* \\ x_1 + 2x_2 = 5.5239 = b_3^* \\ 2x_1 + x_2 = 7.3224 = b_4^* \end{cases}$

故误差平方和为 $I = \|r\|_2^2 = \sum_{i=1}^{m}(b_i - \sum_{k=1}^{n} a_{ik}x_k)^2 = \sum_{i=1}^{m}(b_i - b_i^*)^2$

$$I = (11 - 11.0478)^2 + (3 - 2.9119)^2 + (6 - 5.5239)^2 + (7 - 7.3224)^2$$
$$= 0.34065942$$

范数

L1-范数

L1-范数

$$||x||_1 = \sum_{i=1}^{n} |x_i|$$

L2-范数

L2-范数

$$||x||_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

P-范数

$$||x||_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$$

小结

光流法视觉里程计过程

图像获取：单目照相机、双目照相机或者全向照相机；

图像校正：使用一些图像处理技术来去除透镜畸变；

特征检测：确定感兴趣的描述符，在帧与帧之间匹配特征并构建光流场。

使用相关性来度量两幅图像间的一致性，并不进行长时间的特征跟踪

特征提取、匹配（Lucas–Kanade method）

构建光流场

检查光流场向量是否存在潜在的跟踪误差，移除外点

由光流场估计照相机的运动

可选方法1：使用卡尔曼滤波进行状态估计

可选方法2：查找特征的几何与3D属性，以最小化基于相邻两帧之间的重投影误差的罚函数值。这可以通过数学上的最小化方法或随机采样方法来完成

周期性的重定位跟踪点。（前面的特征点，跟丢了太多之后，重新选择关键帧，重新选择特征点，进行追踪）

光流法的缺点

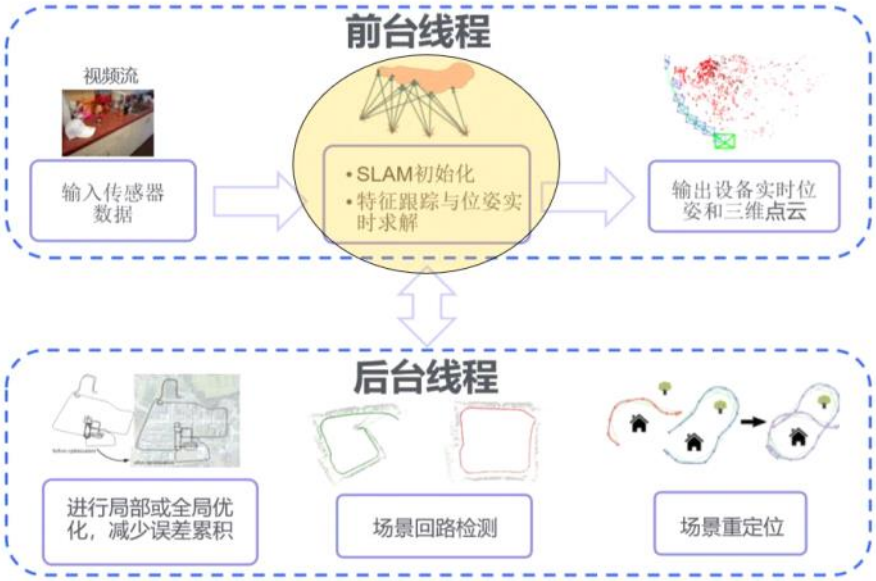
光流仅估计了像素间的平移，没有用到相机本身的几何结构

没有考虑到相机的旋转和图像的缩放

对于边界上的点，光流不好追踪

第五章：SLAM

视觉SLAM的基本框架



地图类型

栅格地图

每个栅格以概率的形式表示

3D的栅格地图采用八叉树存储

适用于路径规划与避障

特征地图

地图比较稀疏

适用于存在较多特征的场景

位姿图

仅包含机器人的位姿信息

有效表达位姿之间的约束

可以与栅格地图的特征地图结合

其他地图

用于场景构建

TSDF

初始化

SfM：从运动中恢复结构，相机标定的过程。只是一个初始标定。

SLAM：相机内参通常会预先标定好，固定不变。

前端

特征点检测

特征匹配

视觉里程计（3D-2D运动估计）

增量式运动估计

SLAM相当于一个前端+全局优化，全局优化中最重要的是回环检测

后端

回环检测

全局优化（没讲）

构建环境地图

回环检测

意义

在传统的视觉SLAM中，当前帧的位姿估计是依赖于之前帧的位姿估计，整

个相机定位过程中误差会不断积累。
通过回环检测，可以有效消除累计误差。

在SLAM过程中，回环检测显得尤为重要，当失去回环检测时，整个SLAM过程就会退化成一个视觉里程计。

词袋模型

TF-IDF

对于TF，分子为当前图片中出现单词i的个数，分母为总共包含的单词个数，这是在线得到的；IDF是词典的属性，已经离线生成，不会发生改变，分子为生成词典时所包含的训练图片数量，分母为出现单词i的图片个数，这个比重分量主要为了凸显差异型，认为在训练时经常出现的单词差异性较小，权重赋予的小一些，举例：如果一个单词在每一张图片都能出现，那么词典所给予的权重分量应该是0，因为太常见了，所以会忽略它

字典的构建

是一个预先训练的结果。

1. 收集数据集中的特征点
2. 计算特征点的描述子
3. 对描述子进行聚类
4. 将聚类中心作为单词

单词

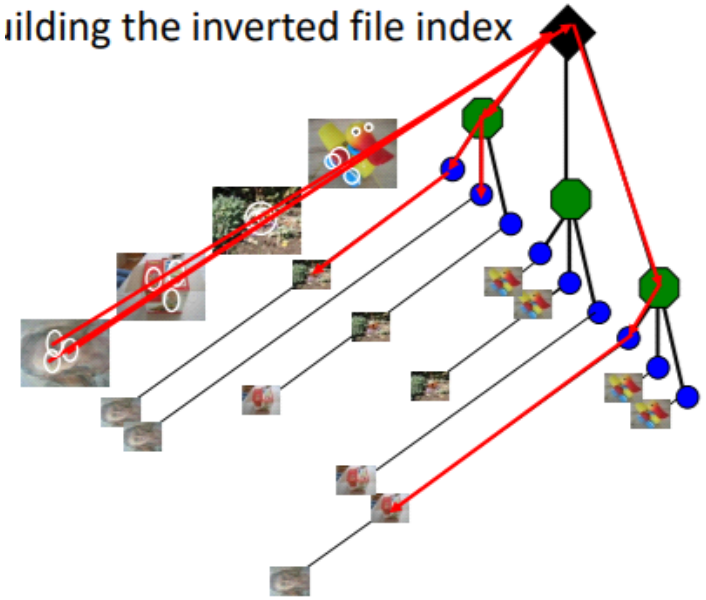
多个特征点组成一个单词。

投票

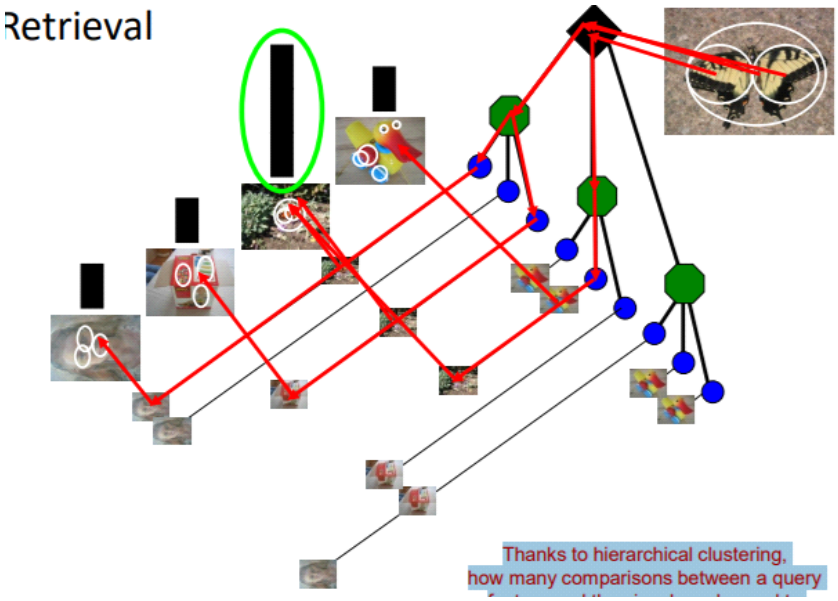
1、逆序索引单词涵盖了所有出现的单词

2、每一个单词指向包含它的一系列图像，这些图像来自图像数据库（DB），在机器人导航并收集新图像时，DB增长。

3、投票数组：具有与数据库中图像一样多的单元格，当前图像中的每一个单词，都给多个图像投票。



每个单词背后都有一列图片。



新图片来了之后，新图片的每个单词会给这些图片投票，最高的几个胜出。

细化得分，选出最相近的图

每个单词的权重有一个归一化过程，权重的组成由tf和idf两部分构成。

$$w_t^i = \mathrm{tf}\left(i, I_t\right) \times \mathrm{idf}\left(i\right), \quad w_t^i = w_t^i / \sum w_t^i$$

$$\mathrm{tf}\left(i, I_t\right) = \frac{n_{iI_t}}{n_{I_t}}, \quad \text{在生成Bow向量时产生的权重分量}$$

$$\mathrm{idf}\left(i\right) = \log \frac{N}{n_i}, \quad \text{生成字典时产生的权重分量}$$

$$\eta_i = \mathrm{TF}_i \times \mathrm{IDF}_i$$

每幅图像的词袋

$$A = \{(w_1, \eta_1), (w_2, \eta_2), \ldots, (w_N, \eta_N)\} \triangleq \boldsymbol{v}_A$$

比较两幅图像VA VB的相似性：

$$s\left(\boldsymbol{v}_A-\boldsymbol{v}_B\right)=2\sum_{i=1}^{N}\left|\boldsymbol{v}_{Ai}\right|+\left|\boldsymbol{v}_{Bi}\right|-\left|\boldsymbol{v}_{Ai}-\boldsymbol{v}_{Bi}\right|$$

SLAM运行结果

计算自身位置

构建环境地图

SLAM举例

PTAM

ORB-SLAM

PKSLAM

LSD-SLAM

DSO

SVO

RDSLAM

回环检测的流程

组匹配

时间一致性校验

结构一致性校验

第六章：路径规划

路径规划的概念

依据某个或某些优化准则(如工作代价最小、行走路线最短、行走时间最短等),在其工作空间中找到一条从起始状态到目标状态的能避开障碍物的最优路径。

寻找路径算法

Dij算法

Dijkstra算法从物体所在的初始点开始，访问图中的结点。它迭代检查待检查结点集中的结点，并把和该结点最靠近的尚未检查的结点加入待检查结点集。该结点集从初始结点向外扩展，直到到达目标结点。

BFS算法

它选择离目标最近的结点。基于贪心策略。仅考虑到达目标的代价，忽略当前已花费的代价。

A*算法

A*=dijkstra+BFS算法

各种距离

曼哈顿距离

```
function heuristic(node) =
  dx = abs(node.x - goal.x)
  dy = abs(node.y - goal.y)
  return D * (dx + dy)
```

x坐标和y坐标差的绝对值之和。

对角线距离

```
function heuristic(node) =
  dx = abs(node.x - goal.x)
```

```
    dy = abs(node.y - goal.y)
return D * (dx + dy) + (D2 - 2 * D) * min(dx, dy)
```

曼哈顿距离减去（2*边长-对角线）*min(dx,dy)

欧几里得距离

```
function heuristic(node) =
    dx = abs(node.x - goal.x)
    dy = abs(node.y - goal.y)
    return D * sqrt(dx * dx + dy * dy)
```

欧几里得距离平方

```
function heuristic(node) =
    dx = abs(node.x - goal.x)
    dy = abs(node.y - goal.y)
    return D * (dx * dx + dy * dy)
```

Breaking Ties

稍微改变argmin[f(n)] = g(n) + h(n)的h的衡量单位。

heuristic *= (1.0 + p)

选择因子p使得p < 移动一步（step）的最小代价 / 期望的最长路径长度

随机路图法PRM

1. 在无障碍空间中随机采样n点。
2. 把n个点按照可连通性连接，获得概率路线图G。
3. 设定起点终点，分别找出离起点和终点最近的图节点V∈G。
4. 以两个节点为起点和终点使用A*或Dijkstra算法搜索路径。

可视图法

1. 取出地图中空白区域。
2. 将空白区域边界的角点V取出，两两间按连通性连接成G。
3. 把起点与终点作为节点按连通性加入G。
4. 使用A*或Dijkstra算法搜索路径。

局部避障

Bug算法

绕障碍物一圈。然后再从距目标最近的点离开。

Bug2

能够直接移动至目标时，就立即分离。（半圈）

人工势场法

- Artificial Potential Field（APF）算法流程：
  1. 设计目的地和障碍物的势场公式；
  2. 用势场梯度计算机器人的参考速度；
- 计算方法：

目的地吸引：

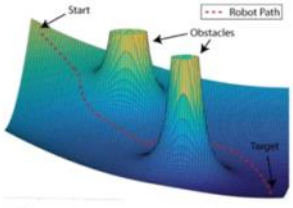
$$\vec{v}_d^{att}(\vec{p}_d,\vec{p}_t)=-\lambda_1(\vec{p}_d-\vec{p}_t)$$

障碍物排斥：

$$\vec{v}_d^{rep}(\vec{p}_d,\vec{p}_o)=-\eta_1\frac{\vec{p}_{do}}{\|\vec{p}_{do}\|^4}$$

再把两个式子求和即可得到参考速度。

优点	缺点
不需要再计算轨迹，计算量小；	易陷入局部最低点； 无法保证绝对无碰撞 狭窄处无法通过； 狭窄通道中摆动；



图片来源：A Novel Potential Field Controller for Use on Aerial Robots

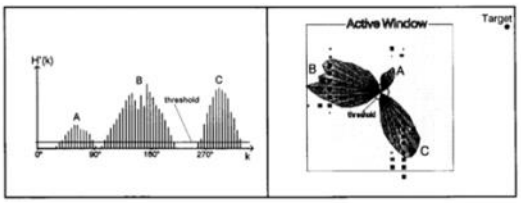
向量场直方图

Vector Field Histogram（VFH）算法流程：

a. 根据不同角度方向障碍物距离，计算直方图；

$$m_{i,j} = (c_{i,j})^2 (a - bd_{i,j})$$

b. 根据直方图，阈值以下的空间即为安全区空间；



图片来源：The vector field histogram-fast obstacle avoidance for mobile robots

优点	缺点
计算量小；	阈值选择对避障影响很大； 没考虑运动限制；

由这个直方图，我们可以计算操纵方向。首先确定使车辆通过的足够大的所有开放通路，然后对每一个这样的候选通路计算其费用函数，选择具有最低费用的通路。费用函数 G 有三项：

$$G = a \cdot \text{目标方向} + b \cdot \text{轮子方向} + c \cdot \text{以前方向} \quad (6.11)$$

目标方向＝与目标一致的机器人路径；

轮子方向＝新方向和当前轮子方向之差；

以前方向＝以前所选方向和新方向之差。

计算这些项使得离目标方向大的偏离，导致在“目标方向”项产生大的花费。费用函数 G 中的参数 a、b、c 可调整机器人的行为。

第七章：运动控制

什么是运动

移动过程是移动体与环境的物理交互过程.

关键问题有：

- 稳定性
- 接触的特征
- 环境类型

设计移动考虑因素

- 驱动器的规模
- 结构的复杂度
- 控制的成本
- 能效

足式移动机器人设计流程

- 腿的数量
- 稳定性
  - 静态稳定性　动态稳定性
- 腿的自由度
- 足行步态设计
- 动力学考虑

轮式移动机器人设计流程

1. 轮子结构

2. 稳定性

   3个轮子静态稳定性

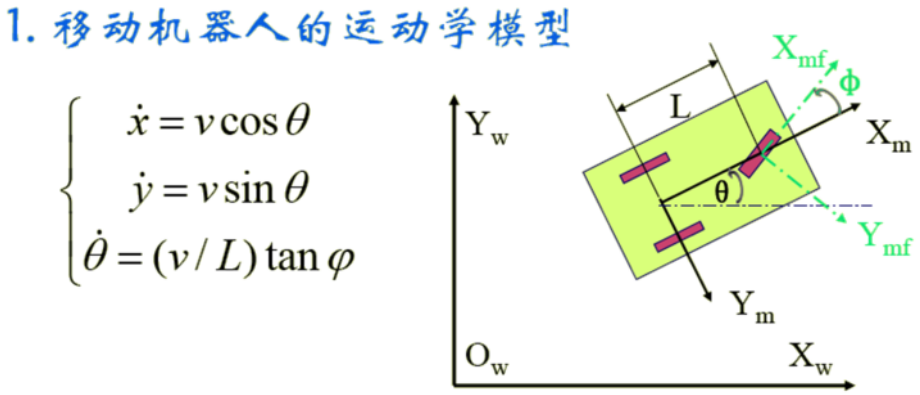
   更多轮子动态稳定性/悬挂

3. 轮子的数量与部署

   单轮，2轮，3轮，4轮.....

4. 悬挂系统设计
5. 驱动转向的关系

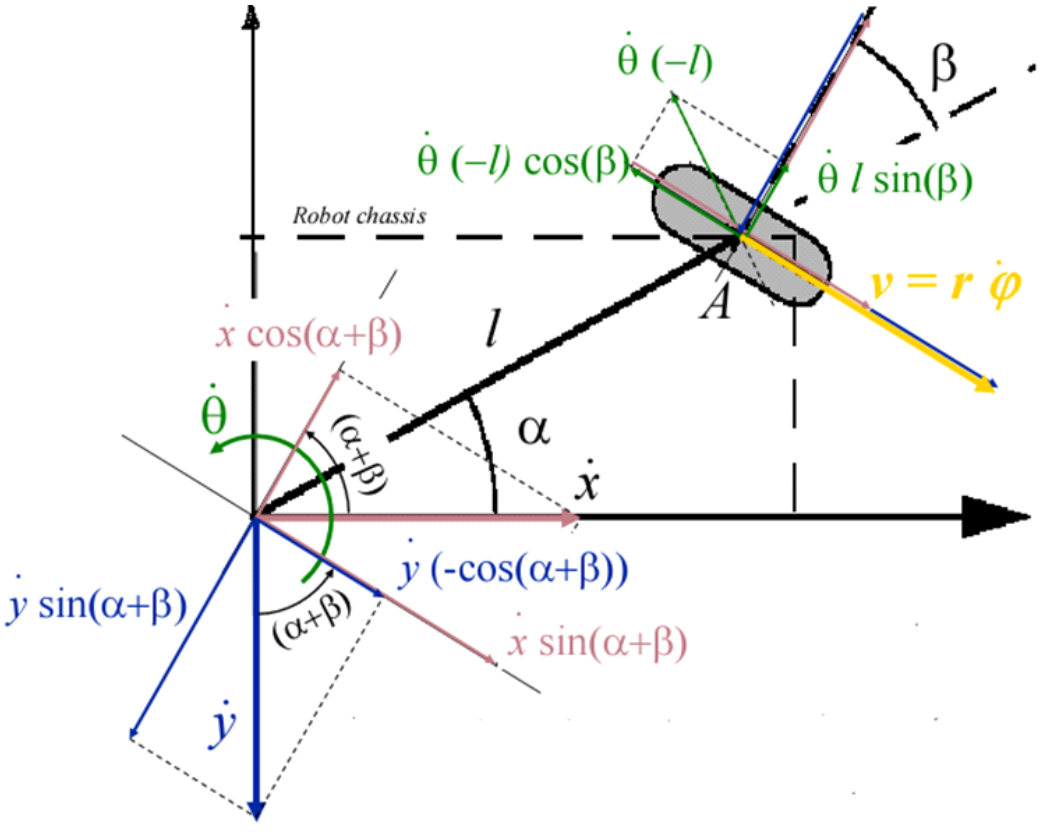
运动变换



$$\dot{\xi}_R = R(\theta)\dot{\xi}_I$$

R(θ)是旋转矩阵，PPT中的R(θ)上标符号有问题，考试中只需要写对旋转矩阵即可。

运动约束



$$\left[\sin(\alpha+\beta) \ -\cos(\alpha+\beta) \ (-l)\cos\beta\right] R(\theta)\dot{\xi}_I - r\dot{\varphi} = 0$$

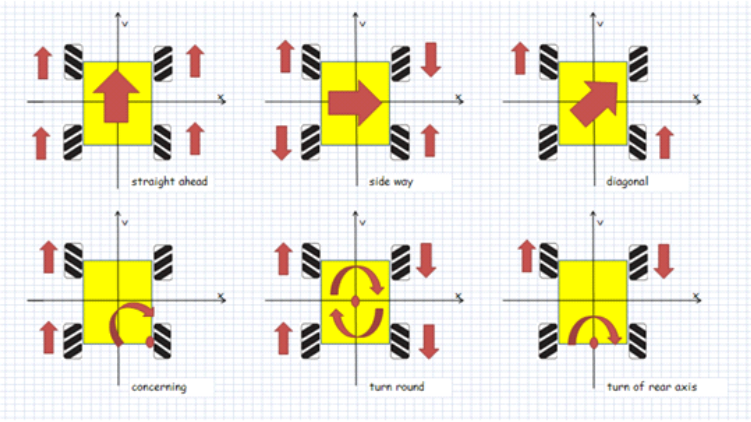
$$\left[\cos(\alpha+\beta) \ \sin(\alpha+\beta) \ l\sin\beta\right] R(\theta)\dot{\xi}_I = 0$$

第一个是滚动约束，第二个是滑动约束。

滚动方向上等于rφ，滑动方向上为0。要求会分析。

瑞典轮

一种很神奇的轮子，他有三个自由度，常见的是45°的瑞典轮，如下



四个轮子是呈X形对起来的。

问答题

建立全局参考坐标系的必要性

全局坐标系是三维空间物体所在的坐标系，模型的顶点坐标就是基于这个坐标系来表达的。它是我们考虑的最大范围，在机器人路径规划等地方，都需要用到全局参考坐标系。

建立局部参考坐标系的必要性

在机器人移动的时候，机器人可能会遇到各种障碍，由于可以从机器人的一些传感器（双目相机、超声波测距、红外传感器）获得信息，而这些信息都是相对于机器人的信息，所以有必要建立局部参考坐标系。

同步传动

所有的轮子转动被同一马达驱动：定义平台的移动速度

所有轮子的转向被同一马达驱动：设定平台的前进方向

机器人坐标系统的方位总是一样的：只能移动本体但不能转动本体，只有两个自由度

在旋转的时候，可以有阿克曼转向。沿着弯道转弯时，利用四连杆的相等曲柄使内侧轮的转向角比外侧轮大大约2~4度，使四个轮子路径的圆心大致上交会于后轴的延长线上瞬时转向中心，让车辆可以顺畅的转弯

差分传动

转弯行驶或在不平路面上行驶时，使左右车轮以不同转速滚动，即保证两侧驱动车轮作纯滚动运动。

无论 WMR 运动到何处，其左、右驱动轮间轮距 L 是不会改变的，因此左右轮的坐标与轮间距 L 的关系为

$$(x_L - x_R)^2 + (y_L - y_R)^2 = L^2$$

PID

概述

PID：比例，积分，微分。

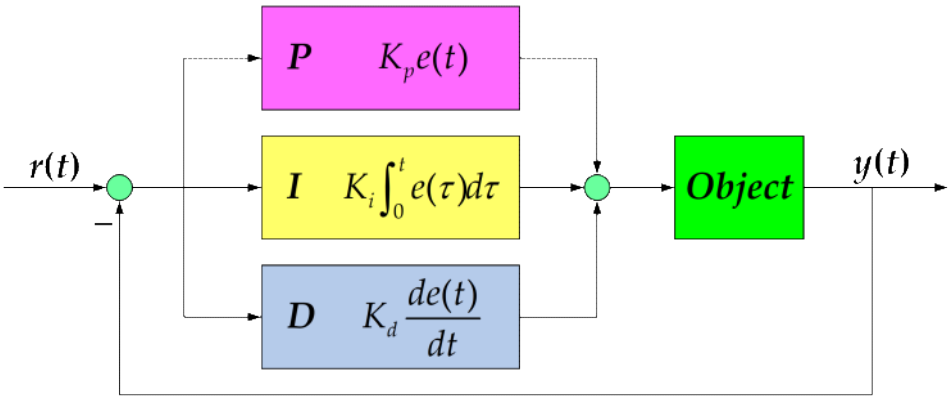
概念：系统偏差的比例(Proportional)、积分(Integral)和微分(Derivative)的综合控制，简称PID控制
特点：算法简单、鲁棒性强和可靠性高

PID控制器的输入一般是系统输出与一个参考值的差值，即控制偏差，然后把这个(基于时间序列的)差值用于计算新的控制量。

PID控制器主要适用于基本线性和动态特性不随时间变化的系统。

PID控制器的输出是由比例控制、积分控制和微分控制三项组成，三项在控制器中所起的控制作用相互独立。

连续



$$\mathrm{u}(t) = \mathrm{MV}(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{d}{dt} e(t)$$

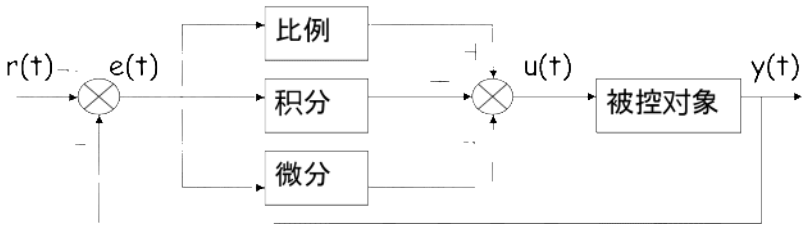
其中
K_p：比例增益，是调适参数
K_i：积分增益，也是调适参数
K_d：微分增益，也是调适参数
e：误差=设定值（SP）- 回授值（PV）
t：目前时间
τ：积分变数，数值从0到目前时间t

离散

在采样周期远小于信号变化周期时，可作如下近似：

$$\begin{cases} u(t) \approx u(k) \\ e(t) \approx e(k) \\ \int_0^t e(t)dt \approx \sum_{j=0}^{k} e(j)\Delta t = T\sum_{j=0}^{k} e(j) \\ \frac{de}{dt} \approx \frac{e(k)-e(k-1)}{\Delta t} = \frac{e(k)-e(k-1)}{T} \end{cases}$$

式中，T为**采样周期**；k为采样序号，则有



$$u(k) = K_p\{e(k) + \frac{T}{T_i}\sum_{j=0}^{k} e(j) + \frac{T_d}{T}[e(k)-e(k-1)]\} + u_0$$

其中，积分系数

$$K_i = K_p\frac{T}{T_i}$$

微分系数

$$K_d = K_p\frac{T_d}{T}$$

可以简写为

$$u(k) = K_p e(k) + K_i\sum_{j=0}^{k} e(j) + K_d[e(k)-e(k-1)] + u_0$$

位置式

对于公式

$$u(k) = K_p e(k) + K_i\sum_{j=0}^{k} e(j) + K_d[e(k)-e(k-1)] + u_0$$

由此得出的控制量为全量值输出，也就是每次的输出值都与执行机构的位置(如控制阀门的开度)一一对应，所以把它称之为位置式数字PID控制算法。

增量式

当控制系统中的执行器为步进电机、电动调节阀、多圈电位器等具有保持历史位置功能的装置时，需要的不是控制量的绝对数值，而是其增量值。因此，需要由数字PID位置式导出数字PID控制算法的增量式。

对数字PID位置式取增量，即数字控制器输出的是相邻两次采样时刻所计算的位置值之差：

$$\begin{aligned} \Delta u(k) &= u(k) - u(k-1) \\ &= K_p[e(k)-e(k-1)] + K_i e(k) + K_d[e(k)-2e(k-1)+e(k-2)] \end{aligned}$$

由于该式得出的是数字PID控制器输出**控制量的增量值**，因此，称之为增量式数字PID控制算法。它只需要保持三个采样时刻的偏差值。

增量式和位置 式控制算法比较

1. 增量式可以消除控制器的积分饱和：

   位置式算法每次输出均与过去的所有状态有关，计算时要对e(k)进行累加，容易造成积分饱和，计算机运算工作量也很大，而在增量式中由于消去了积分项，从而可消除控制器的积分饱和，在精度不足时，计算误差对控制量的影响较小，容易取得较好的控制效果（只存三个偏差值即可）。

2. 增量式易于实现从手动到自动的无扰动切换。

   为实现手动—自动无扰切换，在切换瞬时，必须首先将计算机的输出值设置为阀门原始开度 。由于增量式计算只与本次的偏差值有关，与阀门原来的位置无关，其输出对应于阀门位置的变化部分，因此，易于实现从手动到自动的无扰动切换。

3. 采用增量式算法时所用的执行器本身都具有保持功能，即使计算机发生故障，执行器仍能保持在原位，不会对生产造成恶劣影响。

4. 增量式控制算法的不足：积分截断效应大，有静态误差，溢出影响大等

编程举例

PID控制器程序：

```
//PID Algorithms
 static int r_old=0, e_old=0, e_old2=0;
…
e_func = v_des –v_act;                    //当前时刻的error
r_mot = r_old + Kp*(e_func – e_old) + Ki*(e_func+e_old)/2
+Kd*(e_func -2*e_old + e_old2);   // PID
r_mot = min(r_mot, MAX_output_Motor); //限定r_mot的范围
r_mot = max(r_mot, MIN_output_Motor); //限定r_mot的范围
// Move the slider window
r_old = r_mot;
e_old2 = e_old;
e_old = e_func;
```

PID，最后r_mot的结果是位置式的，有一个变化，计算的是Ki*(e_func+e_old)/2 ，如果单看r_mot-r_old的部分，是增量式的（废话）。

对于公式

$$u(k) = K_p e(k) + K_i \sum_{j=0}^{k} e(j) + K_d[e(k) - e(k-1)] + u_0$$

和

$$\Delta u(k) = u(k) - u(k-1)$$
$$= K_p[e(k) - e(k-1)] + K_i e(k) + K_d[e(k) - 2e(k-1) + e(k-2)]$$

作业

一：可佳机器人

见第一章

二：Cmake

构成libhello.so库

ADD_LIBRARY(hello SHARED ${LIBHELLO_SRC})

三：见复习