

第1季

第3章：OpenCV图像处理



主讲人：张虎

(小虎哥哥爱学习)

- 先导课
- 第1季：快速梳理知识要点与学习方法 ✓
- 第2季：详细推导数学公式与代码解析
- 第3季：代码实操以及真实机器人调试
- 答疑课

----- (永久免费 • 系列课程 • 长期更新) -----

本书内容安排

一、编程基础篇

第1章：ROS入门必备知识

第2章：C++编程范式

第3章：OpenCV图像处理

二、硬件基础篇

第4章：机器人传感器

第5章：机器人主机

第6章：机器人底盘

三、SLAM篇

第7章：SLAM中的数学基础

第8章：激光SLAM系统

第9章：视觉SLAM系统

第10章：其他SLAM系统

四、自主导航篇

第11章：自主导航中的数学基础

第12章：典型自主导航系统

第13章：机器人SLAM导航综合实战

机器人、图像处理和 OpenCV 是什么关系？



机器人

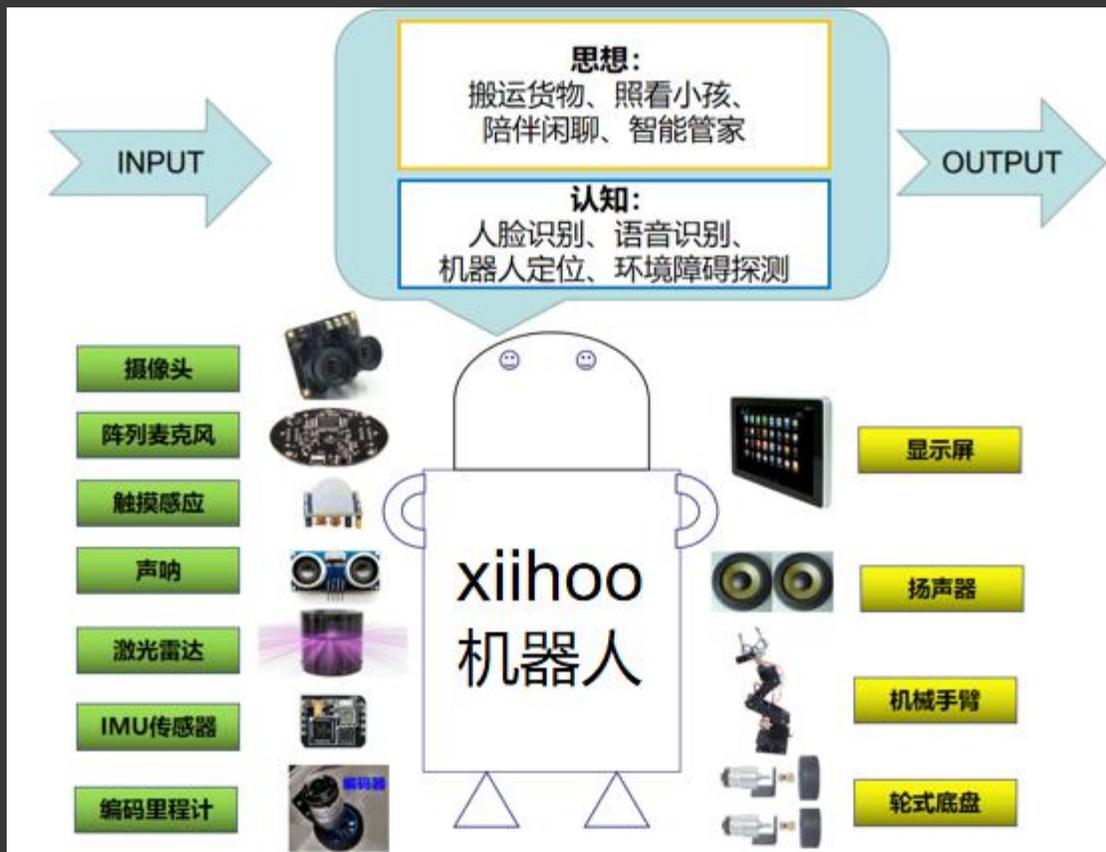
图像
处理



OpenCV

```
1 #include <opencv2/opencv.hpp>
2
3 int main(int argc, char** argv)
4 {
5     cv::Mat img=cv::imread("1.jpg");
6     cv::imshow("[img1]",img);
7     cv::waitKey(0);
8
9     return 0;
10 }
```

① 机器人，为什么需要图像处理技术？



眼 ↔ 摄像头 ↔ 图像

耳 ↔ 麦克风

口 ↔ 扬声器

鼻 ↔ 气体传感器

手 ↔ 机械手臂

脚 ↔ 驱动轮

...

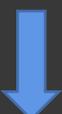
- ✓ 感知
- ✓ 识别
- ✓ 定位
- ✓ 建图
- ✓ 避障
- ✓ ...

① 机器人，为什么需要图像处理技术？

机器人

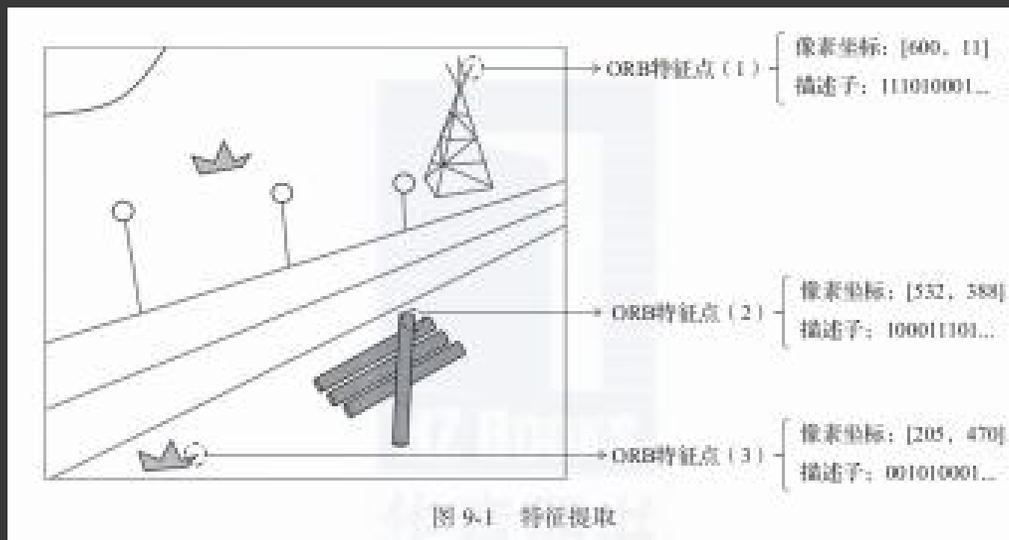


摄像头



图像

- ✓ 感知
- ✓ 识别
- ✓ 定位
- ✓ 建图
- ✓ 避障
- ✓ ...



① 机器人，为什么需要图像处理技术？

机器人

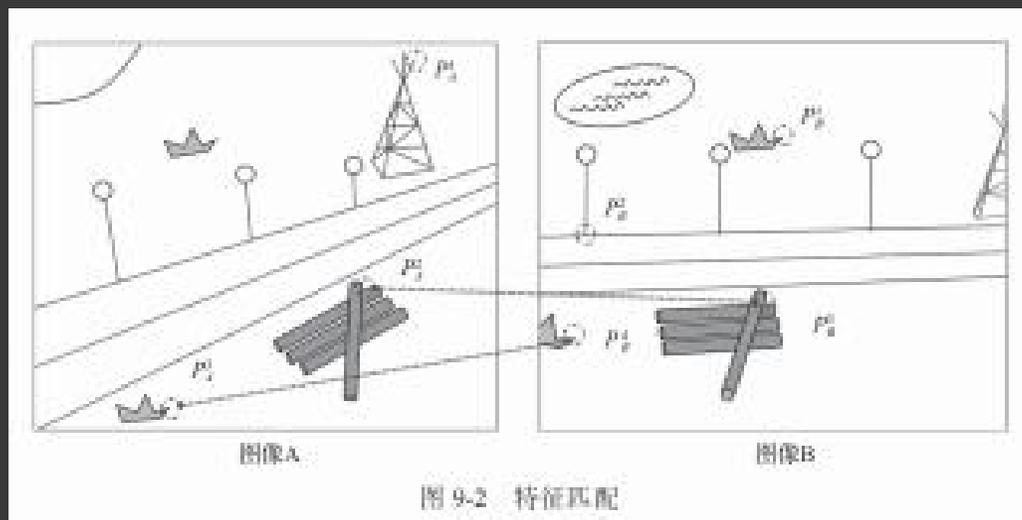
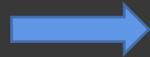


摄像头



图像

- ✓ 感知
- ✓ 识别
- ✓ 定位
- ✓ 建图
- ✓ 避障
- ✓ ...



① 机器人，为什么需要图像处理技术？

机器人

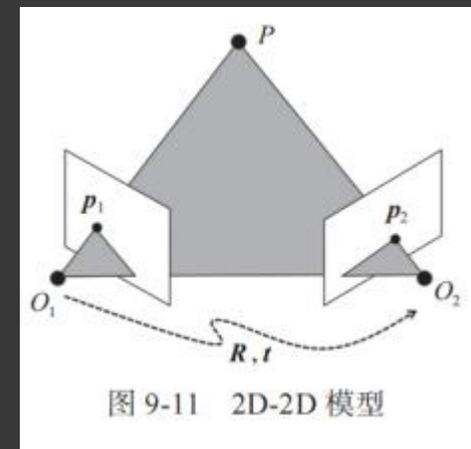
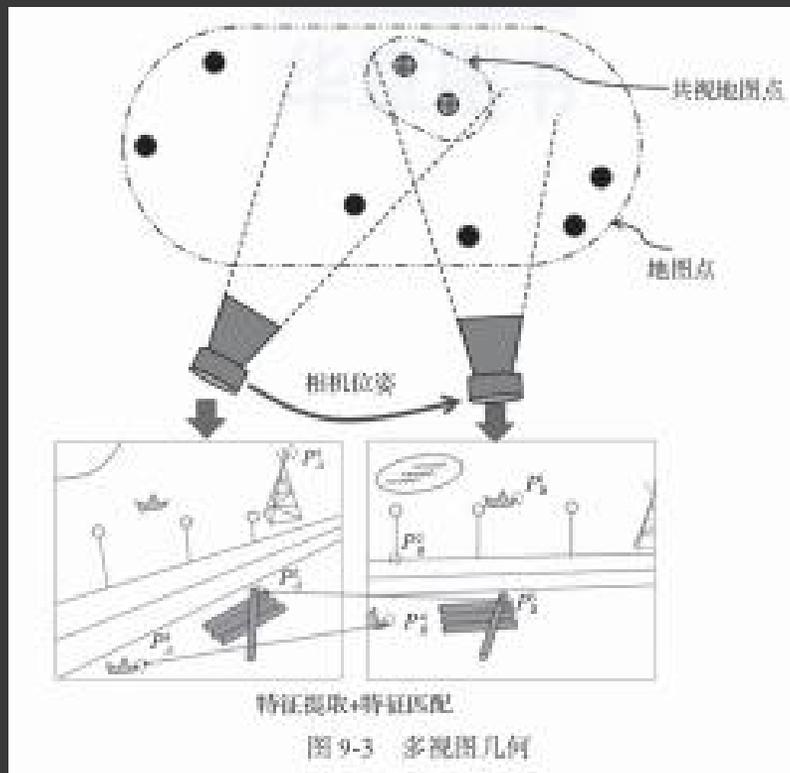


摄像头



图像

- ✓ 感知
- ✓ 识别
- ✓ 定位
- ✓ 建图
- ✓ 避障
- ✓ ...



① 机器人，为什么需要图像处理技术？

机器人

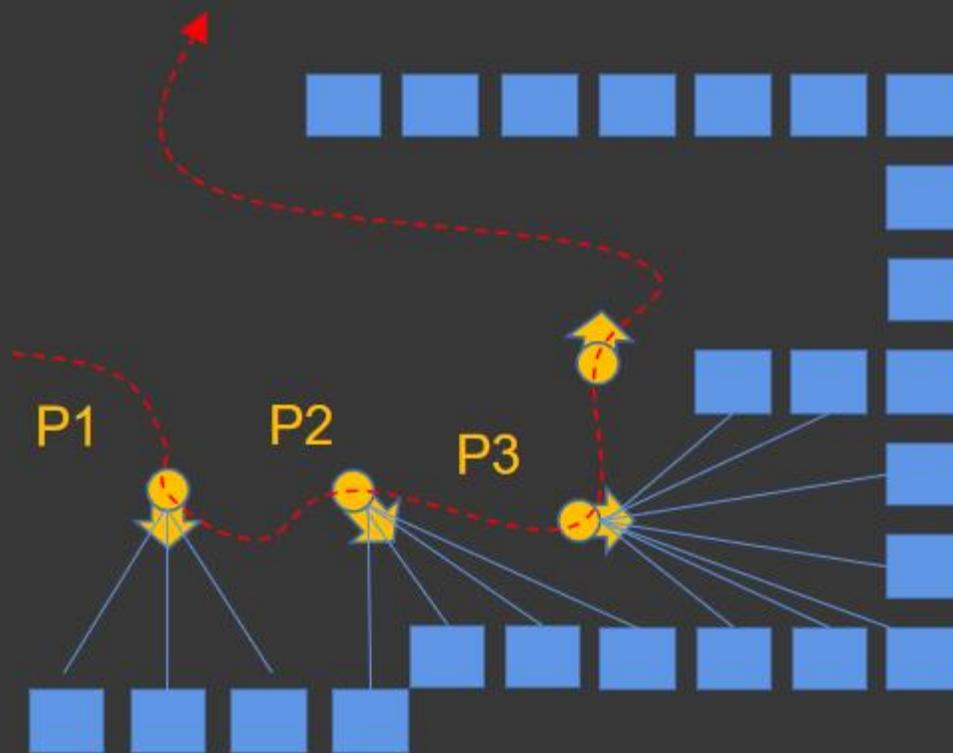


摄像头



图像

- ✓ 感知
- ✓ 识别
- ✓ 定位
- ✓ 建图
- ✓ 避障
- ✓ ...

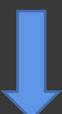


① 机器人，为什么需要图像处理技术？

机器人

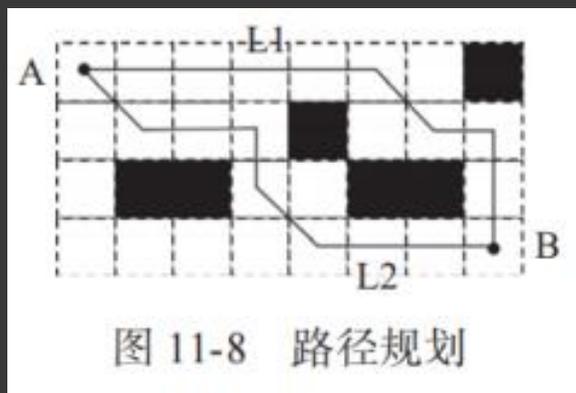
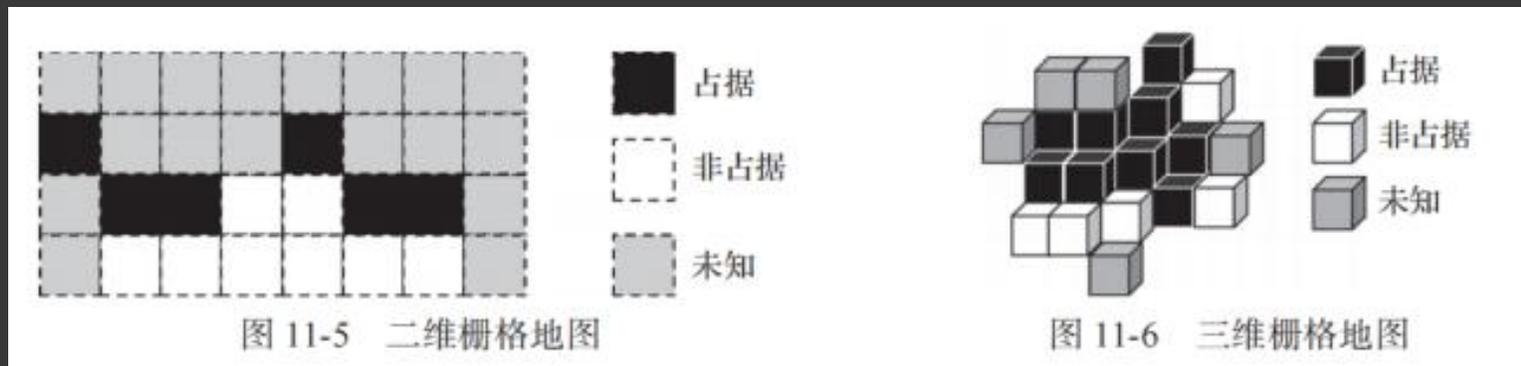


摄像头



图像

- ✓ 感知
- ✓ 识别
- ✓ 定位
- ✓ 建图
- ✓ 避障
- ✓ ...

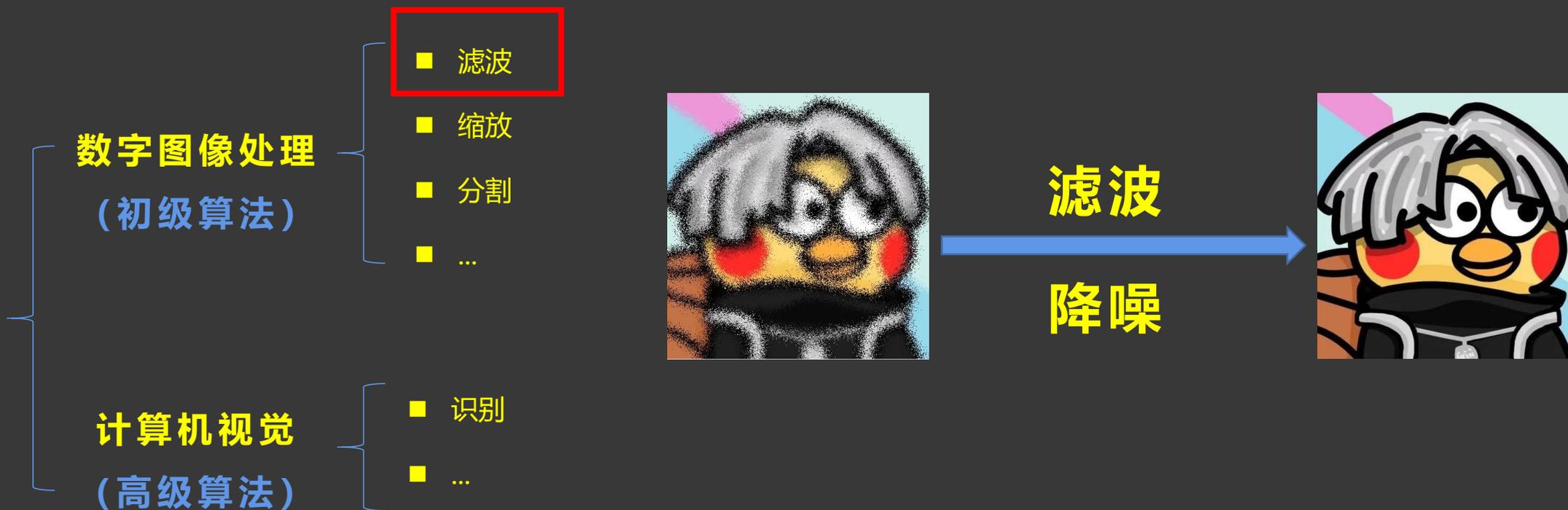


② 图像处理技术究竟是什么？

图像处理是利用计算机对图像进行计算分析的技术，包括数字图像处理和计算机视觉两大领域



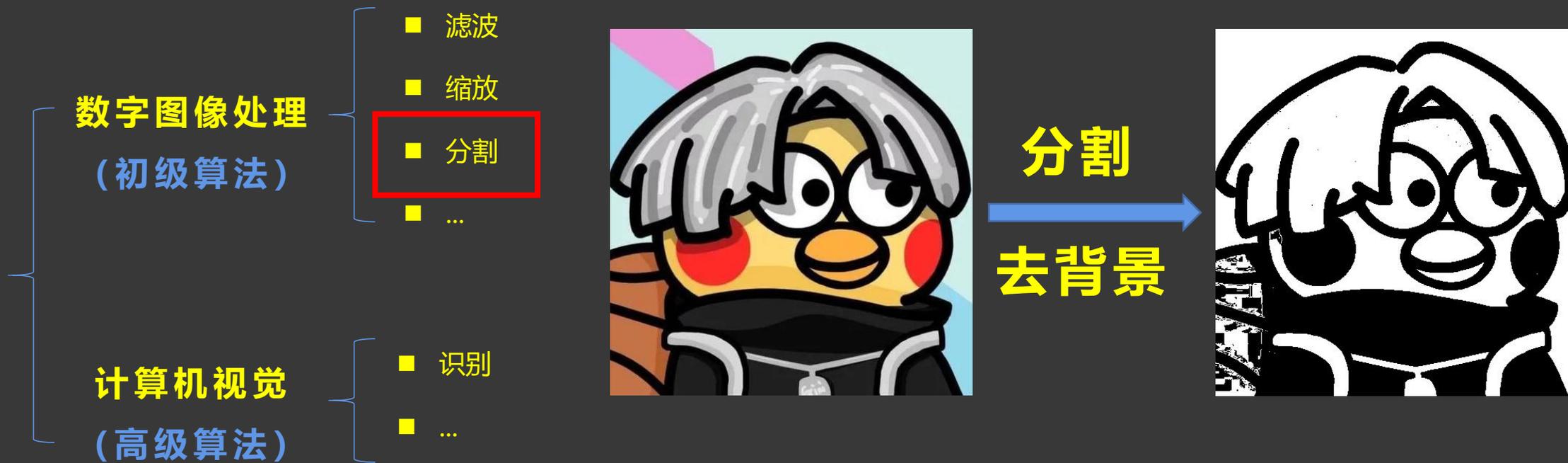
② 图像处理技术究竟是什么？



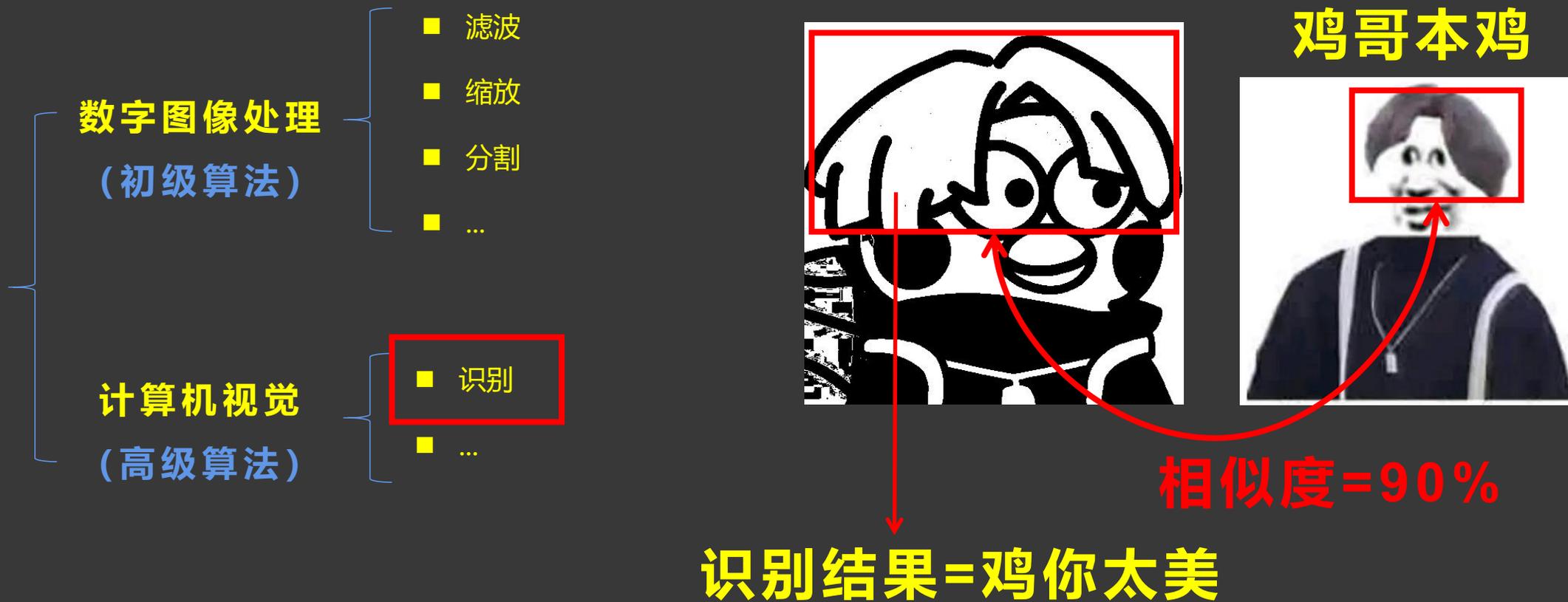
② 图像处理技术究竟是什么？



② 图像处理技术究竟是什么？



② 图像处理技术究竟是什么？



③ 图像处理技术与OpenCV是什么关系？

常见图像处理库

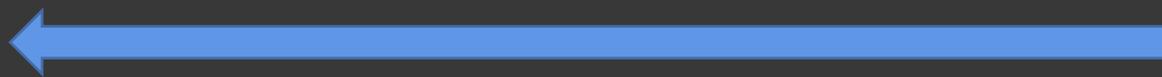
- OpenCV
- Halcon
- Matlab
- PIL
- skimage
- ...

AI图像处理平台

- TensorFlow
- PyTorch
- Caffe
- ...

图像处理办公软件

- PhotoShop
- 美图秀秀
- iSee
- ...



③ 图像处理技术与OpenCV是什么关系?

Windows

Linux

Mac

Android

iOS

...

跨平台

C++调用接口

Python调用接口

Java调用接口

Matlab调用接口

...

调用接口

核心功能模块

图像处理模块

标定模块

最近邻搜索

视频组件

OpenCV库

GPU加速

GUI交互

机器学习模块

目标检测模块

...

(C++语言编写)

内容概要

3.1 认识图像数据

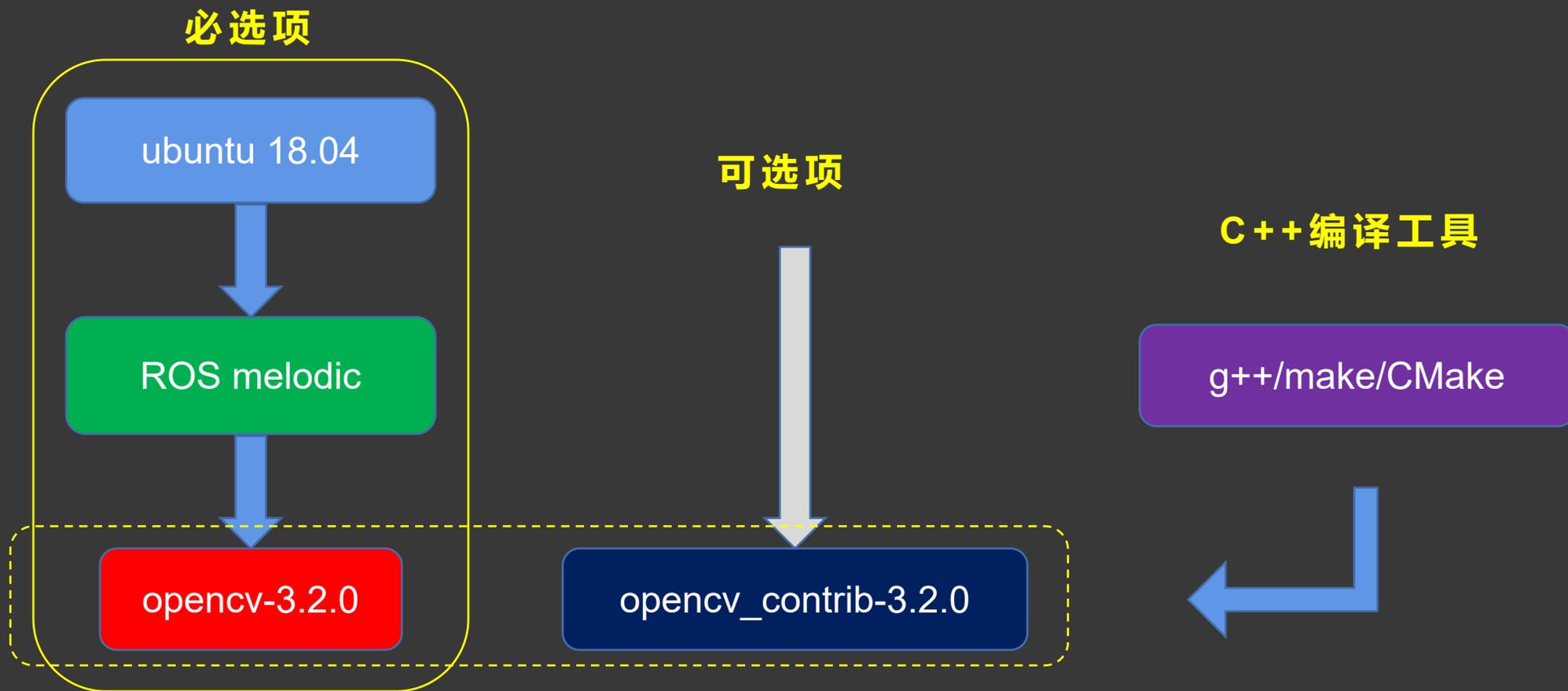
3.2 图像滤波

3.3 图像变换

3.4 图像特征点提取

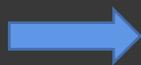
3.1 认识图像数据

开发环境搭建



3.1 认识图像数据

- 获取图像数据



- 访问图像数据

```
cv::Mat img=cv::imread("1.jpg");
```

```
cv::VideoCapture cap("1.mp4");
```

```
cv::Mat frame;
```

```
cap>>frame;
```

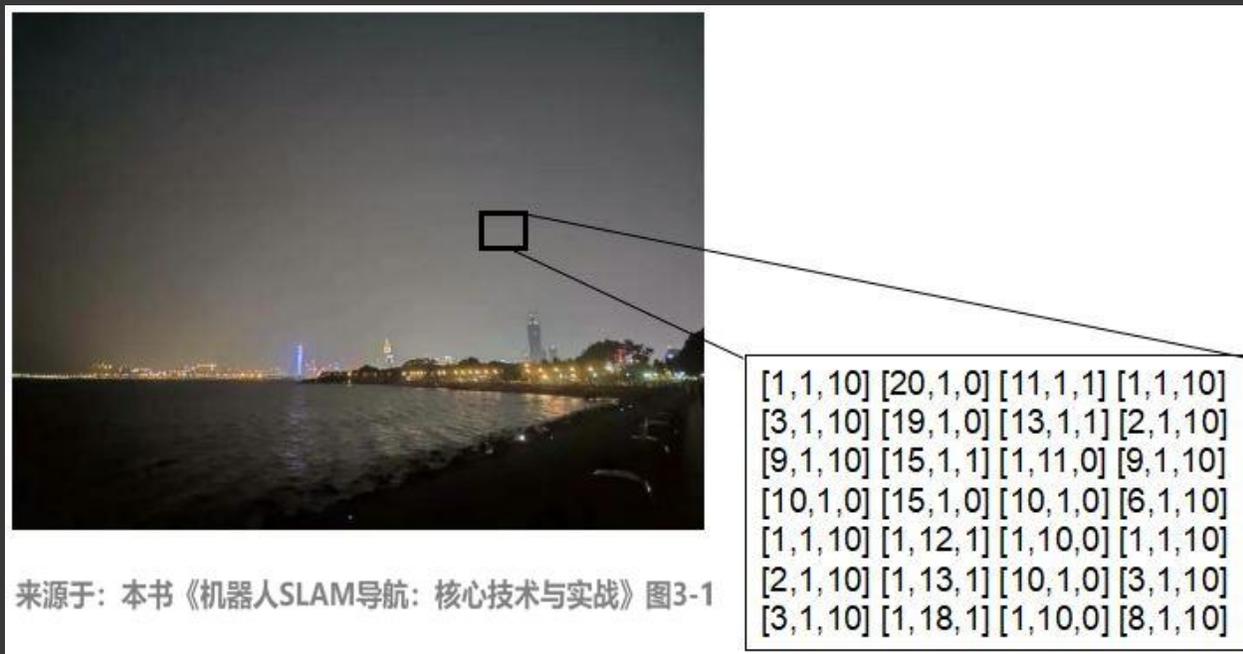
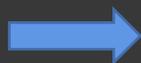
```
cv::VideoCapture cap(0);
```

```
cv::Mat frame;
```

```
cap>>frame;
```

3.1 认识图像数据

- 获取图像数据
- 访问图像数据



```
cv::Mat img(2,2,CV_8UC3,cv::Scalar(1,100,255));
```

cv::Point类、cv::Scalar类、cv::Size类、cv::Rect类

颜色空间转换函数cv::cvtColor(): RGB、HSV、HSI

像素遍历Mat->at<cv::Vec3b>(i,j)[channel]

通道分离cv::split()和cv::merge

内容概要

3.1 认识图像数据

3.2 图像滤波

3.3 图像变换

3.4 图像特征点提取

3.2 图像滤波

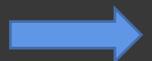
图像是由一个一个像素点组成的，处理图像就是处理这些像素点。

图像像素之间的关联性是重要的信息，不能完全把像素点割裂开来，这一点也正是众多图像算法的出发点。这里就通过图像滤波，来帮助大家具体理解像素之间的这种关联性。

图像滤波的目的是在尽量保留图像特征的前提下，过滤掉图像中的噪声，其滤波效果直接影响到后续图像识别、分析等算法的效果。

3.2 图像滤波

- 线性滤波
- 非线性滤波
- 形态学滤波



一维线性滤波
二维线性滤波

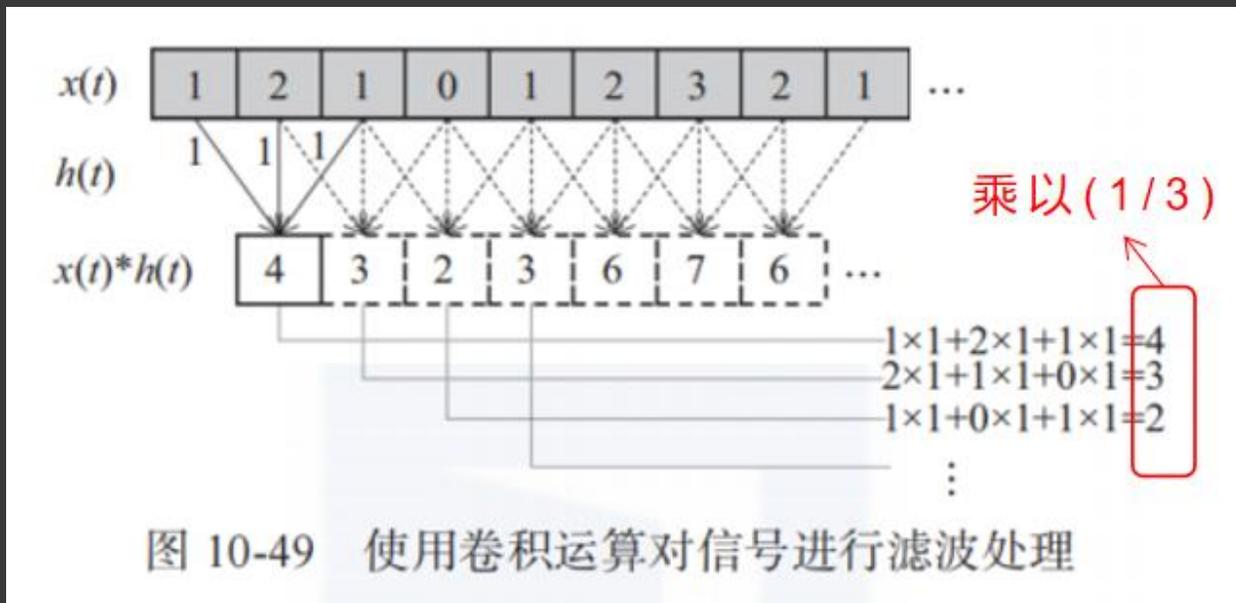
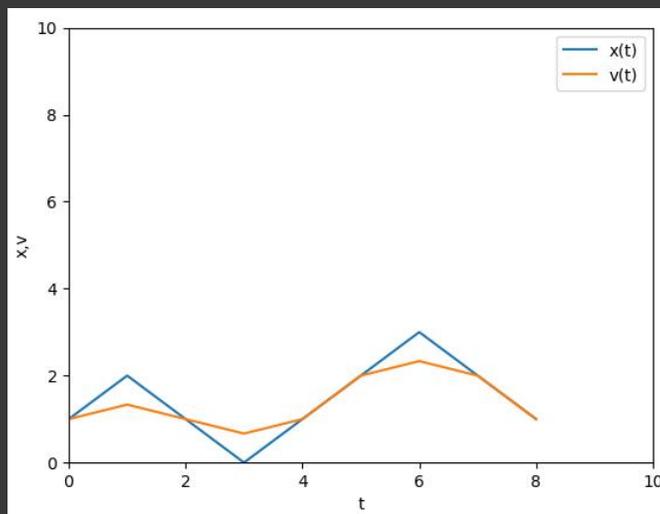
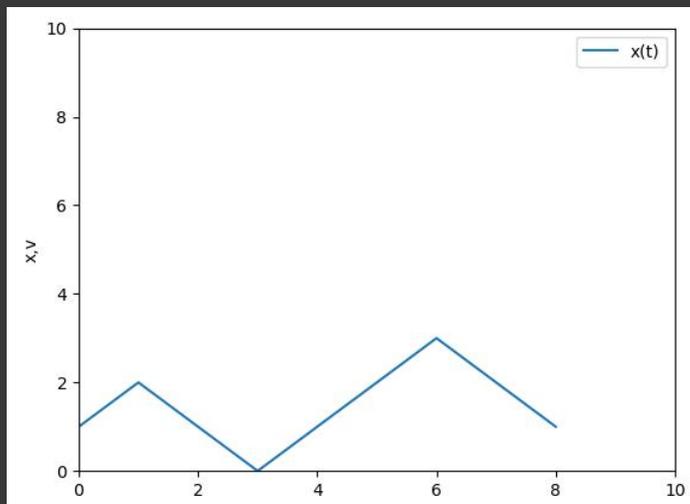


图 10-49 使用卷积运算对信号进行滤波处理

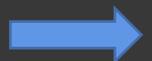


3.2 图像滤波

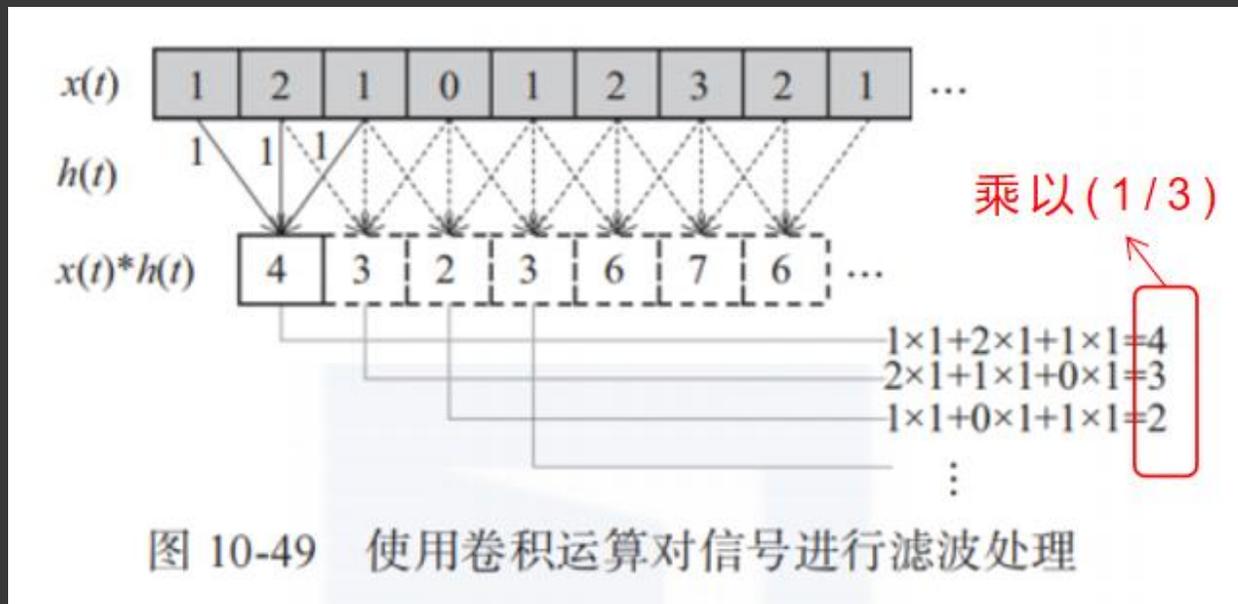
■ 线性滤波

■ 非线性滤波

■ 形态学滤波



一维线性滤波
二维线性滤波



① 一维、二维、...

② 线性、非线性

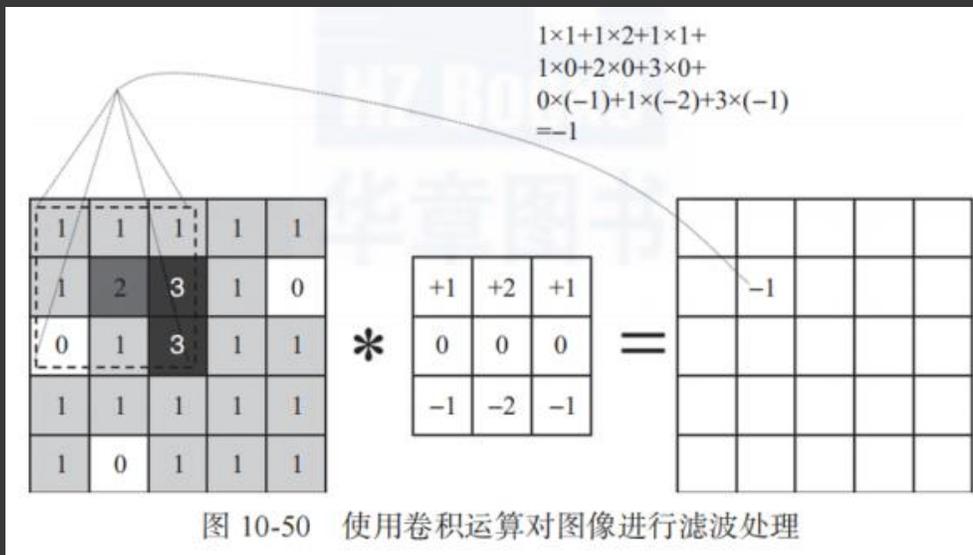
③ 卷积运算

卷积核尺寸

卷积核系数

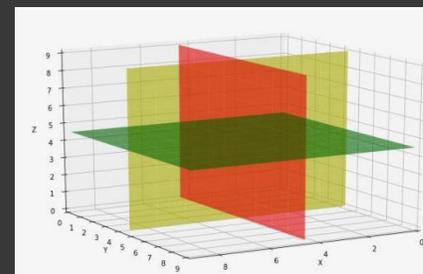
3.2 图像滤波

- 线性滤波
 - 非线性滤波
 - 形态学滤波
- } 一维线性滤波
} 二维线性滤波



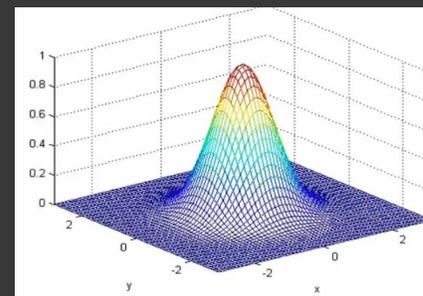
均值滤波

$$h(x, y) = \frac{1}{width \cdot height} \begin{bmatrix} 1 & 1 & \dots & 1 \\ \dots & \dots & \dots & \dots \\ 1 & 1 & \dots & 1 \end{bmatrix}$$



高斯滤波

$$h(x, y) = A \cdot \exp\left(\frac{-(x-u_x)^2}{2\sigma_x^2} + \frac{-(y-u_y)^2}{2\sigma_y^2}\right)$$



3.2 图像滤波

■ 线性滤波

■ 非线性滤波



中值滤波：排序法

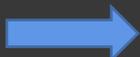
■ 形态学滤波

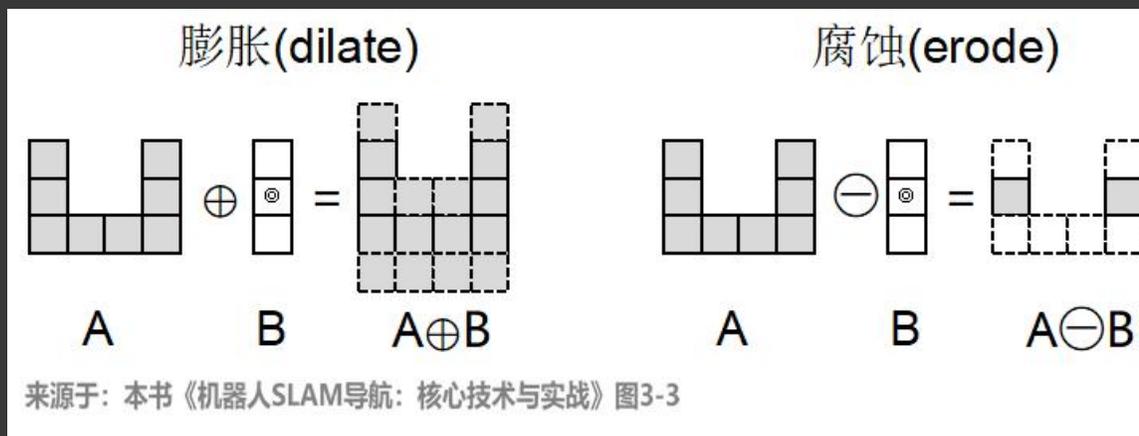
双边滤波：

$$\begin{aligned} h(x, y, m, n) &= q(x, y, m, n) \cdot r(x, y, m, n) \\ &= \exp\left(-\frac{(x-m)^2 + (y-n)^2}{2\sigma_q^2}\right) \cdot \exp\left(-\frac{\|f(x, y) - f(m, n)\|^2}{2\sigma_r^2}\right) \end{aligned}$$

$$g(x, y) = \frac{\sum_{m, n} f(m, n) h(x, y, m, n)}{\sum_{m, n} h(x, y, m, n)}$$

3.2 图像滤波

- 线性滤波
- 非线性滤波
- 形态学滤波 



膨胀 $dilate(A, B) = \max_B(A)$

腐蚀 $erode(A, B) = \min_B(A)$

开运算 $open(A, B) = dilate(erode(A, B))$

闭运算 $close(A, B) = erode(dilate(A, B))$

形态学梯度 $morphgrad(A, B) = dilate(A, B) - erode(A, B)$

顶帽运算 $tophat(A, B) = A - open(A, B)$

黑帽运算 $blackhat(A, B) = close(A, B) - A$

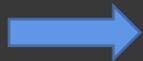
3.2 图像滤波

■ 线性滤波



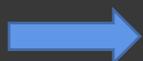
`cv::GaussianBlur()`

■ 非线性滤波



`cv::medianBlur()`
`cv::bilateralFilter()`

■ 形态学滤波



`cv::morphologyEx()`

内容概要

3.1 认识图像数据

3.2 图像滤波

3.3 图像变换

3.4 图像特征点提取

3.3 图像变换

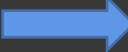
经过3.2节图像滤波的学习，相信大家对图像处理有了一定的了解。不过，图像滤波只是很初级的处理，其目的是提升图像本身的质量。

本节要讲到的图像变换，从改变图像的结构入手，将图像变换成不同的形态。

由于篇幅限制，这里重点讨论在后续视觉SLAM章节中涉及到的一些图像变换算法。其他一些常用图像变换算法将略过，比如频谱变换、小波变换、图像金字塔等，感兴趣可以查阅相关资料。

3.3 图像变换

重映射:

■ 射影变换  就是把原图中某个位置的像素放到另一个位置, 比如把图像水平翻转

■ 霍夫变换

■ 边缘检测

■ 直方图均衡

$$g(x, y) = f(h(x, y))$$

图-1

$g(x, y)$

$g(x, y)$

$g(x, y)$

$g(x, y)$

图-2

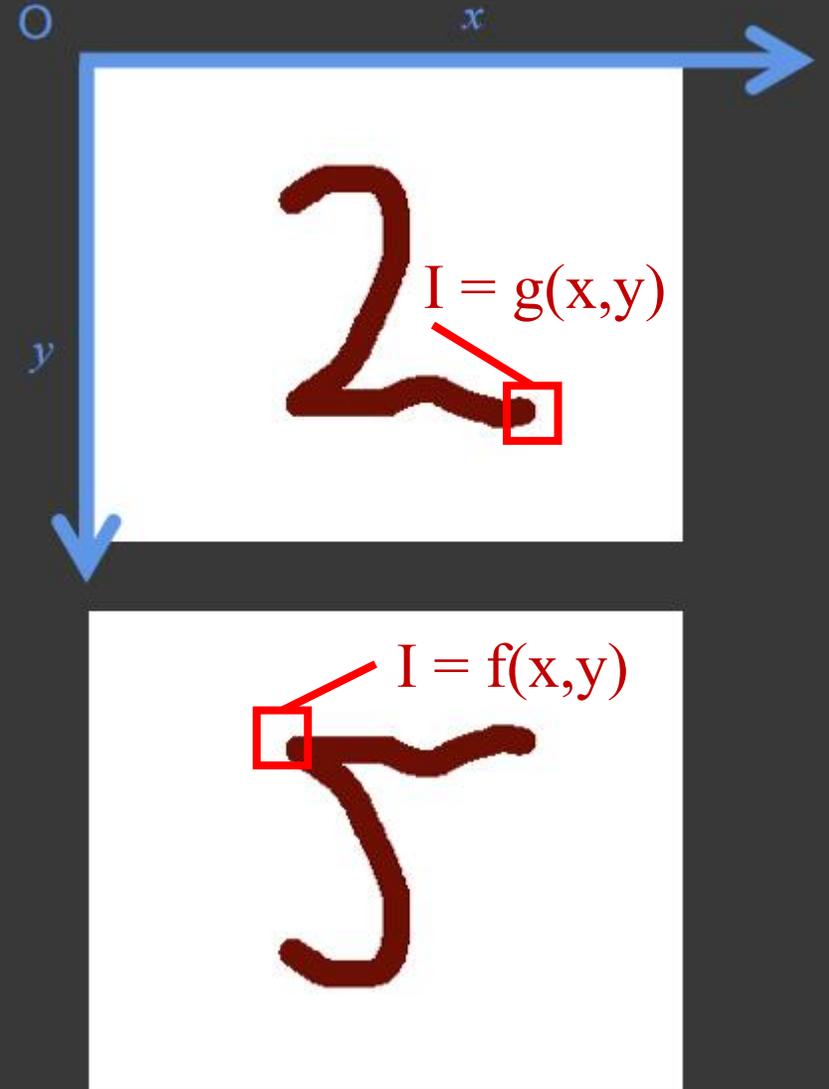
$f(x, y)$

$= f(x', y')$

$= f(x, High-y)$

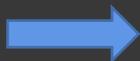
设 $(x', y') = h(x, y) = (x, High-y)$

$= f(h(x, y))$



3.3 图像变换

■ 射影变换



■ 霍夫变换

■ 边缘检测

■ 直方图均衡

欧式变换

$$h(x, y) = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

相似变换

$$h(x, y) = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s \cdot r_{11} & s \cdot r_{12} & t_x \\ s \cdot r_{21} & s \cdot r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

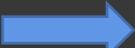
仿射变换

$$h(x, y) = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

射影变换

$$h(x, y) = \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

3.3 图像变换

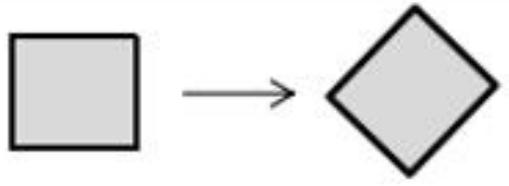
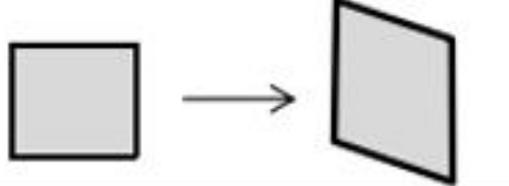
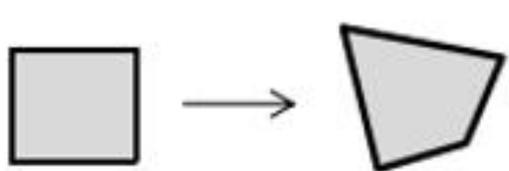
- 射影变换 

- 霍夫变换

- 边缘检测

- 直方图均衡

来源于：本书《机器人SLAM导航：核心技术与实战》表3-1

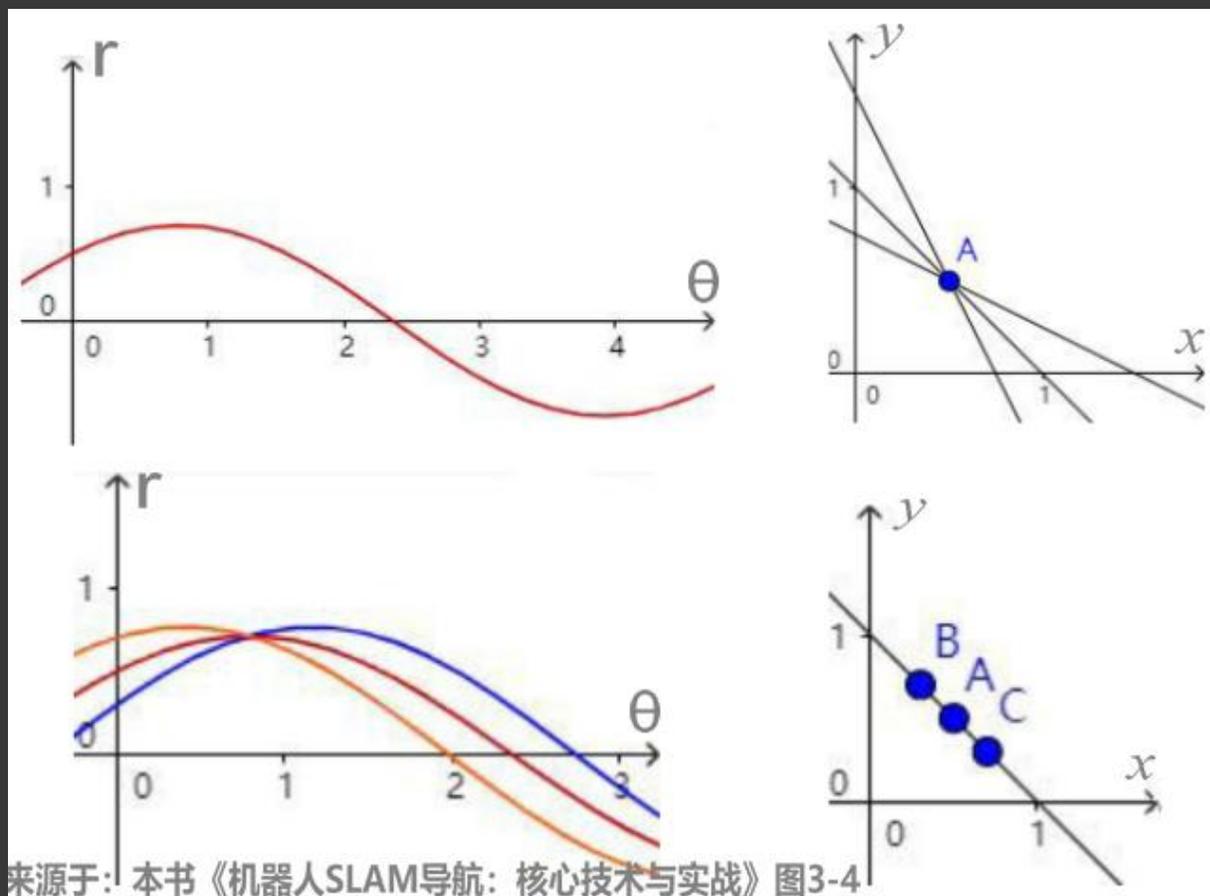
重映射方式	变换矩阵	效果举例
欧式变换 3 dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	
相似变换 4 dof	$\begin{bmatrix} s \cdot r_{11} & s \cdot r_{12} & t_x \\ s \cdot r_{21} & s \cdot r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	
仿射变换 6 dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	
射影变换 8 dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$	

3.3 图像变换

- 射影变换
- **霍夫变换** →
- 边缘检测
- 直方图均衡

直角坐标系中，距离原点 r 且与 x 轴夹角为 θ 的直线方程：

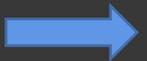
$$r = x \cdot \cos \theta + y \cdot \sin \theta$$



来源于：本书《机器人SLAM导航：核心技术与实战》图3-4

3.3 图像变换

- 射影变换
- 霍夫变换
- **边缘检测**
- 直方图均衡



sobel算法

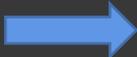
$$grad_x = I * \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

$$grad_y = I * \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

$$grad = \sqrt{grad_x^2 + grad_y^2}$$

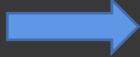
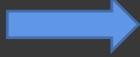
canny算法

3.3 图像变换

- 射影变换
- 霍夫变换
- 边缘检测
- 直方图均衡 



3.3 图像变换

- 射影变换  `cv::warpPerspective()`
- 霍夫变换  `cv::HoughLines()`
`cv::HoughLinesP()`
- 边缘检测  `cv::Sobel()`
`cv::Canny()`
- 直方图均衡  `cv::calcHist()`
`cv::equalizeHist()`

内容概要

3.1 认识图像数据

3.2 图像滤波

3.3 图像变换

3.4 图像特征点提取

3.4 图像特征点提取

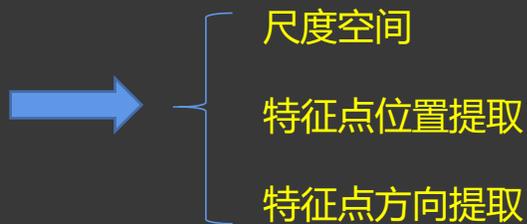
特征点提取算法能帮助计算机获取**图像的区域特征信息**，并应用于图像识别、图像匹配、三维重建、物体跟踪等领域。在实际工程中，具有很高的应用价值。

在图像领域，特征点 (feature points) 也常常被称为**关键点 (key points)** 或**兴趣点 (interest points)**。

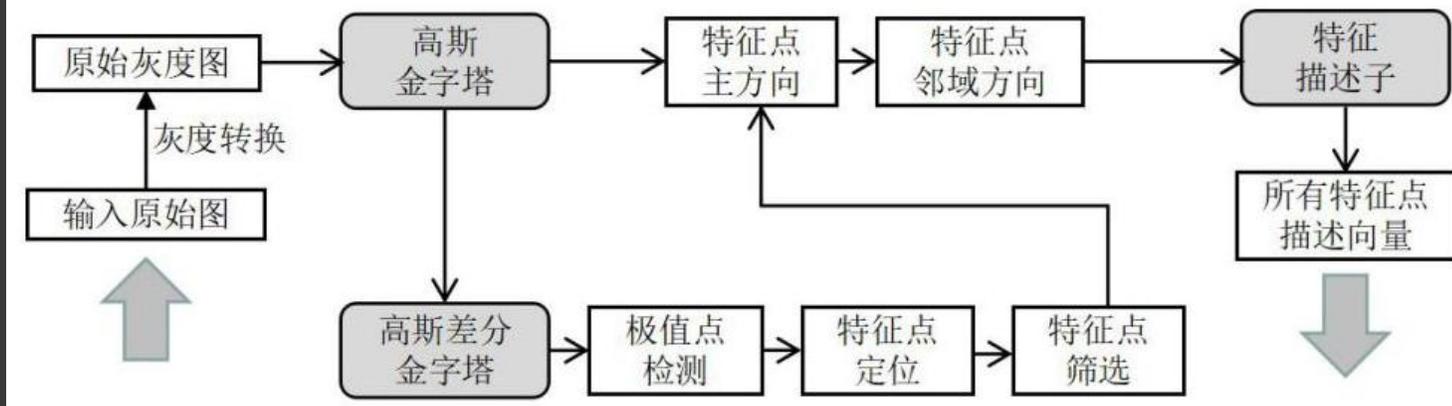
特征点的提取有多种算法，可以从图像纹理信息来提取，也可以通过图像区域灰度统计信息来提取，或者通过频谱变化、小波变换等变换后的特殊空间进行提取。

3.4 图像特征点提取

- SIFT特征点
- SURF特征点
- ORB特征点



来源于：书《机器人SLAM导航：核心技术与实战》图3-7



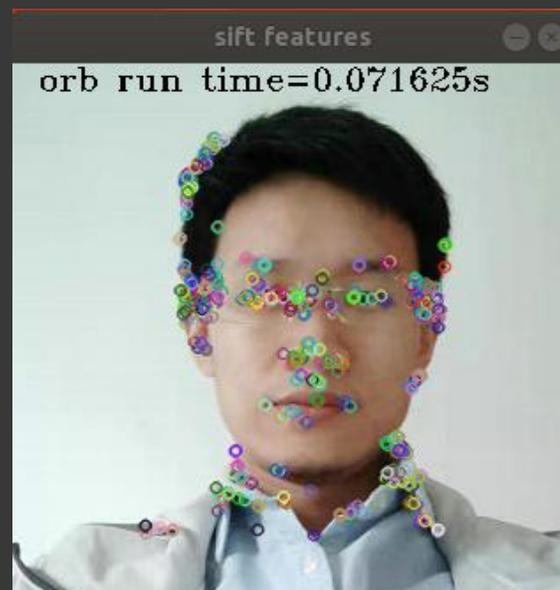
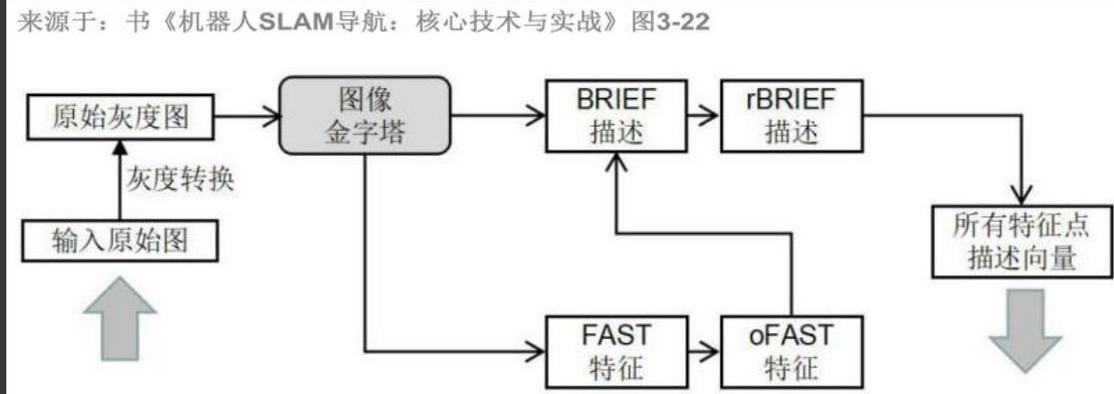
3.4 图像特征点提取

- SIFT特征点
 - SURF特征点
 - ORB特征点
-
- 尺度空间 (利用盒式滤波近似计算Hessian矩阵)
 - 特征点位置提取
 - 特征点方向提取 (统计扇形区域小波特征点)



3.4 图像特征点提取

- SIFT特征点
 - SURF特征点
 - ORB特征点
- 尺度空间 (不同尺度图像直接拼接成一张大图)
 特征点位置提取 (提取oFAST)
 特征点方向提取 (领域中心到质心的方向)



3.4 图像特征点提取

- SIFT特征点
- SURF特征点
- ORB特征点

本节的重点放在 SIFT、SURF 和 ORB 这 3 种特征点提取的原理讲解中，如果对 OpenCV 中具体实现程序感兴趣，可以阅读 OpenCV3 中相应的源代码，源代码的路径如下。

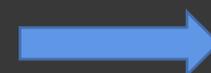
- ❑ SIFT 源码路径：`opencv_contrib/modules/xfeatures2d/src/sift.cpp`。
- ❑ SURF 源码路径：`opencv_contrib/modules/xfeatures2d/src/surf.cpp`。
- ❑ ORB 源码路径：`opencv/modules/features2d/src/orb.cpp`。

拓展

机器人



单目相机



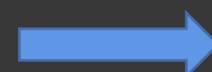
图像 (2D)

双目相机



RGB-D相机

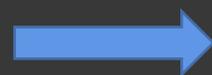
激光雷达 (单线、多线)



点云 (3D)

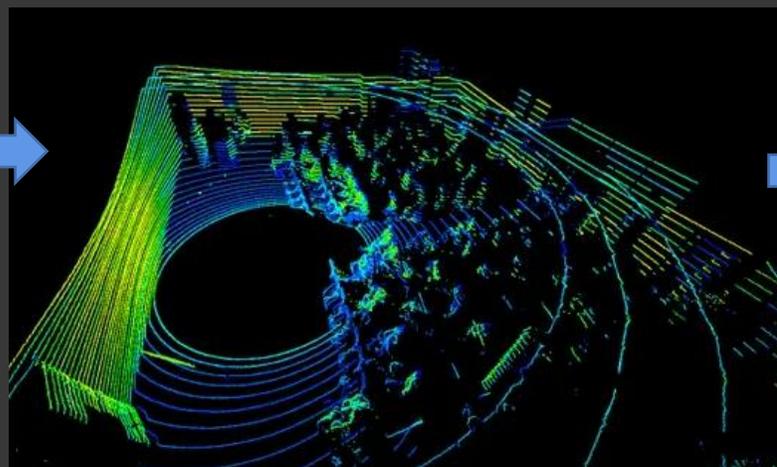
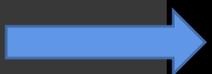
拓展

图像 (2D)



OpenCV
图像处理库

点云 (3D)



PCL
点云处理库

- 例程源码下载：https://github.com/xiihoo/Books_Robot_SLAM_Navigation
- 课件PPT下载：www.xiihoo.com

敬请关注,长期更新...

下集预告