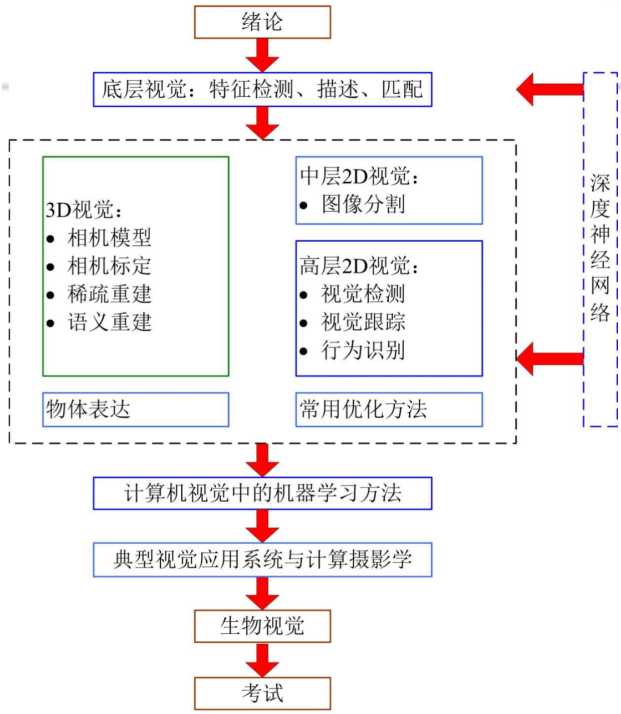
计算机视觉

国科大2023年春，授课老师：董秋雷，高伟，申抒含。整理者：蔡怀广

计算机视觉的研究目的是使计算机具备能从图像认知环境信息的能力。（感知：图像到信息）



- 计算机视觉
  - 底层视觉：特征检测
    - 边缘检测
      - 微分滤波器提取边缘
      - Canny边缘检测器
    - 特征点检测
      - Harris角点检测算法
      - FAST
  - 底层视觉：特征描述和匹配
    - SIFT：特征点检测、描述与匹配
      - 1.SIFT特征的特点
      - 2.SIFT的流程
      - 3.SIFT的原理
    - 特征匹配
      - Cross-correlation
    - ORB（Oriented FAST and Rotated BRIEF）
    - 基于CNN的特征点与描述子学习
    - 空间点匹配：ICP(Iterative Closest Point)
    - 鲁棒匹配的RANSAC框架
  - 3D视觉
  - 3D视觉：相机模型
    - 成像模型发展脉络
    - 相机模型
      - 欧式空间和射影空间
      - 相机模型的数学表达

        - 相机模型的内参数矩阵
        - 两视图几何
- 3D视觉：稀疏重建
    - 三角化（Triangulation）
    - 相机标定（Camera Calibration）
        - 1. 使用三维标定物
        - 2. 使用平面标定板（最常用的方法，2000）
        - 3. 利用消影点标定（需要假设K中只有f未知）
        - 总结
    - 姿态估计（Pose Estimation）
    - 稀疏重建（Sparse Reconstruction）
    - 重投影误差最小化问题的求解
- 3D视觉：三维建模
    - 立体视觉（Stereo）
    - 视差（Disparity）与深度（Depth）
    - 多视图立体重建 （Multiple View Stereo,MVS）
- 3D视觉：三维表达与语义重建
    - 点云网格建模：获取封闭完整的三角网格模型
    - 三维语义建模：获取三维点/三角面片的语义类别属性
    - 三维矢量建模：获取小体积、紧致、规范的矢量化表达
- 中层2D视觉：图像分割
    - 早期的图像分割方法
    - 基于特定理论的方法
        - 聚类
        - Mean-Shift
        - Graph Cut
    - 深度神经网络完成图像任务
        - Semantic Segmentation(语义分割)
        - Object Detection(目标检测)
        - Instance Segmentation(实例分割)
- 高层2D视觉：检测
    - 运动检测
        - 背景差法（background subtraction）：混合高斯模型
        - 光流（optical flow）
        - 帧间差分（frame differencing）
    - 目标检测
        - AdaBoost
        - 基于DNN的物体检测
- 高层2D视觉：跟踪
    - 目标跟踪
        - 运动目标的表示方法
        - 传统目标跟踪方法
        - 基于相关滤波的跟踪方法（MOSSE,2010）
        - 基于DNN的跟踪方法
    - 视觉定位
- 高层2D视觉：行为识别
    - 运动表达
        - 运动表达——运动轨迹
        - 运动表达——时空图表达
    - 行为识别

底层视觉：特征检测

边缘检测

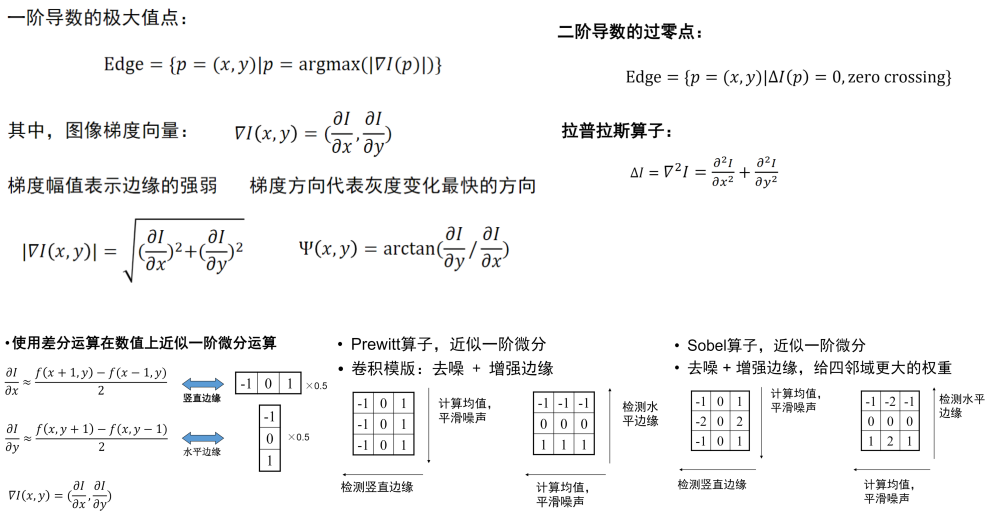
图像边缘的产生？物体的边界、表面方向的改变、不同的颜色、光照明暗的变化。边缘定义？图像灰度构成的曲面上的陡峭区域。

为什么要提取边缘？可以代表物体；边缘特征对于图像的变化不敏感（几何，灰度，光照方向）；可以用于物体检测。

提取边缘的方法（一般针对灰度图象）：

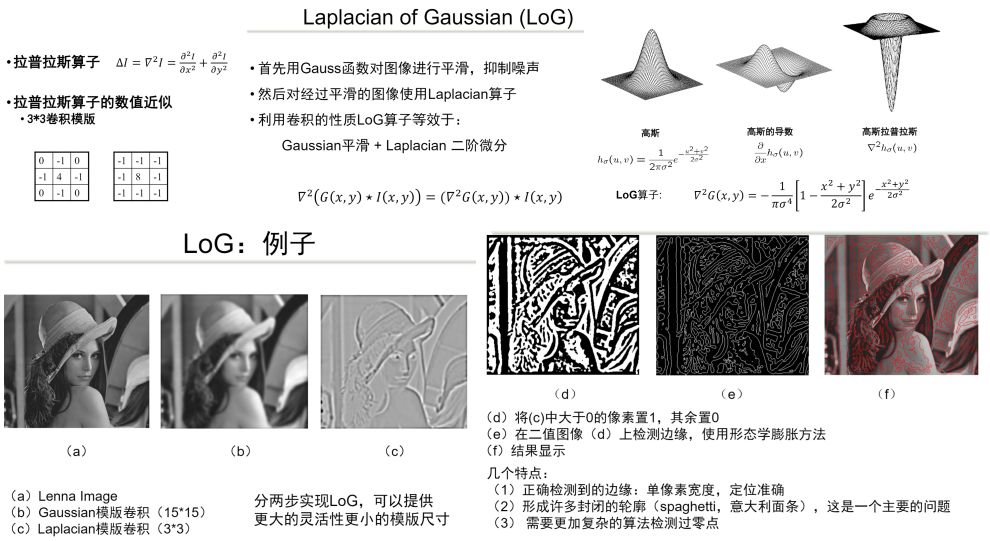
微分滤波器提取边缘

- 一阶微分滤波器（梯度算子）：Prewitt、Sobel，检测到图像一阶导数极大值就是边缘了



- 二阶微分滤波器：拉普拉斯算子（Laplacian），高斯拉普拉斯算子（Laplacianof Gaussian，LoG），检测到图像二阶导数过零点就是边缘了
- 当图像存在噪声时，求梯度会得到大量锯齿状结果（拉普拉斯算子要二次求导数，对噪声非常敏感；除此之外，拉普拉斯算子的运算结果是标量（幅值），丢失了方向信息）；解决噪声方法是先对图像进行滤波（比如高斯核函数和原图像卷积从而平滑）进而去噪，再去求梯度。

- LoG: 因其形状，也称为Mexican hat；求和为0，即对平坦图像区域的响应为0（$\iint \nabla^2 G(x,y)dxdy = 0$）

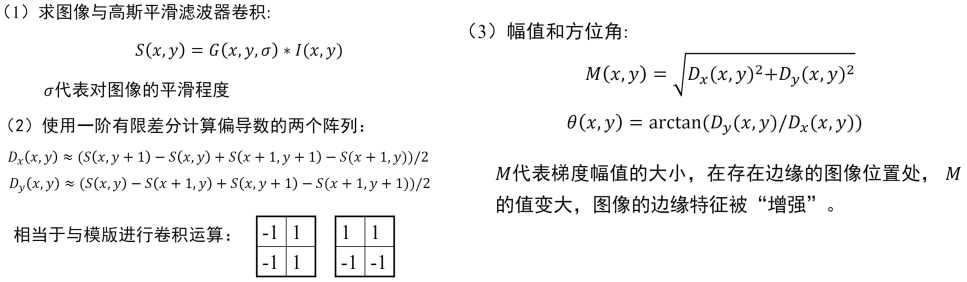


Canny边缘检测器

- 好的边缘检测器应该满足：1. 好的边缘检测性能（Good detection）：对边缘的响应大于对噪声的响应 2. 好的定位性能（Good localization）：其极值点应接近边缘的实际位置 3. 低的错误检测率（Low false positives）：在边缘附近只有一个极值点

- Canny边缘检测算法流程：

1. 计算图像梯度梯度（幅值还有方向） 1.1 用高斯核函数和原图像卷积从而平滑图像（抗噪） 1.2 使用一阶有限差分计算偏导数（实际上等价于图像与模版进行卷积运算） 1.3 从偏导数得到幅值和方向（边缘处的幅值会比较大，于是便找到边缘了）

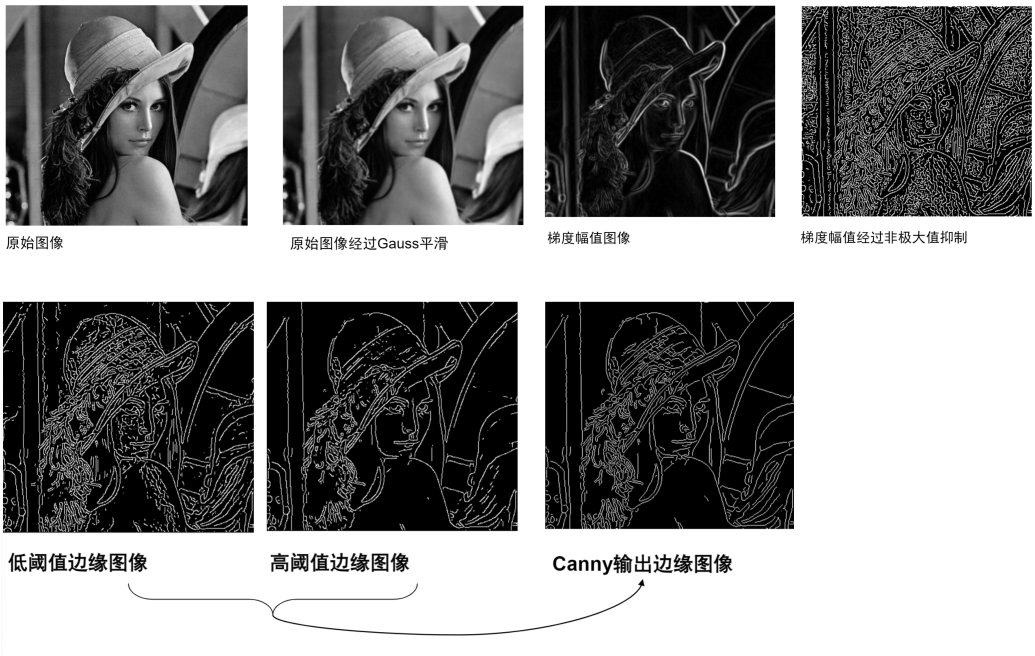


2. 梯度的非极大值抑制（NMS：Non-Maxima Suppression）：对于梯度幅值图像，仅保留梯度方向上的极大值点。 2.1 目的： 第一步认为梯度比较大的就是边缘处（接着定义一个阈值就完成边缘检测了），但存在边缘处有一片区域的点的梯度都比较大（这是因为边缘是有宽度的），为了让边缘处只有一个极值点，就要“抑制”一些另外的点。 2.2 方法： 梯度幅值图像记为M(x,y)，得到的结果记为N(x,y)。首先初始化N(x,y) = M(x,y)。然后对于每个点，在梯度方向和反梯度方向各找n个像素点；若M(x,y)不是这些点中的最大点，则将N(x,y)置零，否则保持N(x,y)不变。 2.3 结果： N（x, y）包含边缘的宽度为1个像素

3. 双阈值提取边缘点 2.1 目的： 第二步得到的N（x, y）使用阈值进行二值化会有问题：若使用大的阈值，将会得到少量边缘点，于是很多实际边缘点没有得到（产生许多空隙）；若使用小的阈值，会得到大量边缘点，同时也会有大量错误检测。这实际上是单阈值带来的问题。 2.2 方法： 双阈值提取边缘点：用两个阈值（$T1, T2$）提取边缘（也就是二值化）；其中 $T2$远大于 $T1$；这样 $T2$得到的高阈值边缘图（$E2(x,y)$）点少但更加可靠、$T1$得到的低阈值边缘图（$E1(x,y)$）虽然有更大的错误率但是点更多。 然后将两个边缘图连接起来：1. 将$E2(x,y)$中相连的边缘点输出为一幅边缘图像$E(x,y)$ 2. 对于$E(x,y)$中每条边，从端点出发在$E1(x,y)$中寻找其延长的部分，直至与

$E(x,y)$中另外一条边的端点相连，否则认为$E1(x,y)$中没有它延长的部分3. 将$E(x,y)$作为结果输出

- 示意图



- 算法评价 Canny算子的优点：参数较少；计算效率高；得到的边缘连续完整。 参数的选择：1. Gauss滤波的尺度 2. 双阈值的选择(经验性的：LOW=HIGH*0.4)

特征点检测

提取特征点的用途：使用特征点来代表图像的内容。需要特征点有一定特异性，并且在不同图像中都能检测出来。

角点是图像的一种特征点，指在图像中出现尖锐变化的像素点。角点具备在在不同的视角、光照和图像变换下的稳定性。

Harris角点检测算法

- 思想： 为每个像素点附上一个窗口（比如邻近9方格），角点便定义为窗口向任意方向的移动都导致图像灰度的明显变化的点。

- 算法流程：

  1. 将图像转换为灰度图像
  2. 利用Sobel滤波器计算I_x, I_y，从而通过乘法算出I_x^2, I_xI_y, I_y^2
  3. 将高斯滤波器分别作用于I_x^2, I_xI_y, I_y^2
  4. 计算每个像素的R
  5. 对角点响应函数R进行阈值处理
  6. 提取R的局部极大值，其对应的点就是角点

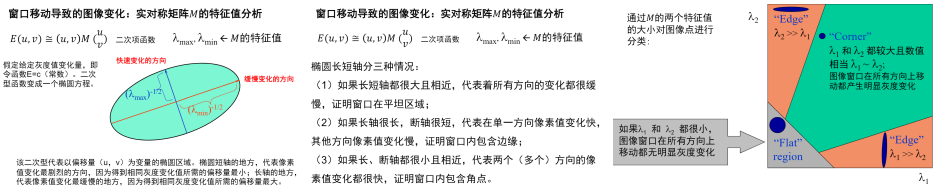
- 推导：

  1. 对于每一个点，附上一个窗口（(x,y)在窗口内则$w(x,y)>0$，否则$w(x,y)=0$，可以是高斯函数）。那么将窗口平移$[u,v]$将产生灰度变化 $E(u,v)=\sum_{x,y} w(x,y)[I(x+u,y+v)-I(x,y)]^2$，而$I(x+u,y+v)=I(x,y)+I_xu+I_yv+O\left(u^2+v^2\right)$，因此 $E(u,v)=$

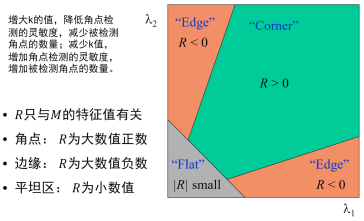
$\sum_{x,y} w(x,y)\left[I_x u + I_y v + O\left(u^2+v^2\right)\right]^2$，进一步由于 $\left(I_x u + I_y v\right)^2 = I_x^2 u^2 + 2 I_x I_y uv + I_y^2 v^2 = (u,v)\begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}$

2. 于是对于局部微小的移动量$[u,v]$（任意方向，此时$O\left(u^2+v^2\right)=0$）存在：$E(u,v) \cong (u,v) M \begin{pmatrix} u \\ v \end{pmatrix}$，其中 $M = \sum_{x,y} w(x,y)\begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$，注意这里的 M 和(x,y)相关，每个(x,y)确定一个窗口（比如邻近8个像素），再由这9个像素的梯度值得到M。而这也可以通过滤波（卷积操作）完成。

3. 分析M：使用M 的特征值表达图像点局部灰度变化的情况；（实际上是做了实验，发现椭圆的三种形状和平坦、边缘、角点对得上，于是便从椭圆形状入手设计R）



4. 定义角点响应函数R：一个好的角点沿着任意方向移动都将导致明显的图像灰度变化，即：R具有大的正数值。 $R = \det M - k(\operatorname{trace} M)^2, \det M = \lambda_1\lambda_2, \operatorname{trace} M = \lambda_1 + \lambda_2$。$k$是empirical constant, $k = 0.04 - 0.06$。



- Harris角点的性质
  - 角点响应函数R对于图像的旋转具有不变性：图像旋转前后对应的椭圆形状不变（特征值不变）
  - 对于图像灰度的仿射变化具有部分的不变性：M的计算只使用了图像导数，因此对于灰度亮度变化不变（$I \rightarrow I + a$），对于图像灰度的对比度变化（$I \rightarrow aI$）。
  - 对于图像几何尺度变化不具有不变性，随尺度变化，Harris角点检测的性能下降。弧形边缘缩得很小变成角点。
  - Shi-Tomasi发现，角点的稳定性和矩阵M的较小特征值有关，于是直接用较小的那个特征值作为R，这样就不需调整k值。
- 代码

  首先读取图片

```
import cv2
import numpy as np
# 读取图片
img = cv2.imread('lena512color.tiff')
# 转换为灰度图
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

  然后利用Sobel滤波器计算梯度

```
# cv2.Sobel 算导数
# - src: the input image
```

```
# - ddepth: the depth of the output image. Use cv2.CV_64F for 64-bit floating point output
#   or cv2.CV_8U for 8-bit unsigned integer output.
# - dx: the order of the derivative in x direction
# - dy: the order of the derivative in y direction
# - ksize: the size of the Sobel kernel. It must be 1, 3, 5, or 7.
I_x = cv2.Sobel(gray, cv2.CV_64F, 1, 0, ksize=3)
I_y = cv2.Sobel(gray, cv2.CV_64F, 0, 1, ksize=3)
I_x2 = I_x ** 2
I_y2 = I_y ** 2
I_xy = I_x * I_y
```

接着计算每个像素的R，并对角点响应函数R进行阈值处理（这里我设为1.0）

```
window_size = 3
k = 0.04
corner_list = []
offset = window_size // 2
height, width = gray.shape
for y in range(offset, height - offset):
    for x in range(offset, width - offset):
        # 对每一个点对应的窗口计算得到一个分数r，根据r的大小来判定这个点是否为harris特征角点。
        S_x2 = np.sum(I_x2[y - offset:y + offset + 1, x - offset:x + offset + 1])
        S_y2 = np.sum(I_y2[y - offset:y + offset + 1, x - offset:x + offset + 1])
        S_xy = np.sum(I_xy[y - offset:y + offset + 1, x - offset:x + offset + 1])
        det = (S_x2 * S_y2) - (S_xy ** 2)
        trace = S_x2 + S_y2
        r = det - k * (trace ** 2)
        # 对角点响应函数R进行阈值处理
        if r > 1.0:
            corner_list.append([x, y, r])
```

最后提取R的局部极大值（这一部分耗时特别长，因此我设置为从大到小500个点就结束了）

```
dst = np.zeros(gray.shape, dtype=np.float32)

# 对r进行非极大值抑制，进而减少得到的角点数量
corner_list = sorted(corner_list, key=lambda x: x[2], reverse=True)
new_corner_list = []
for i in range(len(corner_list)):
    for j in range(i+1, len(corner_list)):
        if abs(corner_list[i][0] - corner_list[j][0]) < window_size and
                    abs(corner_list[i][1] - corner_list[j][1]) < window_size:
            corner_list[j][2] = 0
    if corner_list[i][2] > 0:
        new_corner_list.append(corner_list[i])
    if len(new_corner_list) >= 500:
        break
corner_list = new_corner_list
for corner in corner_list:
    x, y, r = corner
    dst[y, x] = r
```

最后美化一下输出图像：

```
# 膨胀操作，让检测出的角点的位置更加明显
dst = cv2.dilate(dst, None)
img[dst > 0.01 * dst.max()] = [0, 0, 255]


# 显示结果
cv2.imshow('Harris Corners', img)
cv2.imwrite('Harris_Corners_lena.png', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

FAST

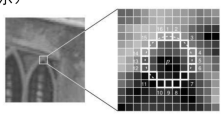
FAST(Features from Accelerated Segment Test, 2010)

- 算法特点：快（1ms内出结果，是Harris的20倍），兼顾准确性。常用于机器人等实时性要求高的领域。
- 思想：若该点的灰度值比其周围领域内足够多的像素点的灰度值大或者小（人为设置阈值t），则该点可能为角点。
- 添加了决策树这种机器学习方法来加速检测。
- 同样有非极大值抑制的步骤，没有旋转不变性，和Harris一样没有尺度不变性。

FAST算法步骤

1. 从图片中选取一个像素P，下面我们将判断它是否是一个特征点。我们首先把它的亮度值设为I_p。
2. 设定一个合适的阈值t。
3.考虑以该像素点为中心的一个半径等于3像素的离散化的Bresenham圆，这个圆的边界上有16个像素.
4.如果在这个大小为16个像素的圆上有n个连续的像素点，它们的像素值要么都比I_p+t大，要么都比I_p−t小，那么它就是一个角点。(白色虚线所示）

n的值可以设置为12
或者9，实验证明选择
9可能会有更好的效果。



- 对于图像中的每一个点，我们都要去遍历其邻域圆上的16个点的像素，效率较低
- 提出了一种高效的测试（high speed test）来快速排除一大部分非角点的像素。n=12

1.仅检查在位置1，9，5和13四个位置的像素，首先检测位置1和位置9，如果它们都比阈值暗或比阈值亮，再检测位置5和位置13。

2.如果p是一个角点，那么上述四个像素点中至少有3个应该必须都大于I_p+t或者小于I_p−t，因为若是一个角点，超过四分之三圆的部分应该满足判断条件。如果不满足，那么p不可能是一个角点。

3.对于所有点做上面这一部分初步的检测后，符合条件的将成为候选的角点，再对候选的角点，做完整的测试，即检测圆上的所有点。

算法缺点

- 当我们设置n<12时就不能使用快速算法来过滤非角点的点；
- 快速测试像素的选择和排序依赖于特征外观的分布情况
- 在后续的进一步详细测试中，之前的4次测试结果被丢弃了
- 处于邻域关系的多个特征被检测出来

机器学习一个角点检测算法：

- 选取角点提取场景下的多张测试图片
- 使用FAST算法计算每幅图像的特征，不采用high speed test方法
- 对于邻域圆上的每一个位置$x \in \{1,\dots,16\}$，其对应的像素相对于像素p而言表示为$p \to x$，该像素有三种状态：

$$S_{p\to x}=\begin{cases} d, I_{p\to x} \le I_p - t\ (darker) \\ s, I_p - t < I_{p\to x} < I_p + t\ (similar) \\ b, I_p + t \le I_{p\to x}\ (brighter) \end{cases}$$

- P表示所有训练图像中像素的集合，根据选择的位置x，那么可以把集合P根据上述三种类别状态，分成三个子集P_d, P_s, P_b
- 定义一个布尔变量K_p，如果p是角点，该值为1，否则为0
- 使用ID3算法递归计算所有子集，直到所有子集熵为0
- 创建的决策树用于FAST检测

机器学习一个角点检测算法：

非极大值抑制：

1.为每一个检测到的特征点计算它的响应大小（score function）V。这里V定义为点p和它周围16个像素点的绝对偏差的和。

2.考虑两个相邻的特征点，并比较它们的V值。

3. V 值较低的点将会被删除。

FAST需要改进的地方：

1.由于FAST算法依赖于一个阈值t，因此算法还需要人为干涉；

2. FAST算法不产生多尺度特征而且FAST特征点没有方向信息，这样就会失去旋转不变性

Table 1. Timing results for a selection of feature detectors run on fields (768 × 288) of a PAL video sequence in milliseconds, and as a percentage of the processing budget per frame. Note that since PAL and NTSC, DV and 30Hz VGA (common for web-cams) have approximately the same pixel rate, the percentages are widely applicable. Approximately 500 features per field are detected.

Detector	Opteron 2.6GHz		Pentium III 850MHz	
	ms	%	ms	%
Fast $n = 9$ (non-max suppression)	1.33	6.65	5.29	26.5
Fast $n = 9$ (raw)	1.08	5.40	4.34	21.7
Fast $n = 12$ (non-max suppression)	1.34	6.70	4.60	23.0
Fast $n = 12$ (raw)	1.17	5.85	4.31	21.5
Original FAST $n = 12$ (non-max suppression)	1.59	7.95	9.60	48.0
Original FAST $n = 12$ (raw)	1.49	7.45	9.25	48.5
Harris	24.0	120	166	830
DoG	60.1	301	345	1280
SUSAN	7.58	37.9	27.5	137.5

目前还没有方法可以兼顾处理如下问题：抑制噪声；定位精度；计算复杂程度

底层视觉：特征描述和匹配

SIFT：特征点检测、描述与匹配

- Scale Invariant Feature Transform(尺度不变的特征变化), Lowe 1999, 2004.
- 为何希望尺度不变？对于双目视觉，两个相机拍摄的两张照片的尺度相近，容易完成同个物体在两张图像上的特征点提取，之后再进行匹配；但对于单目视觉，两张图片见的尺度差异大，Harris满足不了尺度不变，就可能出现对于同一个物体，第一张图片检测到了角点而第二张就检测不到，无法进行后续的匹配。
- 思想：DoG特征检测＋SIFT 描述子（128维）+匹配。
- 性能：SIFT描述子性能评价最佳（Mikolajczyk&Schmid, 2005），迄今使用最为广泛的一种特征。

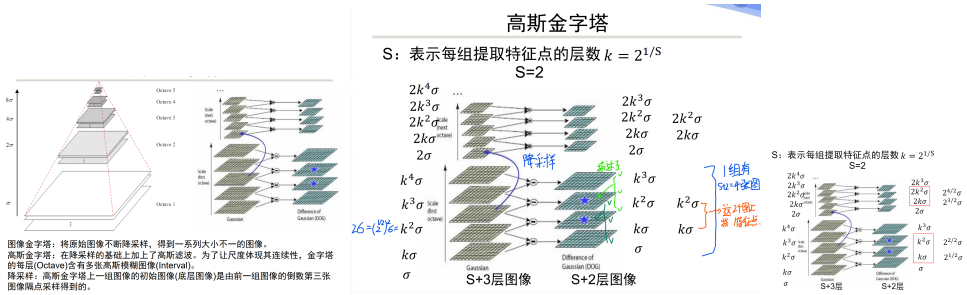
1.SIFT特征的特点

- 不变性：
  - 对图像的旋转和尺度变化具有不变性
  - 对三维视角变化和光照变化具有很强的适应性
  - 局部特征，在遮挡和场景杂乱时仍保持不变性
- 辨别力强：特征之间相互区分的能力（通过SIFT 描述子表示）强，有利于匹配
- 数量较多：一般500×500的图像能提取出约2000个特征点
- 扩展性强：能够与其他形式的特征向量联合

2.SIFT的流程

1. 特征点检测：多尺度空间极值点检测。

   用高斯金字塔找到在多个尺度空间内都有很好表现力的特征点。（金字塔的目的是找到图像不同的尺度空间，高斯指DoG，目的是找到具有尺度不变性的特征点）。然后在三维尺度空间（DoG金字塔同层的相邻的3张图片）中，搜索每个点的26（3^3-1）邻域，若该点为局部极值点，则保存为候选关键点。

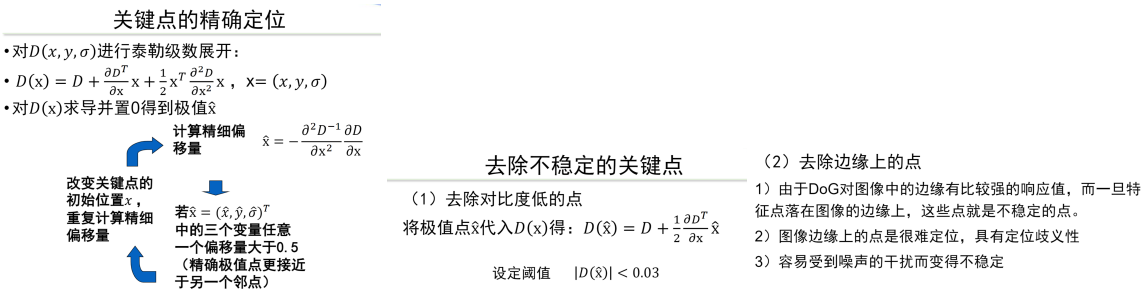


2. 特征点检测：关键点的精确定位。

   动机：因为在离散采样中搜索到的极值点不一定是真实空间的极值点。

   基本原理：对尺度空间DoG函数进行曲线拟合，计算其极值点，从而实现关键点的精确定位。

   接着去除不稳定的关键点：1. 去除对比度低的点 2. 去除边缘上的点



（2）去除边缘上的点

DoG响应峰值往往在横跨边缘的地方有较大的主曲率，而在垂直边缘的方向有较小的主曲率。主曲率与2×2的Hessian矩阵H的特征值成正比

$$H=\begin{bmatrix}D_{xx} & D_{xy}\\ D_{xy} & D_{yy}\end{bmatrix}$$

D 值可以通过求取邻近点像素的差分得到

（2）去除边缘上的点

为了避免直接求特征值，减少计算量。设两个特征值之间的比值为r：$\lambda_1=r\lambda_2$ r=1时最小

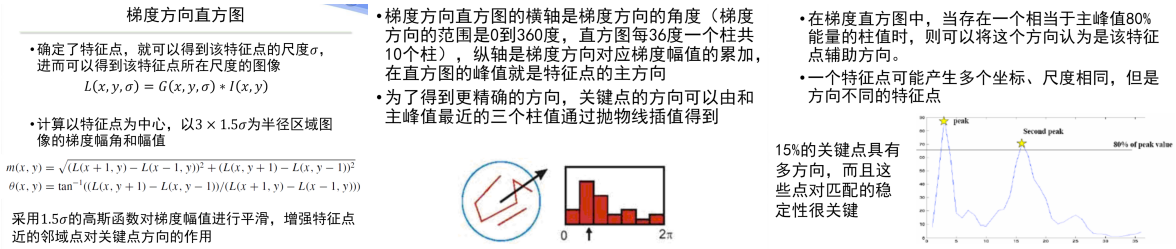
$$\frac{\mathrm{Tr}^2(H)}{\mathrm{Det}(H)}=\frac{(\lambda_1+\lambda_2)^2}{\lambda_1\lambda_2}=\frac{(r\lambda_2+\lambda_2)^2}{r\lambda_2\lambda_2}=\frac{(r+1)^2}{r}>\frac{(T_r+1)^2}{T_r}$$

如果上式成立，则剔除该特征点，否则保留。T_r=10

3. 特征点描述：关键点的主方向计算。 同样，利用图像点周围的信息来描述点。这种描述要有一定的不变性。

   动机：直方图可以克服局部形变，但不具有图像旋转不变性（直方图定义：表示图像中灰度之间的统计关系，用横坐标表示灰度级，纵坐标表示频数），于是我们可以对检测出的特征点对应的窗口进行直方

图统计。为了达到旋转不变性，计算特征点的主方向，然后将旋转前后的主方向进行对齐，再去构造窗口，这样就保证了旋转前后的描述区域是不变的。

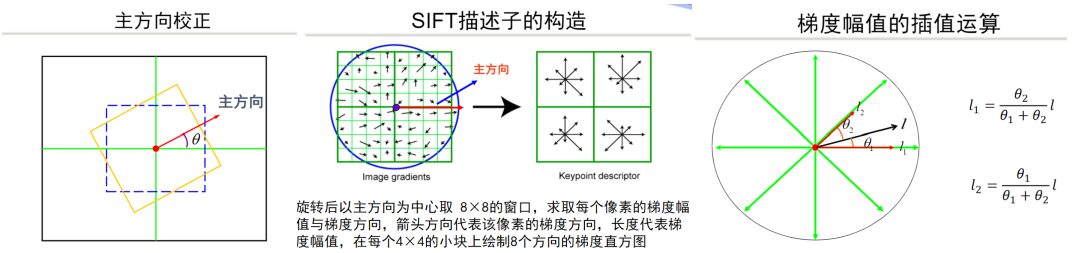


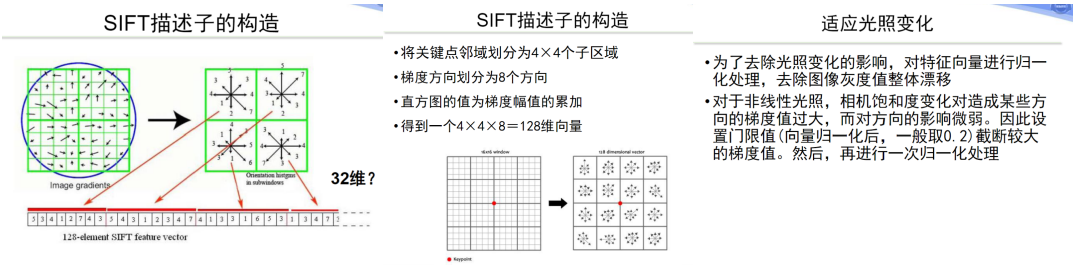
4. 特征点描述：描述子的构造。 方法：

    1. 校正旋转主方向，确保旋转不变性。

    2. 生成描述子，最终形成一个128维的特征向量 2.1 将关键点邻域划分为4×4个子区域 2.2 梯度方向划分为8个方向 2.3 直方图的值为梯度幅值的累加 2.4 得到一个4×4×8＝128维向量（下面前四个示意图算出来是32，这是为了好介绍，实际上采用了更大的区域来描述）

    3. 归一化处理，将特征向量长度进行归一化处理，进一步去除光照的影响





5. 特征点匹配：

    匹配算法：NNDR (Nearest Neighbor Distance Ratio)：$NNDR=\frac{distance(m,m1)}{distance(m,m2)}<r$，$m1$是$m$最近邻，$m2$是$m$次近邻，若两对点距离之间差距显著就认为匹配上了。

    搜索策略：BBF算法(Best Bin First)（Beis&Lowe, 1997）或者ANN

3.SIFT的原理

我们需要检测在尺度变化时仍然稳定的特征。尺度归一化的LoG空间具有真正的尺度不变性(Lindegerg1994)。从尺度归一化LoG空间中提取的图像特征(极大值和极小值)的尺度稳定性最好，优于梯度、Hessian或Harris角点函数。但是运算量过大。于是用高斯差分（DoG: Difference of Gaussian）近似尺度归一化LoG。SIFT算法就在DoG尺度空间中提取极值点并进行优化从而获取特征点。

3.1 由LoG到DoG：

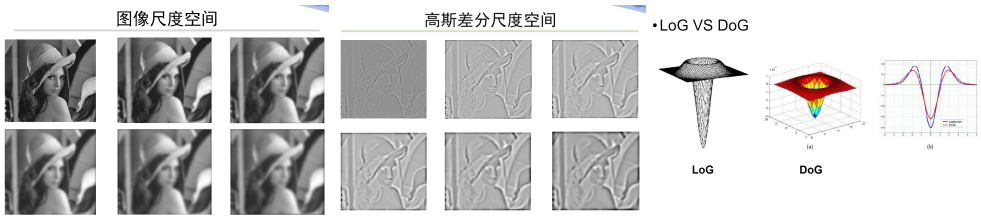
1. 定义的$I(x,y)$的尺度空间（就是给原图像不同程度的平滑得到图像空间）：$L(x,y,\sigma)=G(x,y,\sigma)*I(x,y)$，其中$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$。尺度参数$\sigma$，当$\sigma$连续变化，$L(x,y,\sigma)$便构成图像的尺度空间。

2. scale-normalized Laplacian of Gaussian(尺度归一化的LoG空间)有真正的尺度不变性：$\sigma^2\nabla^2 G = \sigma^2\left(G_{xx}(x,y,\sigma)+G_{yy}(x,y,\sigma)\right)$

3. 高斯差分是对尺度归一化LoG的一个很好的近似：由性质 $\frac{\partial G}{\partial \sigma} = \sigma\nabla^2 G$ 得到 $\sigma\nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x,y,k\sigma)-G(x,y,\sigma)}{k\sigma-\sigma}$， 于是$G(x,y,k\sigma)-G(x,y,\sigma)\approx(k-1)\sigma^2\nabla^2 G$；虽然有个(k-1)的常数因子，但一方面我们已经找到了快速近似计算$\sigma^2\nabla^2 G$的方法，另一方面可以(k-1)在所有尺度空间上都一样

4. 定义 $D(x,y,\sigma) = (G(x,y,k\sigma)-G(x,y,\sigma)) * I(x,y) = L(x,y,k\sigma)-L(x,y,\sigma)$，因此我们在$D(x,y,\sigma)$上进行极值点检测即可。这便从尺度归一化的LoG空间变到DoG尺度空间进行极值点检测。

5. 比较LoG和DoG：LoG是Laplacian of Gaussian，先对图像高斯滤波，再使用Laplacian算子求二阶微分。DoG是Difference of Gaussian，先对图像进行不同程度的高斯滤波，然后结果相减。



3.2 高效计算

定理：“原图的1/2下采样得到的图”进行σ高斯卷积，就等价于“原图”进行2σ高斯卷积后再进行1/2采样。

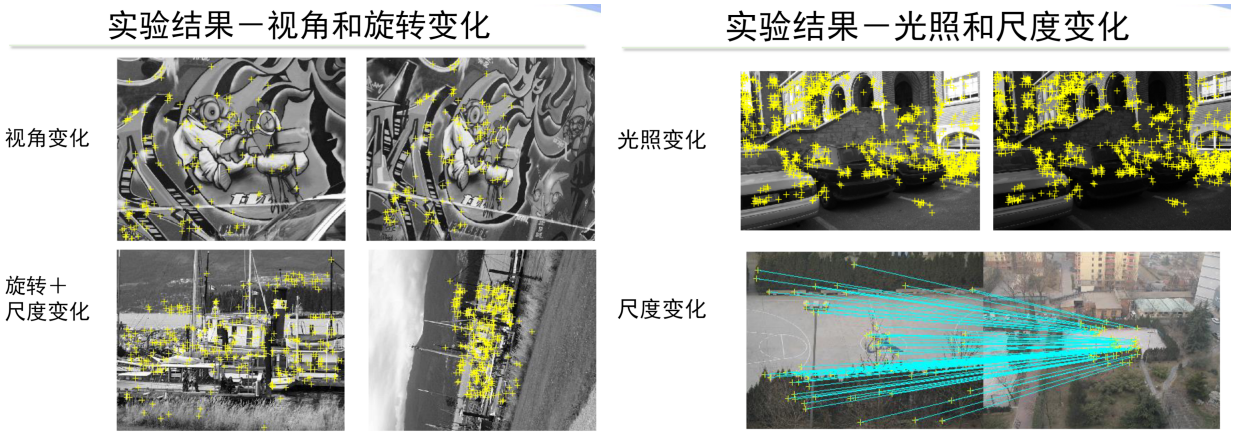
因此上面高斯金字塔的第二层的第一张图的 2σ指的是原图进行2σ高斯卷积后再进行1/2采样得到的结果，也指的是对于“原图的1/2下采样得到的图”进行σ高斯卷积得到的结果，我们要算的是后者，后者可以由先前结果（原图进行2σ高斯卷积，第一层的第3张图）高效得到。

3.3 SIFT描述子的特点：

1. 直方图统计：提高了对图像局部形变的适应能力
2. 子区域划分：弥补了丢失的位置信息，增强辨别力
3. 16×16的邻域和4×4的子区域都进行了类似于高斯函数的加权处理，强化中心区域，淡化边缘区域的影响

3.4 SIFT描述子的不变性：

1. 尺度不变：根据关键点的尺度选取高斯图像和邻域大小
2. 旋转不变：将邻域内每点的位置和梯度方向根据关键点的主方向进行旋转
3. 适应复杂几何变形：采用分块直方图统计、高斯加权等细节处理
4. 适应复杂光照变化：线性光照：归一化128维向量；非线性光照：将128维中所有大于0.2的元素赋值为0.2



特征匹配

匹配的定义：确定不同图像中对应空间同一物体的投影的过程；匹配是基于多幅图像视觉问题的的基本步骤。分类：点匹配；直线匹配；曲线匹配；区域匹配。

点匹配的思路：利用图像点周围的信息来描述点，如灰度信息，颜色信息，梯度信息等，然后进行相似性度量。

Cross-correlation

原理：利用相关函数（感觉就是协方差的思想），评价两幅图像特征点邻域的灰度相似性以确定对应点。

过程：首先对于两幅图，左图的特征点附近形成一个窗口，在右图的搜索区域中通过计算相关函数来找相关的窗口（可能找到很多个）。这样便从左图的一个特征点找到右图的一个候选匹配集。同理，我们也能从右图的一个特征点找到左图的一个候选匹配集。最后理想中匹配的点对应该是彼此存在于对方的候选匹配集中。

相关函数

$$Score(m_1, m_2) = \frac{\sum_{i=-n}^{n}\sum_{j=-m}^{m}[I_1(u_1+i, v_1+j) - \bar{I}_1(u_1, v_1)] \times [I_2(u_2+i, v_2+j) - \bar{I}_2(u_2, v_2)]}{(2n+1)(2m+1)\sqrt{\sigma^2(I_1) \times \sigma^2(I_2)}}$$

$$\bar{I}_k(u, v) = \frac{\sum_{i=-n}^{n}\sum_{j=-m}^{m} I_k(u+i, v+j)}{(2n+1)(2m+1)}$$ **均值**

$$\sigma(I_k) = \sqrt{\frac{\sum_{i=-n}^{n}\sum_{j=-m}^{m}[I_k(u+i, v+j) - \bar{I}_k(u, v)]^2}{(2n+1)(2m+1)}}$$ **标准差**

特点：1. 基于图像灰度 2. 如何确定窗口大小和形状是最大的问题 3. 没有旋转不变性（图像旋转后窗口内容发生变化） 4. 对光照变化敏感 5. 计算代价大

SIFT的特征点匹配就是要改进这个。

ORB（Oriented FAST and Rotated BRIEF）

动机：和FAST一样，快速性、兼顾准确性；oFAST是去除边缘、多尺度、抗旋转（带方向性)的FAST，用于检测。BRIEF(Binary Robust Independent Elementary Features)采用二进制描述子，特点：高速、低存储。匹配采用汉明距离（Hamming Distance）：在信息论中，两个等长字符串之间的汉明距离是两个字符串对应位置的不同字符的个数。

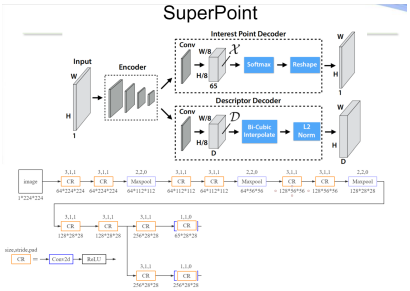
ORB是SIFT一种很弱的替代。虽然快（比SIFT快300倍以上）效果一般。

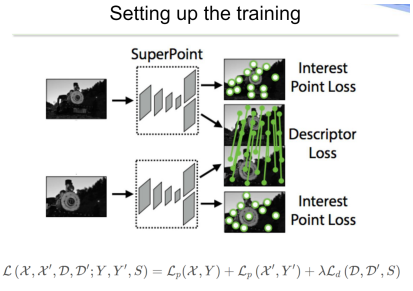
基于CNN的特征点与描述子学习

SuperPoint(2018 cvprw)：把VGG(2015)当做backbone，对输入图像给出两个输出：关键点位置和关键点的描述，这两个任务共享一部分主干网络结构。（可以学习一下网络设计、损失函数设计）

对于关键点检测的任务：将检测看作是分类任务，将图像分为 $8*8$ 的不重合区域，这个区域对应着65个标签（关键点的64中可能位置和无关键点，多个关键点的话随便选一个），让网络的输出是 $W/8*H/8*65$ ，65个数在通过softmax就输出对应的 $8*8$ 区域特征点的位置。

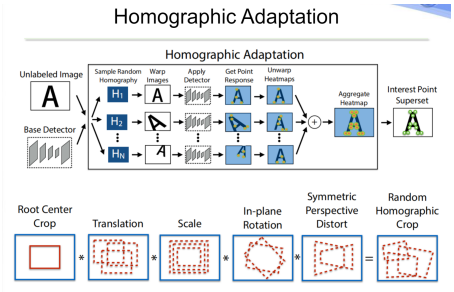
对于关键点描述的任务：根据关键点位置从 $W/8*H/8$ 个通道选择输出（要插值），损失函数暗含了匹配的思想。详见CSDN





$$\mathcal{L}(\mathcal{X},\mathcal{X}',\mathcal{D},\mathcal{D}';Y,Y',S)=\mathcal{L}_p(\mathcal{X},Y)+\mathcal{L}_p(\mathcal{X}',Y')+\lambda\mathcal{L}_d(\mathcal{D},\mathcal{D}',S)$$

需要大规模的有检测出特征点的图像的数据集：采用Synthetic Training的方式（简单有效、但噪声大）自动标注图像——首先从简单的、有标注的、合成的数据集上训练出神经网络，然后用此神经网络通过Homographic Adaptation的方式（类似于数据增广，将原图通过不同几何变形得到一批图像，然后通过同一个神经网络标注出兴趣点，再从变换后的图像恢复回原图，将找到的不同兴趣点进行聚合）为复杂的、无标注的、真实的数据集打上标签。

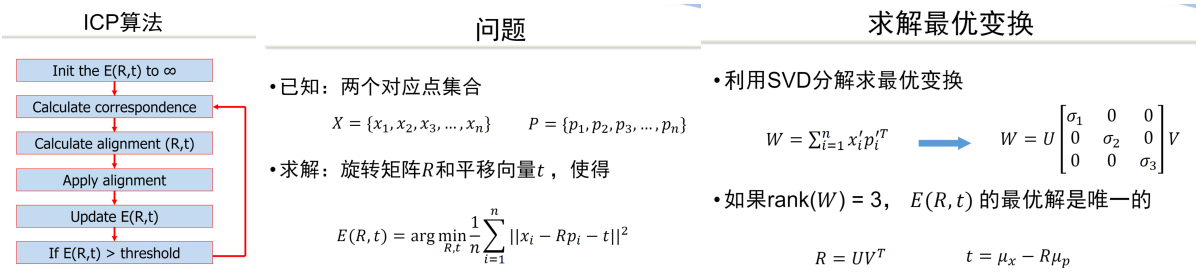


空间点匹配：ICP(Iterative Closest Point)

和前面SIFT等方法不一样，如果不知道点之间的对应关系，如何做两组点集之间的匹配。

思想：认为两个匹配的曲线、平面应该可以通过旋转和平移重合一部分。而如果知道正确的点对应，那么两个点集之间的相对旋转和平移有闭合解。

方法：1. 找到对应点（注意下面图二的下标）在ICP中，假设最近的点为对应点。2. 求解R,t。循环两部直到收敛。也就是对一条曲线不断使用旋转、平移变换到另一条曲线，虽然中间对应点会变。



变体：

1. 点集选取的方式(selecting)：选择所有点；均匀采样(Uniform Sampling)；随机采样(Random Sampling)；法方向空间均匀采样(Normal-space Uniform Sampling)
2. 点集匹配的方式(matching)：最近邻点(Closest point)；法方向最近邻点(Normal shooting)；投影法(Projection)）
3. 点集对应权重(weighting)：固定权重；距离权重；法方向权重

鲁棒匹配的RANSAC框架

鲁棒匹配问题特点：大量外点存在的数据直接拟合模型会有严重偏差。

RANSAC：排除外点干扰的算法：用排除外点的数据，重新拟合模型，得到最终结果。下面的#$S(M_p(J))$看做是赞成票，外点投反对票。

RANSAC估计模型M的一般步骤

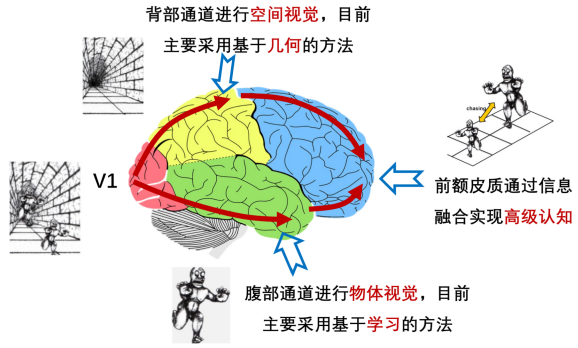
- 确定求解模型M，所需要的最小数据点的个数n。由n个数据点组成的子集称为模型M的一个样本；
- 从数据点集D中随机地抽取一个样本J，由该样本计算模型的一个实例$M_p(J)$，确定与$M_p(J)$之间几何距离< 阈值 t 的数据点所构成的集合，并记为$S(M_p(J))$，称为实例$M_p(J)$的一致集；
- 如果在一致集$S(M_p(J))$中数据点的个数# $S(M_p(J))$>阈值T，则用$S(M_p(J))$重新估计模型M，并输出结果；如果# $S(M_p(J))$<阈值T，返回到步骤2；
- 经过K次随机抽样，选择最大的一致集$S(M_p(J))$，用$S(M_p(J))$重新估计模型M，并输出结果。

3D视觉

目的：从二维图片出三维场景。

物体视觉就是识别物体，空间视觉是感知空间的结构特性、远近特性等三维信息。空间视觉比物体视觉快得多（“一道黑影闪过，再定睛一看”）。

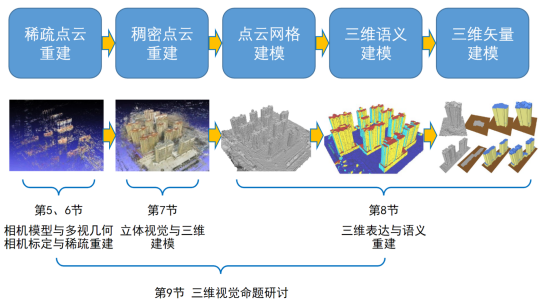
为什么学习三维视觉？



三维视觉典型应用：三维数字城市；三维运动捕捉；3D游戏与文化遗产保护；无人车三维地图与定位；数字孪生。

恢复三维信息的基本要素：恢复场景三维结构（场景重建）、理解场景语义信息（每个场景的基本结构分类）、相机六自由度空间位姿（相机的位置和相机的朝向）。

通过三维视觉课可以学到什么？



一个完整的图像三维重建系统

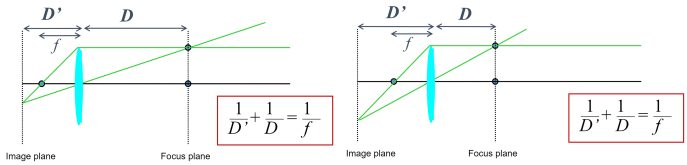
3D视觉：相机模型

成像模型发展脉络

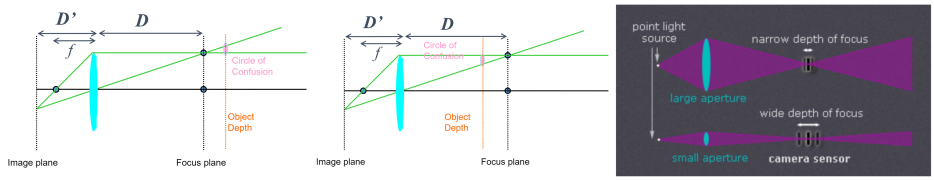
1. 将一张胶片放在物体前方，可以获得图像吗？不能，从物体的同个位置反射的光会布满整个胶片。结果就是完全曝光的全白底片。

2. 在物体和胶片之间，增加一块带有小孔的屏障（小孔成像）来阻挡了大部分光线，但是胶片上获得倒立的图像。屏障上的小孔称之为光圈。问题：小光圈（对应的曝光时间增长，曝光时间长的话拍不清运动的物体）；大光圈（对应的曝光时间短、得到模糊图像，这是因为从物体的同个位置反射的光有多条通过光圈）。但是光圈过小会产生衍射现象，同样导致图像模糊。

3. 透镜系统：用凸透镜汇聚从物体的同个位置反射的光，在曝光时间短的同时得到清晰图像（要满足下面公式：物体到凸透镜的距离倒数加成像面到凸透镜的距离倒数等于焦距倒数）。



4. 透镜系统的副产物：景深。在理想物距的附近一段距离也会清晰成像，而之外就模糊，这样就形成了景深。景深和光圈大小（就是凸透镜裸露在外的大小）有关。大光圈对应小景深，小光圈对应大景深。理想设置：小光圈，光线充足：这样清晰成像和景深范围大。



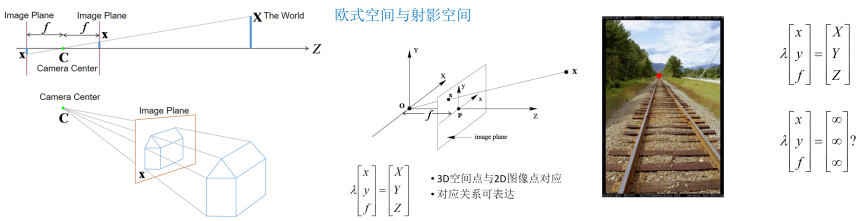
相机模型

在之后内容中，将相机模型看做是针孔相机模型（忽略凸透镜影响，忽略光圈影响；在精度要求高的应用中比如某种相机标定，就要考虑凸透镜会对针孔相机模型产生影响）。

欧式空间和射影空间

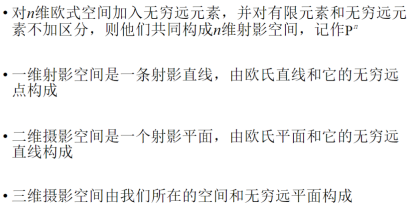
欧式空间：

- 首先将针孔成像模型等价变换：将成像面等价对应到相机中心（透镜光心）前（图一）
- 以相机中心为原点，可以写出比例关系（图二）
- 图片上的一个点，对应现实中无穷远点的时候，我们发现这个点无法参与我们的集合计算（因为有无穷值）（图三）。也就是说，在欧式空间中不是所有2D-3D点对都可以参加计算的，于是要换到射影空间。

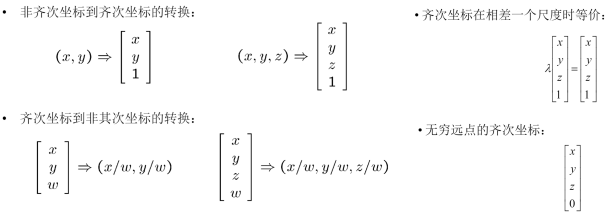


射影空间：

- 定义：把欧式空间的存在但不参与计算的无穷远元素纳入到欧式空间中，并且和有限远元素同等对待。
- 一维的欧式空间就是直线，两个无穷远点被看做是射影空间的一个无穷远点；二维的欧式空间是平面，所有的无穷远点被当成射影空间的一条直线，称为该平面的无穷远直线；三维空间中的所有无穷远点组成射影空间的一个平面，称为这个空间的无穷远平面。

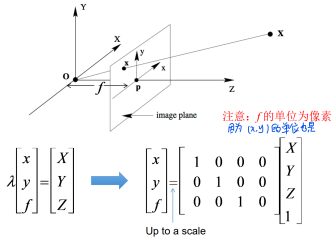


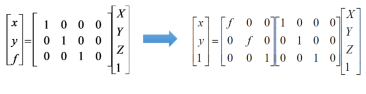
- 齐次坐标是射影空间的坐标表达方式。

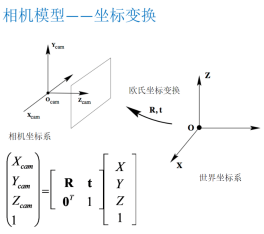


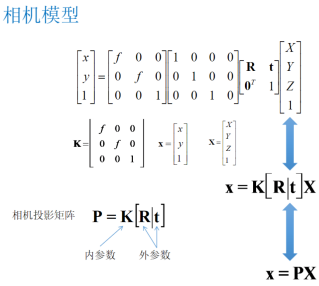
相机模型的数学表达

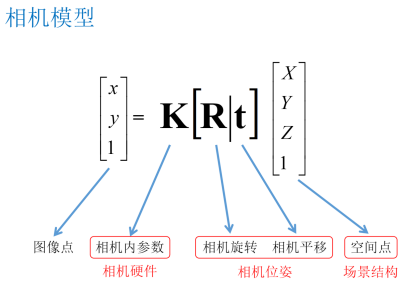
- 第一张图从欧式空间变到射影空间（由齐次坐标性质丢弃λ），f 的单位是像素（含义表示焦距在图像上会跨越多少个像素值）。其实左侧的f是在欧式空间中的，但右侧的f 我们将其看做是射影空间的多的那个维度，有点巧合。
- 第二张图的左侧等式等号两端分别乘以右侧等式的等号右侧第一个矩阵，再利用齐次坐标性质化简得到。
- 第三张图将相机坐标系换到世界坐标系（注意这里的矩阵可以看做是任意两个相机坐标系之间的转换矩阵！），看做是世界坐标系乘以一个旋转矩阵、平移向量得到相机坐标系（也就是第二张图的$[X\ Y\ Z\ 1]^T$，第三张图的$[X_{cam}\ Y_{cam}\ Z_{cam}\ 1]^T$）
- 第四张图的P的维度是$3*4$，就是相机六自由度空间位姿。





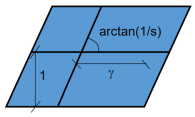






相机模型的内参数矩阵

$$\mathbf{K}=\begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{K}=\begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{K}=\begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

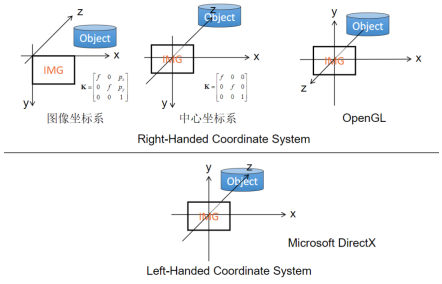


简化版　　完整版

CCD/CMOS, $f_x=f_y$, $s=0$

注意：f的单位为像素（f物理尺寸/单个CCD感光元宽度）

- 完整的内参有5个可变量，当感光元是正方形的时候$f_x = f_y$；如果感光元是平行四边形，$s \neq 0$。在今天的制造工艺，可以安全地认为CCD/CMOS就是理想的正方形，见上面第二个图。
- p_x, p_y是相机的光轴穿过图像的位置（主点）。理想情况下光轴就在图像中心，但现实中相机基本有偏移。
- 下面第一个图的最左侧图像坐标系是最常用的坐标系（原点在图像一角，避免负值），理想中p_x, p_y就是图像x,y轴长度的一半，但一般不满足。 中心坐标系是最理想的情况，一般不使用。
- OpenGL和Microsoft DirectX有着不同的坐标系。
- 前面推导都有个假设：镜头没发生畸变，发生畸变的时候现实中的直线在图像中会会变成曲线，见下面第二个图。其中r是点到图像中心的距离；k_1, k_2是两个畸变系数（镜头更复杂会有更多畸变系数）。这两个也是相机的内参数。
- 总结一下：内参至少一个焦距，至少两个主点坐标，至少两个畸变系数。

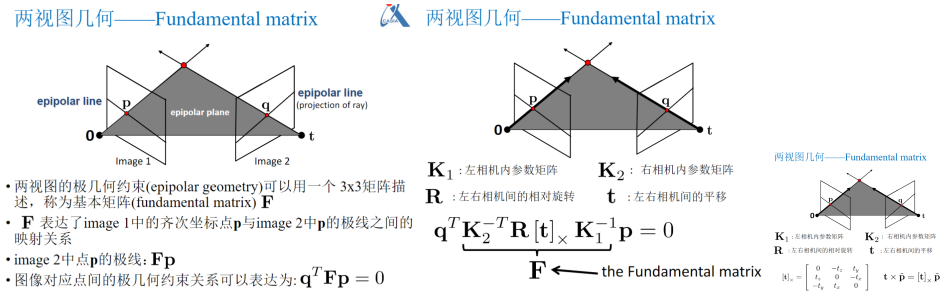


镜头畸变

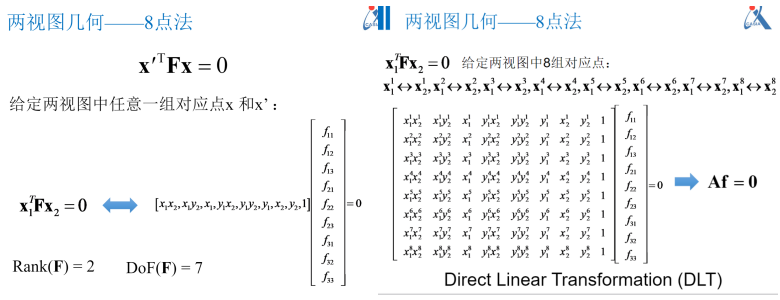
$$r^2 = \|\mathbf{x}\|^2 = x^2 + y^2$$
$$\mathbf{x}' = (1 + k_1 r^2 + k_2 r^4)\mathbf{x}$$

两视图几何

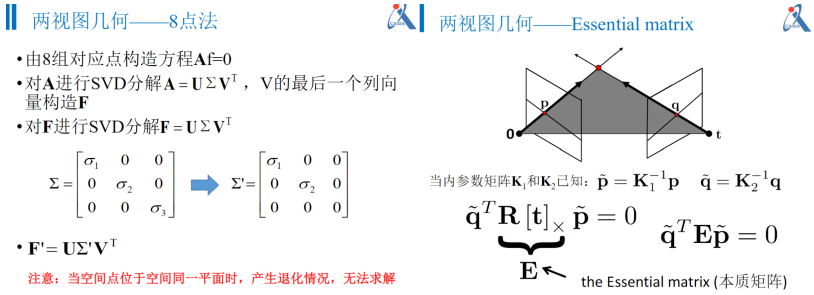
- 通过单幅图像无法重建场景结构（想一想同一个物体的三个垂直角度上的图）。
- 两视图几何是多视图几何中最简单的场景，通过两个图片（拍的相机可以不一样）重建所有信息。



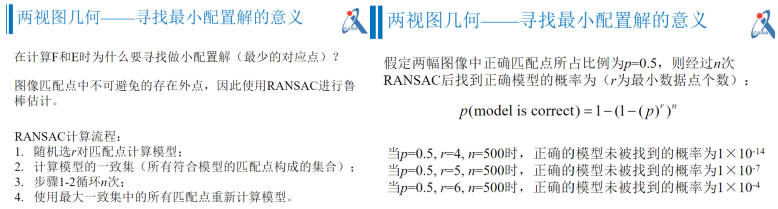
- 上面的第一张图将世界坐标系建立在image 1对应的相机的光心（点O）上。
- image 2上的线叫做image 1的点p在image 2上的epipolar line(对极线)。若image 2上有点p的对应点，那么它一定在这条epipolar line上。两条射线、两条epipolar line 构成了epipolar plane，这个平面包含了所有已知量和未知量，于是便从三维空间的求解到了二维的epipolar line上。
- 基本矩阵F是重要的概念，点q在线Fp上就是点的转置乘以线的方程等于0。p, q就是点的齐次坐标（3维）。基本矩阵反映的是空间中点在两个图像上对应点之间的约束关系。



- 8点法求基本矩阵：8组对应点（8组(p,q)）！关于8点法详见知乎
- 为何8组点？上面左图中，首先由于等号是齐次意义的，F有尺度歧义性（F增大或减小n倍都一样），于是可以直接将f_{33}保持为1；除此之外，F有几何性质约束导致$Rank(F)=2$。因此基本矩阵F自由度为7（F只有7个可以自由变化的量）；因此7组对应点就可以解了。但是$Rank(F)=2$的约束很难显式加到解析计算中的，因此为了简化计算，不考虑这个约束而选择多加一个点。
- 解这个方程的方式是对A进行SVD分解，将最小特征值对应的特征向量的每个元素重新排列就当做f。这种直接解析地解出未知量的方法叫DLT，意味着这种算法非常好。



- 因为图像匹配有误差，为了让从8点法解出的F 满足$Rank(F)=2$的约束，算一个F的近似值。
- 退化：当8个方程不是线性独立的。
- 相对合理地假设相机内参已知，然后求本质矩阵E。你当然可以用8点法求E，但是因为K_1, K_2已知，E是比F自由度更低。可以通过5点（5组点）法求本质矩阵E。
- 当求出E后，R, t是可以唯一地分解出来的。



- n越大，p越大（$p=0.5$是正常值），成功概率越高。同时r越小，成功概率越高。这意味着r越小，求解精度指数级增大（上面右图）。

3D视觉：稀疏重建

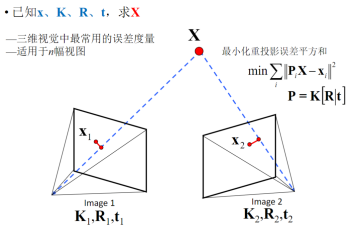
从多视角图像到相机位姿+场景结构（稀疏重建的步骤）

- 三角化：已知**x**、**K**、**R**、**t**，求**X**
- 相机标定：已知**x**、**X**，求**K**、**R**、**t**
- 姿态估计：已知**x**、**X** 、**K** ，求**R**、**t**
- 稀疏重建：已知**x**，求**K**、**R**、**t**、**X**
- 重投影误差最小化 $\min \sum_i \|\mathbf{P}\mathbf{X}_i - \mathbf{x}_i\|^2$
- 通过线性方法求解初始值（代数误差最小化）,通过非线性优化迭代求精（几何误差最小化）
- 相机标定与稀疏重建实践：Photo Tourism

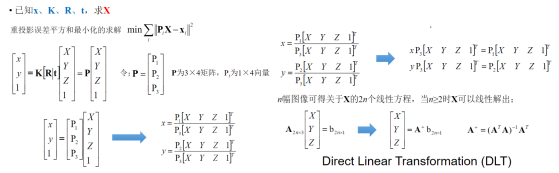
三角化（**Triangulation**）

常见于工业测量，其中多目相机已经预先标定好参数了。

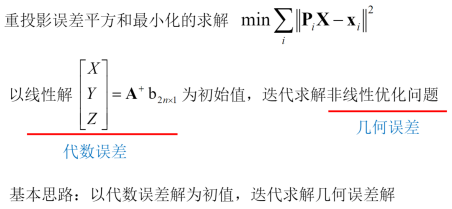
- 两视图三角化中由于误差的存在，匹配的点对应的视线往往不相交，于是采用两条视线公垂线的中点作为最终三角化的点。
- 这种公垂线只适合两视图，多视图没有这种概念，于是采用重投影误差平方和作为优化目标（很直观的几何概念）。

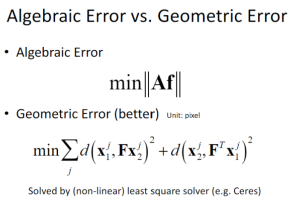


- 可以通过DLT求闭式解。未知量是3个，1个点给出2个方程，而X最多在图像上对应1个点，因此至少要2个点（2副图像）才能求解。A_{2n*3}不是方阵，采用伪逆，这样解准确么？不准确，因为这种解法得到的解只是使得代数意义下（此处得到的是满足超定方程的最小二乘解）误差最小，不具备我们想要的重投影误差平方和最小的的几何意义（叫做代数误差解）。



- DLT解出来的是代数误差解（代数意义下的最优解），作为初始值去求几何误差解（几何意义下的最优解）。



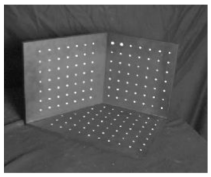


相机标定（**Camera Calibration**）

求相机内外参数。需要知道多组2D—3D对应，并知道对应点的2D与3D坐标。

1. 使用三维标定物

- 需要一个人工制作的标定物，要求标定物结构已知，标定物上特征点易于提取。



$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

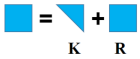
- 一组2D-3D对应点提供关于P的两个线性方程。n组2D-3D对应点提供关于P的2n个线性方程；当n≥6时，P可以通过DLT线性解出（P的自由度为11）
- 然后进一步用QR分解（有个对P的前三列的转置，因此这里叫RQ分解）求内外参。

从**P**中分解相机内外参数**K**、**R**、**t**：

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}] = [\mathbf{KR}|\mathbf{Kt}]$$

$\mathbf{K} = \begin{bmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$ 是上三角阵，**R**是正交矩阵

对**P**的前三列进行RQ分解：

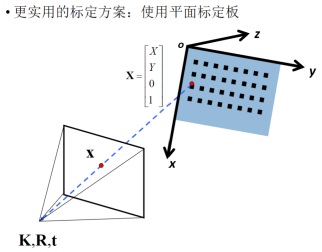


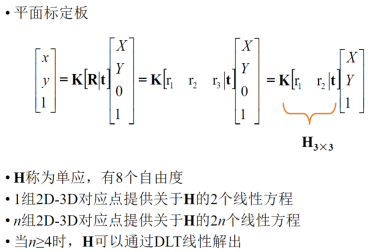
K-1乘以**P**的第四列得到**t**

- 使用三维标定物（如果所有点在同一平面，前面的方程会退化，因此需要保证三维空间点不在同一个平面上）的优缺点：优点：标定精度高；通过一幅图像即可标定。缺点：需要高精度的三维标定块。

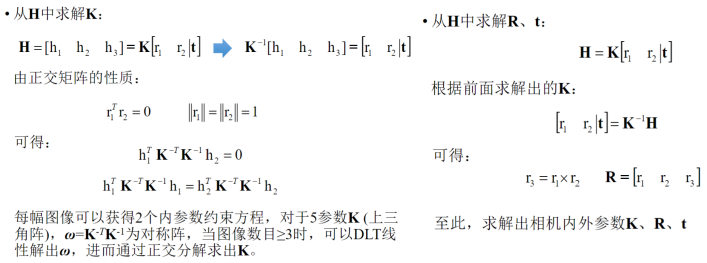
2. 使用平面标定板（最常用的方法，**2000**）

- 优点：容易制作（打印一张黑白棋盘格（容易检测出特征点）、一块足够平的木板）；标定工具箱成熟（Matlab、OpenCV）
- 缺点：标定精度不如三维标定物。
- 相机标定流程：打印一张模板并贴在一个平面上；从不同角度拍摄若干张模板图像；检测图像中的特征点；求解相机内外参数；分析重投影误差；输出标定结果。
- 推导： 以模板平面一角为原点建立世界坐标系，让z坐标为0，推导得到右图。发现此时所有三维点都在平面上，对应的图像点也在平面上。H是$3*3$的矩阵，叫做（平面）单应变换：空间的一个平面通过射影变换到另一个平面。同一个图像的四个点可以求出H。



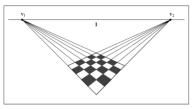


一幅图像给出K的2个方程。多幅图像虽然R, t变了，但K不变，因此用多幅图像可以安全地求K。右图由r_1, r_2叉乘得到r_3。

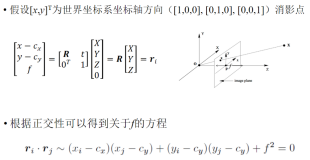


3. 利用消影点标定（需要假设**K**中只有**f**未知）

- 标定板都不用，只利用图像中的信息：消影点越靠近图像中心，f计算的越准确。
- 任何平行的直线都会交于同个消影点，而和直线的位置没有关系。
- (c_x, c_y)是主点的位置，消影点的第四维是0。我们希望找到世界坐标系中和轴平行方向的两个消影点（对应不同轴），这样其对应的$[X, Y, Z]$只有一个位置是1，其余两个是0，这样就完成了右图上面式子的推导。



消影点（vanishing points）：空间平行线在图像投影线的交点，对应三维空间中的无穷远点。消影点只与三维直线方向有关，与其位置无关。



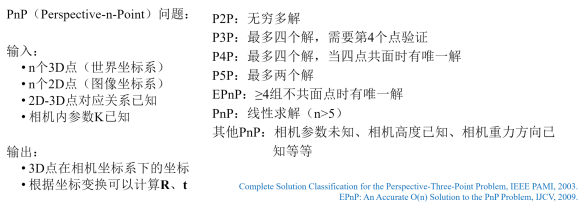
总结

- 前面三种方法都是DLT求解相机标定的初始值；
- DLT求解相机标定的优点：线性求解
- DLT求解相机标定的缺点：不包含相机畸变参数；无法添加其他约束（如已知相机焦距）；最小化代数误差（无几何意义）
- 以上述线性求解的K、R、t为初始值（代数误差最小化解），迭代求解重投影误差最小化问题得到几何误差最小化意义下的K、R、t

姿态估计（**Pose Estimation**）

三维地图已知，重新估计相机位置。已知x、X、K，求R、t。

- 最简单的方法：用与相机标定相同的DLT方法求解；6组2D-3D对应点提供关于P的12个线性方程；DLT线性求解P；然后从P中分解相机内外参数K、R、t；R、t即为所求。
- 姿态估计最少需要几组2D-3D对应点？一组2D-3D对应点提供关于P的2个约束方程，当K已知时，最少需要3组2D-3D对应点（但求得的不是唯一解。R, t一共6个自由度，顺便说一下，完整K是5个自由度，加一起正好是P的11个自由度）。
- PnP(Perspective-n-Point)

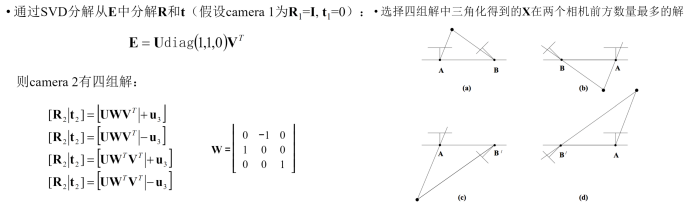


稀疏重建（**Sparse Reconstruction**）

也叫Structure from Motion (SfM), Structure and Motion Estimation

已知x，求K、R、t、X。SLAM比SFM多知道K。离线的SFM一般比在线的SLAM准确。

1. 我们已用5点法求得本质矩阵E，但还有些问题没说：



2. 刚才假设摄像机内参数K已知，实践中如何给定K?通过图片EXIF获取相机焦距、镜头型号等。计算方法：K中f以像素为单位，用焦距的实际长度除以感光元的实际长度再乘以感光元对应的像素个数（下图是以x轴示意，x轴的感光元长度和x轴的像素个数（也就是图像width）之比和y轴是一样的）即可；p_x, p_y理想中位于图像中心，即resolution的一半即可。

Focal length (pixels) = Focal length (mm) x Image width (pixels) / Sensor size (mm)
= 6.0 mm x 600 pixels / 5.75 mm = 626.1 pixels

$$\mathbf{K} = \begin{bmatrix} 626.1 & 0 & 300 \\ 0 & 626.1 & 225 \\ 0 & 0 & 1 \end{bmatrix}$$

3. 在获得K、R、t后，通过三角化计算空间点X。

4. 以给定的K、通过E分解求得的R、t、以及三角化后的X为初始值，迭代求解重投影误差最小化问题：$\min \sum_i ||PX_i - x_i||^2$得到几何误差最小化意义下的K、R、t、X

5. 到目前为止我们完成了两视图的稀疏重建，接下来使用增量式SFM得到多视图的稀疏重建：

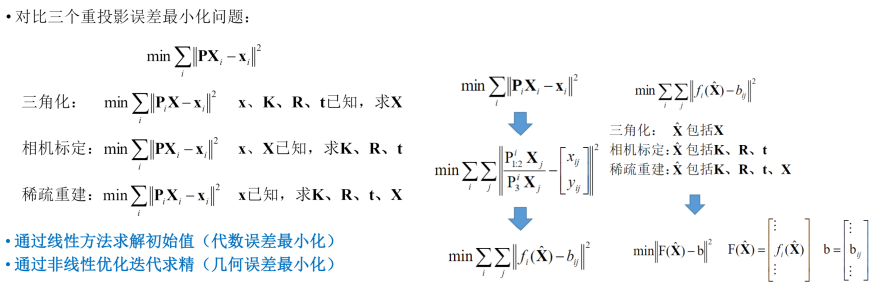
Photo Tourism — 增量式SfM

1. 选择初始图像对
2. 通过两视图SfM计算初始模型
3. 如果图像连接关系图还有未选择的图像，则：
   a. 选择一幅能看到目前模型中最多3D点的图像；
   b. 根据3D-2D点对应估计相机位姿；
   c. 三角化新的特征点Tracks
   d. Bundle adjustment

- 下一步: 在初始模型中添加新图像，同时三角化新的3D点
- 姿态估计(已知x、X、K，求R、t)和三角化(已知x、K、R、t，求X)均可通过DLT求解
- 实际应用中，一次可以添加多幅图像
- 如果剩余图像中单幅图像与模型的最多匹配点数量为N，则匹配点数量多于$0.75N$的图像一次添加

重投影误差最小化问题的求解

为何是非线性的优化问题？原先形式是齐次坐标的，转换到非齐次的时候有个除法。



求解非线性最小二乘（四种不同方法本质区别在于梯度的方向）：

• 重投影误差最小化问题的求解

$$\min\left\|F(\hat{\mathbf{X}})-b\right\|^2$$

重投影误差最小化是一个非线性最小二乘问题，求解非线性最小二乘问题的方法：

- 非线性最小二乘是非线性优化的一类特殊形式；
- 针对一般非线性优化的梯度下降法（1阶）、牛顿法（2阶）；
- 针对非线性最小二乘的高斯牛顿法、LM法（Levenberg-Marquardt）。

• 重投影误差最小化问题的求解

$$\min\left\|F(\hat{\mathbf{X}})-b\right\|^2$$

非线性最小二乘问题迭代优化的基本思路：
1）给定初始值；
2）开始迭代优化
—选择最优移动方向使目标函数值下降最快；
—以一定步长沿最优方向移动当前值；
—如果两次迭代间目标函数值差异小于阈值或迭代次数超出阈值，则转步骤3），否则返回2）；
3）迭代结束，输出当前值。

牛顿法很难用（通常通过扰动的方式求一阶导数（这叫数值求导），二阶导就不好数值求导了）；高斯牛顿法丢到了那个二阶导数，坏处是方向不一定下降，导致越算越差；于是加个阻尼矩阵调整高斯牛顿法。

$$\min\left\|F(\hat{\mathbf{X}})-b\right\|^2$$

Gradient Descent方向为： $\delta(\hat{\mathbf{X}})=-J(\hat{\mathbf{X}})^T r(\hat{\mathbf{X}})$

Newton方向为： $\delta(\hat{\mathbf{X}})=-(J(\hat{\mathbf{X}})^T J(\hat{\mathbf{X}})+S(\hat{\mathbf{X}}))^{-1} J(\hat{\mathbf{X}})^T r(\hat{\mathbf{X}})$

Gauss-Newton方向为： $\delta(\hat{\mathbf{X}})=-(J(\hat{\mathbf{X}})^T J(\hat{\mathbf{X}}))^{-1} J(\hat{\mathbf{X}})^T r(\hat{\mathbf{X}})$

阻尼高斯牛顿方向为： $\delta(\hat{\mathbf{X}})=-(J(\hat{\mathbf{X}})^T J(\hat{\mathbf{X}})+\lambda\mathbf{I})^{-1} J(\hat{\mathbf{X}})^T r(\hat{\mathbf{X}})$

$J(\hat{\mathbf{X}})=\frac{\partial F(\hat{\mathbf{X}})}{\partial \hat{\mathbf{X}}}$ Jacobian矩阵（一阶导数）

$r(\hat{\mathbf{X}})=F(\hat{\mathbf{X}})-b$ 残差向量

$H(\hat{\mathbf{X}})=J(\hat{\mathbf{X}})^T J(\hat{\mathbf{X}})+S(\hat{\mathbf{X}})$ Hessian矩阵（二阶导数）

• 阻尼高斯牛顿法（Damped Gauss-Newton Method）

高斯牛顿法的特点：
- 高斯牛顿法的收敛速度比梯度下降法快的多（超线性收敛，近似二阶收敛）；
- 高斯牛顿法不能保证收敛（梯度下降法和牛顿法收敛）。

阻尼高斯牛顿法：$\delta(\hat{\mathbf{X}},\lambda)=-(J(\hat{\mathbf{X}})^T J(\hat{\mathbf{X}})+\lambda\mathbf{I})^{-1} J(\hat{\mathbf{X}})^T r(\hat{\mathbf{X}})$

$\lim_{\lambda\to 0}\delta(\hat{\mathbf{X}},\lambda)$ 则成为高斯牛顿法

$\lim_{\lambda\to\infty}\delta(\hat{\mathbf{X}},\lambda)$ 则成为梯度下降法

LM法：

• Levenberg-Marquardt Method (LM法)

LM法通过启发式方法在每一步动态调整λ：

$$\delta(\hat{\mathbf{X}},\lambda)=-(J(\hat{\mathbf{X}})^T J(\hat{\mathbf{X}})+\lambda\mathbf{I})^{-1} J(\hat{\mathbf{X}})^T r(\hat{\mathbf{X}})$$

如果误差减少，则令$\lambda\leftarrow 0.1\lambda$

如果误差增大，则令$\lambda\leftarrow 10\lambda$

- LM法是一种启发式的阻尼高斯牛顿法，在计算机视觉中广泛使用。
- 使用LM求解重投影误差最小化的方法称为Bundle Adjustment (捆绑调整，摄影测量中称为光束平差)

3D视觉：三维建模

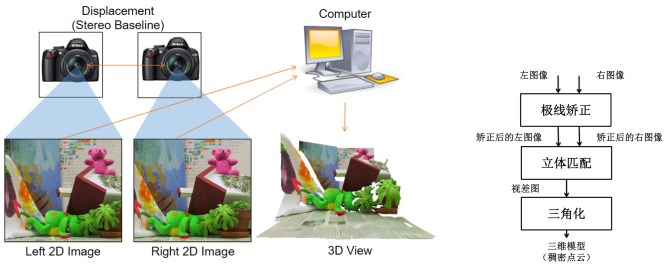
传统的分层三维重建理论：图像->射影空间（不保持线之间的平行和垂直关系）->仿射空间（保持线之间的平行关系，不保持垂直）->度量空间（保持线之间的平行和垂直关系，不保持尺度关系，一般做到这步就行了）->欧式空间（进一步保持尺度关系）。空间之间是子集关系。

在前面的内容中，特征匹配得到的只是整个图片中少部分点，稀疏重建的意思是将匹配的特征点的空间信息还原了，但我们还想还原所有图片的所有像素的空间信息，这就叫做稠密重建。

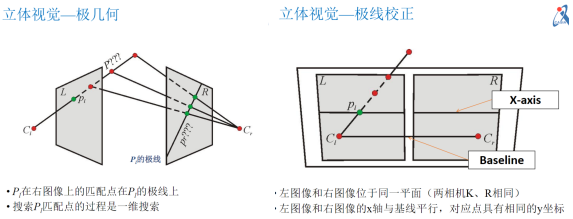
三维建模的过程：已知相机内参、外参（位姿），利用图像信息，重建图像中每一个点的位置信息。得到密集的点云模型。

立体视觉（Stereo）

两视图三维建模：希望左图的每一个点都可以和右图的每一个点对应。



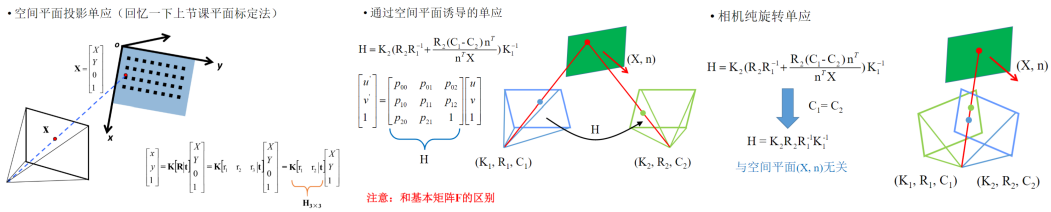
接下来想如何根据左图的点找右图的匹配点：可以在右图对应这个点的极线上找（见下左）；但还是麻烦了点。于是想对右图做一个矫正，让右图的极线穿过左图的点，这样搜索起来就非常快（见下右），这对应的实际场景就是相机内参相同（焦距相同），外参R相同，唯一不同的就是有一个水平移动。



接下来就是找这种矫正的方式，极线校正：利用一个单应变换（只和相机参数相关，见下图）得到一个虚拟相机。

单应变换的几种形式：

- 所有三维点都在一个二维平面上，图上的投影点和三维平面上的点满足的小孔成像关系是一个单应变换的关系（3 * 3的矩阵）；
- 两个相机，图像中所有对应的三维点都在同个平面上，那么空间平面上的点在两个图像上都会有成像，这些两图像的成像点满足一个单应变换的关系（由相机内外参、空间平面的参数（6维向量，(X,n)，X是平面上的点，n是法向量即可定义一个平面）决定），因为单应变换和平面相关，所以叫空间平面诱导的单应。因为H是点对点的对应关系，比基本矩阵F这种点和极线的关系更强。
- 两个相机的光心重合，那么这两个相机的对应点同样满足单应变换关系。可以应用于全景拍摄，实时计算。



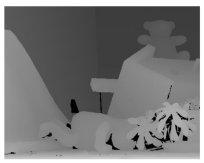
视差（Disparity）与深度（Depth）

人的经验：左右眼分别看同一个位置，跳变大就是近处，跳变远就是远处，无跳变就是无穷远处。

极线校正后，左右图像的一对匹配点在x轴上坐标的差异称为视差（Disparity）；视差的大小与点距离相机距离的远近成反比；从视差可以直接计算深度（深度等于相机光心距离乘以焦距除以视差）。

立体视觉的目的就是通过左右图像计算（稠密）视差图：视差图是一幅灰度图像，像素点的值表示这一点的视差大小，灰度值越高表示视差越大（距离越近）；通过视差图可以得到单视点下（有很多空缺，要多幅图像才能建得比较完整）的三维模型（稠密点云）。

立体视觉—视差图



•通过视差图可以得到单视点下的三维模型（稠密点云）

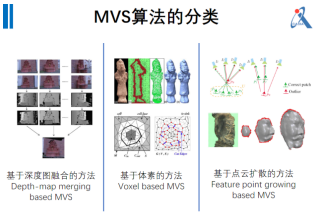
现在我们思考立体匹配：对于左图像上每一点p，在右图极线上寻找对应点（注意经过极线校正后，点p在极线上）。方法：

- 单点匹配法：极线上颜色最相似的点作为对应点
- 窗口匹配法：通过图像上一个正方形窗口区域衡量匹配程度；
- 窗口匹配法中窗口尺寸的选择： 大尺寸窗口有利于解决：弱纹理（窗口内像素值类似）；孔径问题（窗口包含的纹理区域太小，区分度不足）；重复纹理（重复纹理区域窗口内像素值类似，容易产生匹配误差）； 小尺寸窗口有利于解决：前景放大（前景放大效应造成视差图中前景物体比实际大）； 窗口尺寸选择依据：没有最优窗口尺寸能够解决所有问题。 改进：自适应权值窗口匹配法。
- PatchMatch Stereo（速度快，工业软件的baseline，2010）：1. 随机生成每个像素点的深度和法向 2. 从左上向右下传播：1）检测邻域点的深度和法向是否更好（通过1个点算1个单应） 2）检测自身随机扰动后的点是否更好 3. 从右下向左上再传播一次

多视图立体重建 （Multiple View Stereo,MVS）

上面是双视图，现在考虑多视图。

MVS的基本思路：寻找空间中具有图像一致性(Photo-consistency)的点。



基于深度图融合的MVS（双目立体视觉到多目立体视觉的直接推广）：

基于体素（认为空间是一个个小的立方体，也就是体素构成的）的MVS：此时MVS等价为一个3D空间Voxel的标记(Labeling)问题，离散空间的Labeling是一个典型的MRF(Markov Random Field)优化问题，可以用Graph-cuts求解。Pros：生成规则点云；易于提取Mesh (Marching cube algorithm)。Cons：精度取决于voxel粒度；难以处理大场景。

还有基于特征点扩散的MVS，基于CNN的MVS。

3D视觉：三维表达与语义重建

点云网格建模：获取封闭完整的三角网格模型

Shape from X：从X获得3D点云。 Shape (3D Modeling) from SfM+MVS：从明暗恢复形状（Shape from Shading）；从光度立体恢复形状（Shape from Photometric Stereo）；从纹理恢复形状（Shape from Texture）；从焦点恢复形状（Shape from Focus）

相比于稠密点云模型，三维网格模型（3D Mesh）是一种更加精简和紧致的表达方式，同时也可以实现对有噪、有缺失点云模型的进一步精简化和完整化，是目前在渲染、浏览、存储、碰撞检测等领域最常用的三维表达方式。

点云网格化旨在将有噪、有缺失的稠密点云转换为封闭的紧致三角网格模型。

方法：Poisson Surface Reconstruction(2006)，基于Delaunay和MRF的点云网格化(2011)，分布式点云网格化(2017)，Neural Radiance Fields (NeRF,2020)。

三维语义建模：获取三维点/三角面片的语义类别属性

输入：网格/点云、多视角图像 输出：网格/点云语义分割 主要难点：1）海量三角面片/点云的细粒度语义分割 2）二维语义分割与三维模型的一致性融合

方法：基于几何特征；基于模板匹配；端到端分割；二维图像分割的三维融合

三维矢量建模：获取小体积、紧致、规范的矢量化表达

输入：单体部件网格、多视角图像 输出：矢量三维模型 主要难点：1）图像建模网格中的外点、噪声、空洞 2）矢量模型的规范和精细度

方法：网格简化(2004)，网格平面保持简化(2015)，点云平面拟合(2017)。

总结

三维视觉系统应用:

- AR三维定位地图
- 城市实景三维建模
- Tesla Autopilot

三维视觉的科学思想：

1. 层次化处理、问题分解、降维
2. 清晰数学（几何）解释
3. 解析解、全局解、高效局部解

中层**2D**视觉：图像分割

图像分割：把图像分成互不重叠的区域并提取出感兴趣目标的技术。

分类：基本的图像分割（把图像分成不同部分）；语义分割（semantic segmentation）：把每个分割出的物体标注出类别；实例分割（instance segmentation）：把同个类别的所有个体也标注出来。

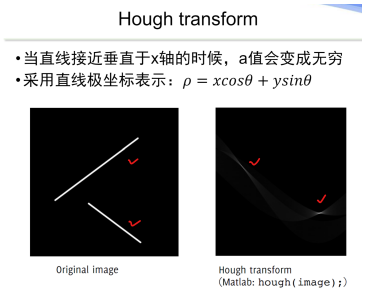
图像分割大的基本依据：1.区域内的一致性2.区域间的不一致性。

早期的图像分割方法

阈值法：通过设定不同的特征阈值，把图像像素点分为若干类．常用的特征包括：灰度、彩色特征、由原始灰度或彩色值变换得到的特征。改进——局部阈值法：将图像分块，分别用全局阈值方法分割，最后再综合。

基于边缘的分割方法：检测边缘，然后根据边缘将图像分割成不同的区域。检测边缘可以用前面的微分算子，但需要人工确定图像的特点采用不同的算法。而Hough transform(1959)便是检测直线的一种方法（也可以检测其他参数化的物体，比如圆或者椭圆等）。由图像空间转到参数空间，在参数空间进行投票找到原图像空间的特殊结构。

检测直线的基本思想是通过某一个点的直线的参数(a,b)集合在参数空间中形成一条直线。若参数空间中两条直线存在交点，则这个交点就对应着 分别确定两条直线的点 相连的直线。假设每个点都有直线通过，那么图像中所有点对应在参数空间形成的焦点（多条直线相交）就对应着图像中的直线。



基于特定理论的方法

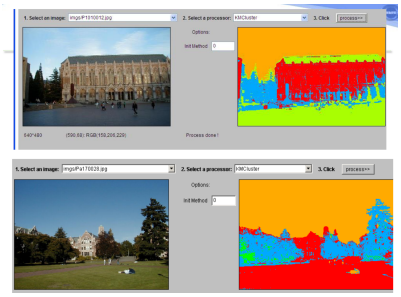
聚类

思想：给定一组数据点，使用聚类算法将每个数据点到分类到图像中的特定组中。理论上，同一组中的数据点应具有相似的属性和特征，而不同组中的数据点的属性和特征则应不同。

K-Means

优点：速度快，时间复杂度O(n) 缺点：初始化——如何一开始就确定聚类个数K；选取质心的方式：一开始质心点的选取是随机的，质心点的位置不可重复且缺乏一致性。

聚类出来的区域是圆形的。



GMM

K-Means的改进：类的形状可以是椭圆了；点的类别不再属于单个类，而是以概率的方式属于多个类。但和K-Means相比，GMM每一步迭代的计算量比较大；基于EM算法，有可能陷入局部极值，需要经过多次迭代；依然需要指定聚类的个数。

GMM

- 高斯混合模型（GMM）比K-Means算法具有更好的灵活性。它是多个高斯分布函数的线性组合，理论上可以拟合出任意类型的分布
- 对于GMM，我们假设数据点满足不同参数下的高斯分布
- 用两个参数来描述聚类的形状：均值和标准差。以二维分布为例，标准差的存在允许聚类的形状可以是任何种类的椭圆形，不再局限于圆形(K-Means)
- 如果数据点符合某个高斯分布，那它就会被归类为那个聚类

- 为了找到每个聚类的高斯参数，要用到期望最大化（EM）的优化算法
- EM（Expectation-Maximum）算法也称期望最大化算法，曾入选“数据挖掘十大算法”
  - Wu X, Kumar V, Quinlan J R, et al. Top 10 algorithms in data mining[J]. Knowledge and information systems, 2008, 14(1): 1-37.
- EM算法是一种迭代优化策略，由于它的计算方法中每一次迭代都分两步，其中一个为期望步（E步），另一个为极大步（M步），所以算法被称为EM算法（Expectation-Maximization Algorithm）

- 基本思想是：首先根据已经给出的观测数据，估计出模型参数的值；然后再依据上一步估计出的参数值估计缺失数据的值，再根据估计出的缺失数据加上之前已经观测到的数据重新再对参数值进行估计，然后反复迭代，直至最后收敛，迭代结束。

K-Means -> EM

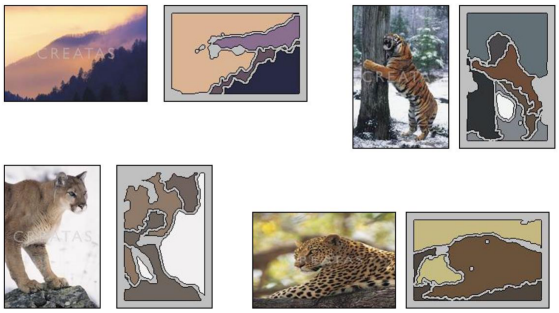
- Boot Step:
  - Initialize K clusters: $C_1, \ldots, C_K$

    (μ_j, Σ_j) and $P(C_j)$ for each cluster j.
- Iteration Step:
  - Expectation Step　Estimate the cluster of each data point

    $$p(C_j \mid x_i) = \frac{p(x_i \mid C_j) \cdot p(C_j)}{p(x_i)} = \frac{p(x_i \mid C_j) \cdot p(C_j)}{\sum_j p(x_i \mid C_j) \cdot p(C_j)}$$
  - Maximization Step

    $$\mu_j = \frac{\sum_i p(C_j \mid x_i) \cdot x_i}{\sum_i p(C_j \mid x_i)} \quad \Sigma_j = \frac{\sum_i p(C_j \mid x_i) \cdot (x_i - \mu_j) \cdot (x_i - \mu_j)^T}{\sum_i p(C_j \mid x_i)} \quad p(C_j) = \frac{\sum_i p(C_j \mid x_i)}{N}$$

  Re-estimate the cluster parameters



Mean-Shift

动机：不需要指定聚类的个数。

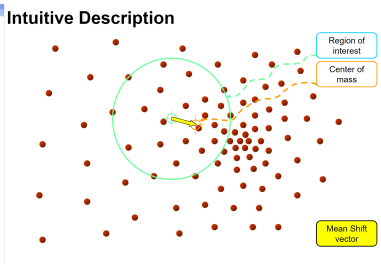
评价：无需指定聚类数目K，聚类中心处于最高密度处，也符合直觉认知；缺点是如何正确选择窗口尺寸h。

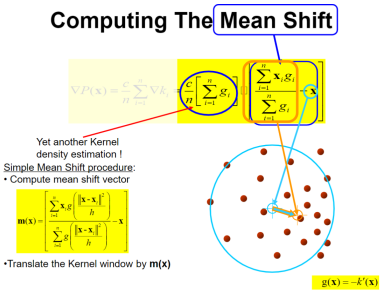
思想：由圆心移动到圆围住区域的质心，直到收敛。

计算方法： Kernel Density Gradient Estimation（核密度估计是Non-Parametric）。去估计核函数梯度的方向，也就是Mean Shift 方向，就是我们移动的方向。注意第三张图的梯度是标量乘以向量，我们关注向量即可。

Mean-Shift

- ■1975年，Fukunaga和Hostetle提出了一种基于一般核函数的非参数密度梯度的估计算法，并给出了保证估计值与真实值之间渐近无偏、一致和均匀连续时核函数应满足的条件。
- ■1999年，Comaniciu将均值移动应用于图像分析。
- ■核心思想:找到概率密度梯度为零的采样点（确定密度函数的最大值），并以此作为特征空间聚类的模式点.





Graph Cut

基本思想：1.将图像用图的方式表示，顶点表示像素，边表示像素之间的关系。图像分割对应图的割集。2.确定图中边的权值，使图像分割目标（能量最小化）对应图的最小割。3.用最大流算法求解最小割问题。

Normalized cut用于图像分割中的应用

具体算法：1. 由图像变成全连接的有权图，权值衡量像素点相似程度 2. 应用Normalized cut的近似算法（求最小割的一种变体）：每次将点集二分类；分类好的点集上继续二分类。

(4) Normalized cut 在图像分割中应用

将一幅图像上所有像素点看作点集V，每两个点之间都建立一条边，得到边集E。为每条边按下面方法赋权

$$w_{ij} = e^{-\frac{||F_i - F_j||^2}{\sigma_I^2}} \times \begin{cases} e^{-\frac{||X_i - X_j||^2}{\sigma_X^2}}, & if \ ||X_i - X_j|| < r \\ 0, & else \end{cases}$$

F是彩色或灰度向量

这样就建立一个赋权无向图$G=(V, E)$

按照前述算法，我们就可以完成对该幅图像分割操作。

具体算法

① 给定一个点集，构建图G(V, E)，边的权为对应两端点的相似度。
② 求解$(D - W)x = \lambda Dx$的特征值及其所对应的特征向量。
③ 用次小特征值所对应的特征向量进行二分类。
④ 若需再分，则在每个分好的类别中重复上述过程。否则终止。

原理：

1. 从图像的分割变到图的分割：

一般的分类问题：
- ■给定一个点集V，按照一定的相似度量（距离）寻求一种划分，将点集V划分成不相交的若干子集合$V_1, V_2 \ldots V_m$。使得每一子集内部的相似度尽量高，而子集之间的相似度尽量低。

两个问题：
- ■ 什么是最优划分准则？
- ■ 有没有有效算法？

用图论的方法来解决分类问题：
- ■在给定点集V中的每一点对i、j之间，建立一条边（i, j），给这条边赋权w_{ij}相似度量。这样就建立了一个无向赋权图。

对于这样的图我们可以用邻接矩阵来表示:

$$\begin{bmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{bmatrix}$$

2. 考虑最简单的二分类情形，以及直接用最小割算法的问题：

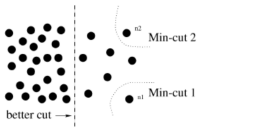
考虑二分类问题

将点集V分成不相交的两部分A、B。则两类别之间的相似性我们可以用图割来度量。

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v)$$

显然最优划分应使$cut(A, B)$取最小值。

需要注意的是：仅考虑用割集的权值之和来度量两个集合之间的相关性，会容易出现孤立分割的问题。如下图所示：



3. 采用改进的最小割度量方式：Normalized cut（为了避免分割出孤立点，考虑分割的两个子集合和完整集合的相似程度）

(2) Normalized cut

一个解决上述问题的办法是通过定义新的类间相似性度量。

Normalized cut(*Ncut*):

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)}$$

这里 $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$

这样包含孤立点的*Ncut*值不会小。

再定义总的类内相似性度量

$$Nassoc(A, B) = \frac{assoc(A, A)}{assoc(A, V)} + \frac{assoc(B, B)}{assoc(B, V)}$$

(2) Normalized cut

通过简单的推导可以证明

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} = 2 - Nassoc(A, B)$$

可见最小化类间相似性和最大化总类内相似性是等价的。

这样就解决了划分准则的问题，即

最优划分对应于最小*Ncut*

通过求最小*Ncut*，就可以得到最优划分。

下一个问题是有没有求最小*Ncut*的有效算法？

4. 为提高效率采用近似算法（具体见论文，有意思的结果），还有将二分类推广到多分类：

(3) 求解最小*Ncut*的近似算法

不幸的是求一个图的*Ncut*是一个NP问题。

但是通过精巧的构造，可以通过求解如下广义特征值问题来得到最小Normalized cut近似解。

令$W = \{w_{ij}\}$

$D = diag(d(1), d(2), \ldots, d(N)), d(i) = \sum_j w_{ij}$

则广义特征值问题 $(D - W)x = \lambda Dx$的次小特征值是最小*Ncut*对应的实数解。该特征值所对应的特征向量对应于最优划分。

(3) 求解最小*Ncut*的近似算法

- 特征向量离散化

由于我们需要特征向量是仅含有不同符号的两个值，故还需要对所得特征向量做离散化处理。即需要选择一个分界点。有两种方法：

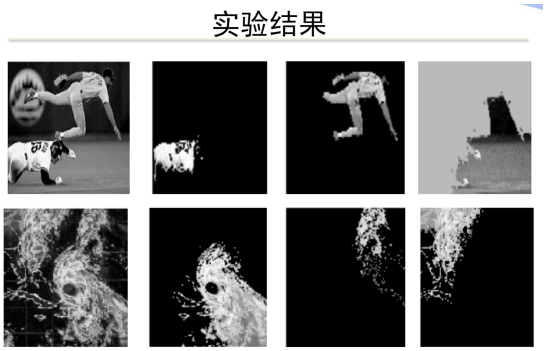
（1）取中点

（2）取0

- 多分类问题

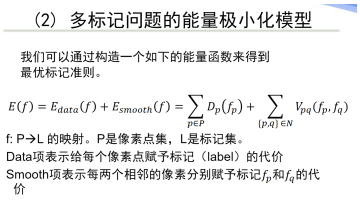
首先用次小特征值所对应特征向量进行二分类。然后用再次小特征值所对应的特征向量对已分好的两类再次细分。或者在每个分好的类别中，重复用上述算法进行分类。

效果：



graph cut算法求解计算机视觉中的能量极小化问题

计算机视觉的很多问题可以看作一个给图像每个像素打上最优标记的问题。 我们可以将最优标记对应到能量函数最小化的问题（多标记指的是标记的数量有多个，而每个像素只能有一个标记）。



可以运用Graph cut算法求解能量极小化问题。

多标记问题：

当标记数量大于2时，已经证明该问题是NP-hard问题。故很难求得该问题的全局极小值。

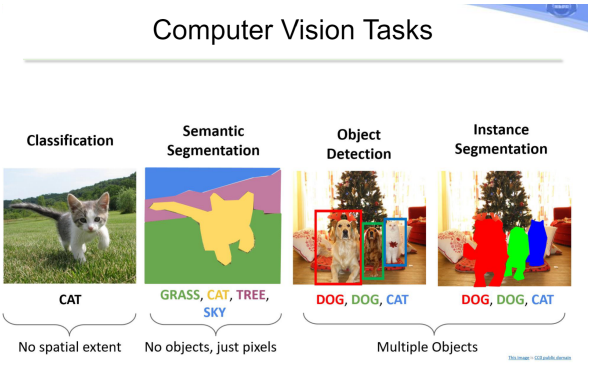
Boykov等构造了两个运用最小割求解该类能量函数的近似极小值的算法：

α-β swap

αexpansion

这两个算法运算速度快，且能得到比较好的结果，从而得到了广泛应用，并使得用能量极小化模型和图割来处理计算机视觉中的一些问题成为目前的一个研究热点。

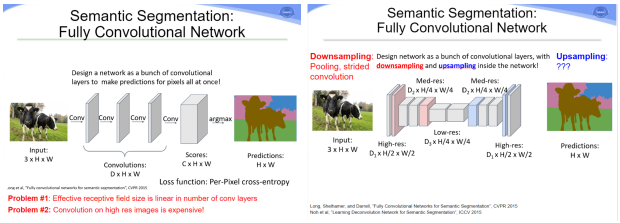
深度神经网络完成图像任务



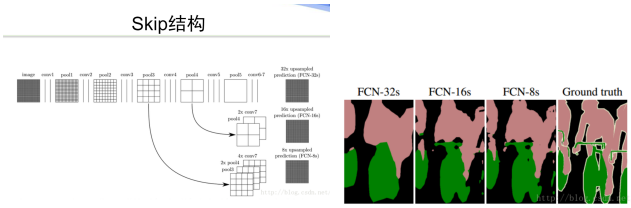
Semantic Segmentation(语义分割**)**

图像的每个像素都分给了一个类别。

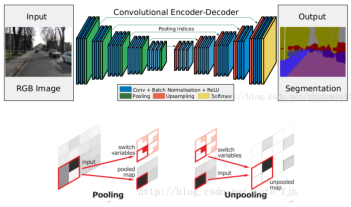
1. Fully Convolutional Network(2015): 输出为每个像素属于不同类别的概率；左图问题：卷积层数多、在大图像的计算量大；解决方法见右图：下采样，但为了恢复图像大小于是要引入上采样（Bilinear Interpolation，Max Unpooling）或者反卷积。FCN采用的是反卷积而不是Max Unpooling。



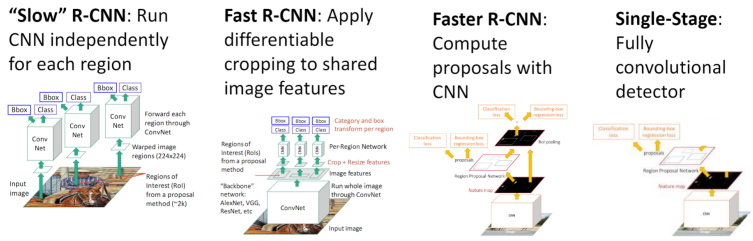
还采用了Skip结构融合网络不同层数的信息。



2. SegNet(2017): 采用Max Unpooling做上采样。效果比FCN更好。



Object Detection(目标检测)

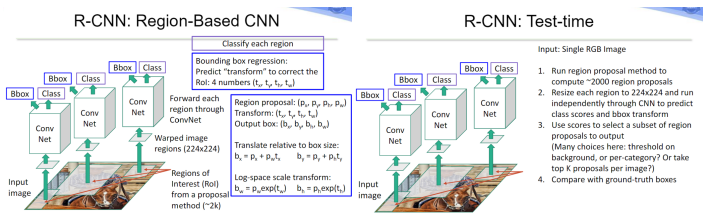


输出是物体的类别和位置（Bounding box 描述(four numbers: x, y, width, height)）。

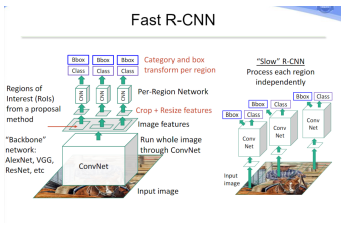
难点：每个图像要输出可变数量的物体，还要输出物体的位置。

解决检测可变数量的物体的方法：滑动窗口，每个窗口进行单个物体检测；问题：遍历所有位置、遍历所有框的大小。改进：Region Proposals：利用图像中的纹理、边缘、颜色等信息，找出可能含有物体的区域。

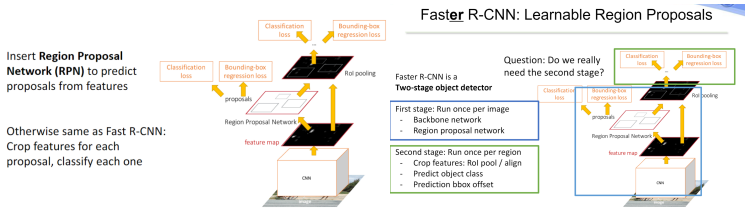
1. R-CNN(2014): 根据Region Proposals检测物体，输出物体类别和更准确的物体位置（用IoU(Intersection over Union)衡量输出的预测的框和标定的框hi见的重合程度，交集比上并集）。



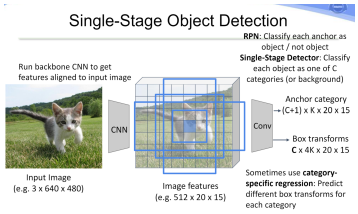
2. Fast R-CNN(2015): 上面的R-CNN的Problem: Very slow! Need to do ~2k forward passes for each image! Solution:Run CNN before warping! 还有Cropping Features的操作：如何在卷积后的区域进行Region Proposals，然后统一到相同维度。结果比R-CNN快10倍。



3. FasterR-CNN(2015): Learnable Region Proposals，在Fast R-CNN的基础上插入一个RPN网络去完成Region Proposals的操作。结果比Fast R-CNN快10倍。改进：这三个的都是2stage的（第一步输出哪儿有框，第二步输出框里面啥物体，更精细的框），可以做1stage的（直接输出物体类别还是背景）。

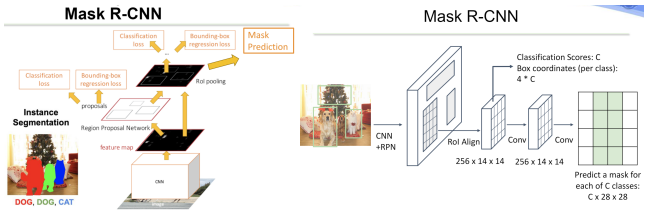


4. Single-Stage Object Detection模型：精度没那么高，但是速度快。



Instance Segmentation(实例分割)

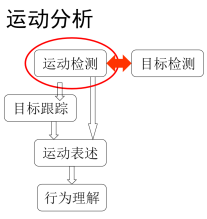
Mask R-CNN：在小框上的每个像素上再输出一个mask，标志其是不是物体。Very Good Results



高层2D视觉：检测

运动分析：研究运动的物体。

什么是运动分析？就是在不需要人为干预的情况下，综合利用计算机视觉、模式识别、图像处理、人工智能等诸多方面的知识对摄像机拍录的图像序列进行自动分析，实现对动态场景中人的定位、跟踪和识别，并在此基础上分析和判断人的行为。



运动检测

运动检测(Motion detection)定义？将运动前景从图像序列中提取出来，也就是说将背景与运动前景分离开。

难点：受天气、光照、阴影等诸多外界因素以及背景物体内在因素的影响；图像中的背景常常是动态变化的（人脸检测系统一半做得比较好也有背景固定的原因）。

两种主要思路：1. 直接利用前景所特有的信息检测前景（适合一些场景的背景有较大变化，而前景的一些特征变化不大。）；2. 先得到背景图象，然后将输入图象减去背景图像从而得到前景图象（适合一些场景的背景相对固定，而前景变化较大的情况）。

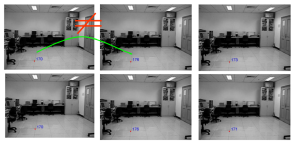
背景差法（**background subtraction**）：混合高斯模型

原理：计算当前图像与背景图像的逐像素的灰度差，再通过设置阈值来确定运动前景区域。

首先要得到背景图像：将若干背景图像（显然这些背景图像要对齐，应该是同个相机连续拍的）求逐点的灰度均值。求均值的原因是多个图像中同个位置的像素点的灰度值都会变！见下图。求均值是一种方式，改进：用高斯模型来假设每个像素点在不同时刻的灰度分布情况。

均值图像

- 为什么要将若干背景图像的均值图像作为背景，而不是直接从中选一张作为背景呢？



单高斯模型

什么是单高斯模型？对于每一个像素，用一个高斯分布来描述其在不同时刻的灰度分布情况的背景模型。也就是单个像素的均值是多少，容忍的变化程度是多少。因此有多少个像素就要有多少个高斯模型。

使用高斯分布来描述背景的假设？背景在图像序列中总是最经常被观测到。

不足：背景往往不是绝对静止的，而是时常变化的！尤其对于室外场景。

混合高斯模型

想法：用多个高斯分布（混合高斯分布）来描述单个像素在不同时刻的灰度。 为何用多个高斯分布？充分多的高斯分布可以拟合很多其他分布，生活中常见的分布可以由多个高斯分布来拟合。

方法：随着视频帧的输入对每个像素逐步建立高斯模型，建立的条件是已有的高斯模型无法刻画像素值的变化（当达到K个的阈值时，就用新的高斯模型代替最差的旧的高斯模型）；否则只更新已有高斯模型的参数。

混合高斯模型

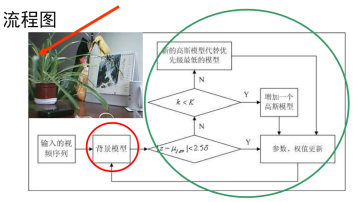
- 设用来描述每个像素的高斯分布共K个（K通常取3－7个），像素z_{uv}的概率函数：

$$P(z_{uv}) = \sum_{j=1}^{K} \omega_{j,uv} N(z_{uv}|\mu_{j,uv}, \Sigma_{j,uv})$$

其中$\omega_{j,uv}$是第j个高斯分布权重

$$N(z|\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}\Sigma^{1/2}} e^{-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)}$$

流程图



背景建模过程（仅针对单个像素）：

1. 初始化：用第一幅图像该点的像素值作为均值，给定一个较大的方差和较小的权值。
2. 模型学习

$\omega_{j,w,t} = (1-\alpha)\omega_{j,w,t-1} + \alpha(M_{j,w,t-1})$, $M_{j,w,t-1}$是个0/1函数

$$\mu_{j,w,t} = (1-\rho)\mu_{j,w,t-1} + \rho z_{j,w,t}$$

$$\sigma_{j,w,t}{}^2 = (1-\rho)\sigma_{j,w,t-1}{}^2 + \rho(z_{j,w,t} - \mu_{j,w,t})^T(z_{j,w,t} - \mu_{j,w,t})$$

$$\rho = \alpha N(z_{j,w,t}|\mu_{j,w,t}, \Sigma_{j,w,t})$$

高斯分布的权重$w_{j,uv}$和优先级$p_{j,uv}$是两个量！最终每个像素使用的背景模型不是K个，而是B个！

- 这K个高斯模型是否应该都用上呢？
  - 由于噪声的影响或前景物体的存在，某些像素值并不能代表背景，因此由这些像素值构造的高斯分布应该去掉。
- 定义各个高斯分布的优先级：

$$p_{j,uv} = \omega_{j,uv}/\sigma_{j,uv}$$

- 如何去掉多余的高斯分布？
  - 前面提到的假设：前景和噪声不会在同一位置太长时间，这样，前景和噪声对应的高斯模型的权值和优先级都比较小，因此可以将K个高斯分布按优先级由高到低排列，用如下策略选取前 B 个分布作为背景模型：
- B的定义：

$$B = \min_b(\sum_{j=1}^{b} \omega_{j,uv} > M)$$

其中M为预设的阈值。（如果M较小，则为单高斯模型）

构建完背景的混合高斯模型后，真正的前景检测：只要当前像素点满足对应的混合高斯模型中的任何一个，都认为它是背景；否则才认为是前景。

前景检测：将待测图像的每一个像素点与该像素点对应的混合高斯模型的各个模型分别进行比较，若有 $|z-\mu_{j,uv}|<a\sigma$（a为一常数），则该点属于背景，否则属于前景。

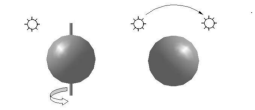
光流（**optical flow**）

光流的定义：当人的眼睛观察运动物体时，物体的景象在人眼的视网膜上形成一系列连续变化的图像，这一系列连续变化的信息不断“流过”视网膜（即图像平面），好像一种光的“流”，故称之为光流（optical flow）。光流是空间运动物体在观测成像面上的像素运动的瞬时速度。光流的研究是利用图像序列中的像素强度数据的时域变化和相关性来确定各自像素位置的“运动”。

点的光流和点的实际运动有什么区别和联系呢？

- 运动场（motion field）：一个运动物体在空间产生一个三维的速度场，运动前后空间对应点在图像上的投影形成一个二维运动场。要确定运动场，得确定三维速度场或者三维结构，运算效率低。
- 光流场（optical flow field）：是指图像亮度模式的表观（或视在）运动，是二维矢量场。它包含的信息即是各像点的瞬时运动速度矢量信息。

为什么要研究光流呢？事实上，仅仅通过图像序列很难计算出物体的空间位置进而得到真实的运动场。而光流表达了图像的变化，包含了目标一定的运动信息，通过计算光流场可以从图像中近似计算不能直接得到的运动场。但也有不准确的情况：



光流等于零，但运动场却不等于零。　光流不等于零，但运动场为零。

如何计算光流呢？1981年，Horn和Schunck创造性地将二维速度场与灰度相联系，引入光流约束方程，得到光流计算的基本算法：

光流约束方程

光流一致性假设（Brightness constancy assumption）

$$I(x+u\Delta t, y+v\Delta t, t+\Delta t)=I(x,y,t)$$

方程左侧按Taylor级数展开，化简：

$$\frac{\partial I}{\partial x}\frac{\partial x}{\partial t}+\frac{\partial I}{\partial y}\frac{\partial y}{\partial t}+\frac{\partial I}{\partial t}=0$$

令 $u=\frac{\partial x}{\partial t}, v=\frac{\partial y}{\partial t}, I_x=\frac{\partial I}{\partial x}, I_y=\frac{\partial I}{\partial y}, I_t=\frac{\partial I}{\partial t}$

则方程 $\frac{\partial I}{\partial x}\frac{\partial x}{\partial t}+\frac{\partial I}{\partial y}\frac{\partial y}{\partial t}+\frac{\partial I}{\partial t}=0$

化为如下光流约束方程：

$$I_x u+I_y v+I_t=0$$

由光流约束方程可见：该方程两个未知数u和v。因此，只使用一个点上的信息是不能确定光流的。人们将这种不确定问题称为孔径问题(aperture problem)。基本的解决思路：添加其它的约束。Lucas-Kanade添加了局部平滑的约束：从上面的对单个点的光流约束变到局部点集的光流约束。

基于梯度的方法——Lucas-Kanade方法

通常附加的约束是：假设光流场是局部平滑的。

Lucas-Kanade方法[3]的约束则更强一些：假设每个像素领域内的像素具有相对的速度。

$$E(u,v)=\sum_{(x,y)\in\Omega}(I_x(x,y)u+I_y(x,y)v+I_t)^2$$

对其两边求偏导：

$$\frac{\partial E(u,v)}{\partial u}=\sum_{(x,y)\in\Omega}2I_x(I_x(x,y)u+I_y(x,y)v+I_t)=0$$

$$\frac{\partial E(u,v)}{\partial v}=\sum_{(x,y)\in\Omega}2I_y(I_x(x,y)u+I_y(x,y)v+I_t)=0$$

$$\Rightarrow \begin{bmatrix}\Sigma I_x{}^2 & \Sigma I_x I_y\\ \Sigma I_x I_y & \Sigma I_y{}^2\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}=-\begin{bmatrix}\Sigma I_x I_t\\ \Sigma I_y I_t\end{bmatrix}$$

$$\Rightarrow (\Sigma\nabla I\nabla I^T)\vec{U}=-\Sigma\nabla I I_t$$

Lucas-Kanade方法

令 $A=\Sigma\nabla I\nabla I^T, \vec{U}=[u,v]^T, b=-\Sigma\nabla I I_t$

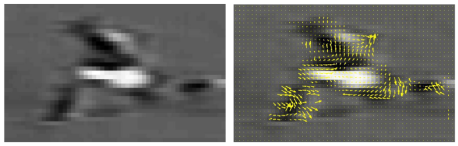
则有：

$$A\vec{U}=b$$

此时，原问题转化为一个最小二乘问题。

注意，考虑可解性！

效果（可以对单个图像找到光流场）：

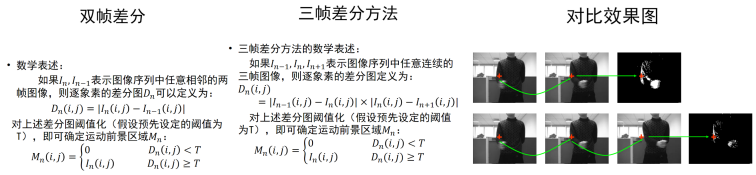


光流法的优点：直观的表述运动模式；不易受运动物体外表的干扰

光流法的不足：计算量比较大（每个点都要算光流）；易产生较大噪声（完全依赖于图像灰度）

帧间差分（**frame differencing**）

原理：在相邻两帧（也可以为多帧）间计算逐像素的灰度差，并通过设置阈值来确定对应运动前景的像素，进而得到运动前景区域。



优点：适用于动态变化的背景环境（因为直接做的是相邻两帧的差异）。简单直接，工业应用中广泛使用。
不足：较难准确检测运动速度过快的物体；较难准确检测运动速度过慢的物体；较难准确检测场景中同时存在的多个运动物体

目标检测

目标检测：自动确定目标在图像中的位置和类别。

AdaBoost

AdaBoost（adaptiveboosting）Boosting算法中的一种变形。

动机：就是根据已有的样本，融合多个弱分类器形成一个整体的强分类器。

发展历史：

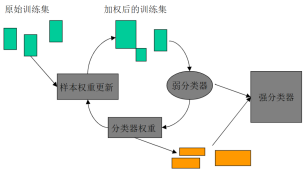
- Proved the fact: learners, each performing only slightly better than random, can be combined to form an arbitrarily good ensemble hypothesis. (Kearns and Valiant 1989)
- Boost-by-majority algorithm (Freund 1990)
- AdaBoost(Freund and Schapire1994-1996)
- Generalized version of AdaBoost(Schapireand Singer 1997)
- AdaBoost in face detection (Viola and Jones 2001)

AdaBoost具体解决两个问题：怎样处理训练样本？怎样合并弱分类器成为一个强分类器？

注意：

- 每个样本都参与到单个分类器的训练过程中，有多个分类器（左图黄色部分表示3个分类器）
- $\forall \alpha_t, e^{\alpha_t} > 1, e^{-\alpha_t} < 1$，降低分类成功样本的权重，提高分类是被样本的权重。
- 每一轮只选择1个分类器。选择方法见右图。

AdaBoost法则



输入：m个样本$(x_1, y_1), \ldots, (x_m, y_m)$. 其中$x_i \in X, y_i \in Y = \{-1, 1\}$

初始化样本权重 $D_{t+1}(i) = 1/m$

For $t = 1$ to T

1. 找到弱分类器h_t ，并得到其分类误差ε_t
2. 计算分类器权重$\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$
3. 更新样本权重

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & if\ h_t(x_i) = y_i \\ e^{\alpha_t} & if\ h_t(x_i) \neq y_i \end{cases}$$

输出：

$$H(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$$

ε_t 越大， α_t 越小

Z_t 是归一化因子

怎样获得每次循环的弱分类器？

1. $h_t = \text{argmin}_{h_j}\{\varepsilon_j = \sum_{i=1}^{m} D_t(i)\,[y_i \neq h_j(x_i)]\}$
2. 先决条件：$\varepsilon_t < 1/2$ ，否则终止循环

弱分类器的例子：如决策树等等。

实例：用AdaBoost处理人脸检测

基于**DNN**的物体检测

RCNN系列，两阶段，见前面内容。

YOLO系列(You Only Look Once: Unified, Real-Time Object Detection, 最早版本2016)，单阶段，将object detection 的框架设计为regression problem，直接从图像像素到bounding box 以及probabilities。

下面的S是人为设定的，B表示每一个$S * S$大小的grid可以含有几个框（5维向量表示），也是认为设定的。

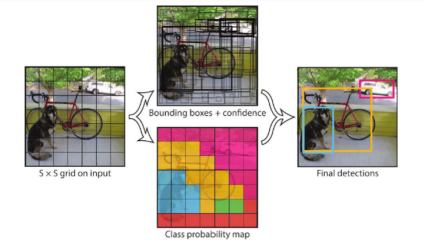
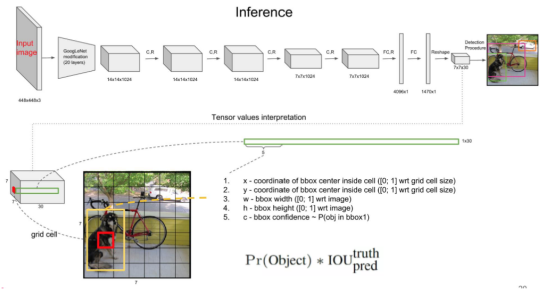


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

图像输入为448x448，取S=7，B=2，一共有20个类别(C=20)。则输出就是7x7x30的一个tensor：



DETR(End-to-End Object Detection with Transformer,2020)

高层2D视觉：跟踪

- ❑ 目标跟踪
  - ❑ 模板匹配法
  - ❑ 基于Kalman滤波器的跟踪方法
  - ❑ 基于相关滤波的跟踪方法
  - ❑ 基于CNN的跟踪方法
- ❑ 视觉定位
  - ❑ 基于Kalman滤波器的定位方法
  - ❑ 基于关键帧的定位方法

跟踪的分类：

- 目标跟踪：在图像序列中持续地估计出感兴趣的运动目标所在区域（位置），形成运动目标的运动轨迹；有时还需要估计出运动目标的某些运动参数（比如速度、加速度等）。
- 相机跟踪（摄像机定位）：通过图像序列，持续地计算出相机的位置、姿态，如SLAM（Simultaneous Localization And Mapping，同步定位与地图创建）。

目标跟踪

目标跟踪问题分类：

- 场景中运动目标的数目：单运动目标vs. 多运动目标 在多目标跟踪过程中，必须考虑到多个目标在场景中会互相遮挡(Occlusion)，合并(Merge)，分离(Split)等情况。 多目标跟踪中的存在数据关联问题(Data Association)。
- 摄像机的数目：单摄像机vs. 多摄像机 多摄像机有望解决因相互遮挡导致的运动目标丢失问题，但多摄像机的信息融合是一个关键性问题。
- 摄像机是否运动：摄像机静止vs. 摄像机运动
- 有多个摄像机，拍摄的场景是否重合。
- 场景中运动目标的类型：刚体vs. 非刚体 交通车辆－刚体；人－非刚体。
- 传感器的种类：可见光图像vs. 红外图像 白天使用可见光图像；晚上使用红外图像。

目标跟踪的应用

- 行为分析：自动跟踪犯罪分子
- 虚拟现实：Vtuber
- 增强现实：抖音特效

运动目标的表示方法

跟踪物体的特定结构：

- 基于点的跟踪：物体的质心或一组特征点集
- 基于区域的跟踪：将运动目标用比较简单的几何形状表示，比如矩形或椭圆等；适合于表示简单的（刚体或非刚体）运动目标。相较于后面要介绍的活动轮廓等表示方法精度较差
- 基于轮廓的跟踪：Contour表示运动目标的外部轮廓；Silhouette表示运动目标外部轮廓内的区域；适用于表示复杂的非刚体运动目标；主动轮廓（Active Contour）利用封闭的曲线轮廓来表示运动目标，并且该轮廓能够自动连续地更新
- 基于模型的跟踪：为物体建立二维形状模型或者立体模型

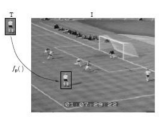
依次由简到繁，采用上述的哪种方法来表示运动目标和 不同的应用场合、运动目标的运动特性、以及对跟踪算法的精度要求等密切相关。

传统目标跟踪方法

自底向上(Bottom-up)的处理方法：数据驱动(Date-driven)的方法，不依赖于先验知识，比如模板匹配(Template Match)和均值漂移(Mean Shift)

自顶向下(Top-down)的处理方法：模型驱动(Model-driven)的方法，依赖于所构建的模型或先验知识，比如卡尔曼滤波器(Kalman Filter)和粒子滤波器(Particle Filter) 。

模板匹配法**(Template Matching)**



- 在前一帧图像中目标位置（或模板T位置）为：(x, y)
- 在当前帧搜寻位置

$(x', y') = (x + dx, y + dy)$

使得

$\arg\max_{dx,dy} \mathbf{cov}(T(x, y), I(d + dx, y + dy))$

- 概念上相对比较简单
- 进行穷尽的搜索计算量非常大

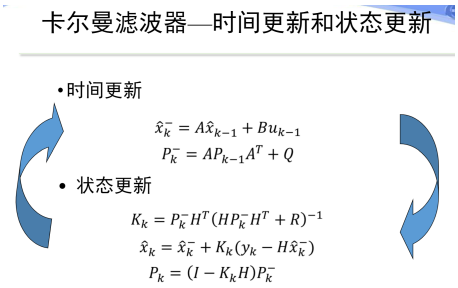
进一步参考：SCHWEITZER, H., BELL, J. W., AND WU, F. 2002. Very fast template matching. In European Conference on Computer Vision (ECCV). 358–372.

卡尔曼滤波器**(Kalman Filter)**

思想：利用线性系统状态方程，基于观测数据对系统状态进行最优估计。

基于卡尔曼滤波器的跟踪的原理：通过建立状态空间模型，把跟踪问题表示为动态系统的状态估计问题。

卡尔曼滤波器算法：



建模：存在系统噪声（影响状态）和测量噪声（影响观测值）。假设噪声服从高斯分布，即$Q = E(ww^T), R = E(vv^T)$，且协方差矩阵Q, R已知。

- 动态系统由状态转移方程和观测方程组成。
- 状态转移方程：

  $x_k = f(x_{k-1}, w_{k-1})$

  f：在很多跟踪问题中是非线性的

  x_k, x_{k-1}：当前时刻与前一时刻的状态

  w_{k-1}：系统噪声

观测转移方程：

$y_k = h(x_k, v_k)$

h：在很多跟踪问题中是非线性的

y_k：测量值

x_k：当前时刻的状态

v_k：测量噪声

基本假设：

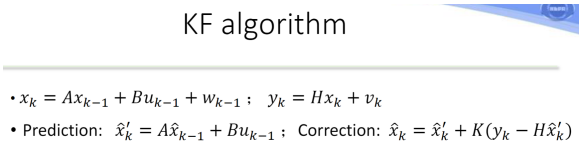
- 后验概率分布$p(x_{k-1}|y_{1:k-1})$为高斯分布
- 动态系统是线性的

  $x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$

  $y_k = Hx_k + v_k$
- 系统噪声和测量噪声时高斯分布的，协方差矩阵分别为Q和R。

推导：

1. 写出算法，求解最优K（Kalmangain）



2. 最小化未知的真实值和修改后的预测值差距P_k，并代入算法的表达式。

- minimize the difference $x_k - \hat{x}_k$

$$e = x - \hat{x};\ P = E(ee^T) = \begin{pmatrix} E(e_1e_1) & E(e_1e_2) & \cdots \\ E(e_2e_1) & E(e_2e_2) & \\ \vdots & & \ddots \end{pmatrix}$$

推导过程：

$$P_k = E\left[e_k e_k^T\right] = E\left[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T\right]$$

$$P_k = E\left[[(I - K_k H)(x_k - \hat{x}'_k) - K_k v_k]\right.$$
$$\left.[(I - K_k H)(x_k - \hat{x}'_k) - K_k v_k]^T\right]$$

3. 将对噪声的假设代入化简

系统状态变量和测量噪声之间是相互独立的

$$P_k = (I - K_k H) E\left[(x_k - \hat{x}'_k)(x_k - \hat{x}'_k)^T\right](I - K_k H) + K_k E\left[v_k v_k^T\right] K_k^T$$

预测值和真实值之间误差的协方差矩阵 $P'_k = E\left[e'_k e'^T_k\right] = E\left[(x_k - \hat{x}'_k)(x_k - \hat{x}'_k)^T\right]$

$$P_k = (I - K_k H) P'_k (I - K_k H)^T + K_k R K_k^T$$
$$P_k = P'_k - K_k H P'_k - P'_k H^T K_k^T + K_k (H P'_k H^T + R) K_k^T$$

4. 解得K，并为了简化表达，写出其他中间量的表达式。

$$\text{最小化 } T[P_k] = T[P'_k] - 2T[K_k H P'_k] + T[K_k (H P'_k H^T + R) K_k^T]$$

$$\frac{dT[P_k]}{dK_k} = -2(H P'_k)^T + 2K_k (H P'_k H^T + R)$$

$$\text{求得增益：} K_k = P'_k H^T (H P'_k H^T + R)^{-1}$$

$$\text{更新} P_k \quad P_k = P'_k - P'_k H^T (H P'_k H^T + R)^{-1} H P'_k = P'_k - K_k H P'_k = (I - K_k H) P'_k$$

$$\text{更新} P'_k \quad P'_{k+1} = E[e'_{k+1} e'^T_{k+1}] = E[(x_{k+1} - \hat{x}'_{k+1})(x_{k+1} - \hat{x}'_{k+1})^T] = E[(A(x_k - \hat{x}_k) + \omega_k)(A(x_k - \hat{x}_k) + \omega_{kk})^T]$$

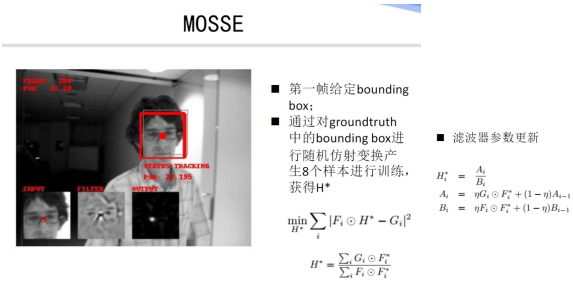
系统状态变量和系统噪声之间是相互独立的

$$= E[(Ae_k)(Ae_k)^T] + E[\omega_k \omega_k^T] = AP_k A^T + Q$$

卡尔曼滤波器的扩展：Extended Kalman Filter (EKF)，Unscented Kalman Filter (UKF)。同样基于高斯分布的假设；状态转移方程和测量方程为非线性函数；沿用KalmanFilter的框架；将非线性函数局部线性化。

基于相关滤波的跟踪方法（**MOSSE,2010**）

基本原理：在视频帧中利用相关核（h）找到响应值最高的位置，即实现跟踪。



不足：1.特征不够稳定（输入为原始灰度像素）。2.较难处理目标尺度变化情况。3.鲁棒性相对较弱。

基于**DNN**的跟踪方法

策略1：DNN特征+ 相关滤波 策略2：直接使用DNN进行目标跟踪

Visual Tracking with Fully Convolutional Networks(2016) 原理：Observation : Different layers encode different types of features. Higher layers capture semantic concepts on object categories,whereas lower layers encode more discriminative features to capture intra class variations.

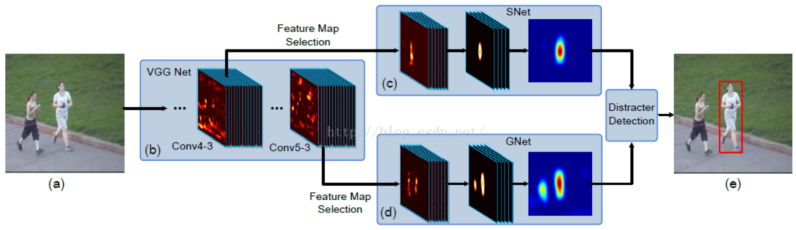


Figure 5. Pipeline of our algorithm. (a) Input ROI region. (b) VGG network. (c) SNet. (d) GNet. (e) Tracking results.

视觉定位

相机跟踪（摄像机定位）：通过图像序列，持续地计算出相机的位置、姿态，如SLAM（Simultaneous Localization And Mapping，同步定位与地图创建）。

任务特点：

- 适用于室内、外定位
- 基于图像的定位可以集成到手机等移动终端，方便廉价
- 基于位置的服务几乎无处不在，如智能机器人、虚拟现实、增强现实等等

应用举例：视觉定位应用之机场导航：在机场拍个照，然后返回你当前的位置和朝向。

分类：

- 离线的视觉定位的难点：图像质量参差不齐；计算精度与复杂度
- 在线的视觉定位的难点：实时性与计算精度；定位失败后的视觉重定位；

在线的视觉定位的方法分类：

- 环境已知：单纯定位；
  1. 基于图像检索的定位方法：定位技术速度快、定位精度相对较低
  2. 基于2D-3D匹配点的定位方法：定位精度相对较高、但依赖于2D-3D匹配精度影响
  3. 基于学习（神经网络）的定位方法：计算稳定性相对较好、定位精度相对较低、泛化能力相对较差
- 环境未知：同步定位与地图构建（Simultaneous Localization and Mapping,SLAM），指移动物体在自身位置不确定的情况下，利用自身的传感器，在未知的环境中创建一个与环境相一致的地图，并同时确定自身在地图中的位置。难点：数据关联；重定位；闭环检测。
  1. 基于滤波器的实时定位方法 核心思想：将相机位置、姿态和地图特征等未知信息作为滤波器的状态量，利用相机的观测特征不断地估计相机相机位置、姿态和地图特征。常用滤波器：卡尔曼滤波器、粒子滤波器等。
  2. 基于几何的实时定位方法（基于关键帧的实时定位方法） 双（多）线程处理：地图线程：利用几何重建方法构建环境地图；定位（跟踪）线程：利用图像信息和地图信息，实时计算相机的位置姿态

高层**2D**视觉：行为识别

运动分析包括：运动表达和行为识别和理解。

应用：自动从监控视频理解发生了什么事件。

人的行为分析难点

1. 人的行为的多样性：个体行为；人与人之间的交互行为；人和物体之间的交互行为
2. 遮挡（人-人；人-物体）情况复杂；
3. 人由于穿着的宽松衣物，阴影以及光照变化等因素的影响
4. Articulated motion:人体各个部位的运动是刚体运动；而人整体的运动是非刚体运动

运动表达

运动表达(Motion representation)目的：刻画运动前景的运动模式

运动表达——运动轨迹

运动轨迹：通过物体跟踪，可以得到物体特征点的轨迹。关键：特征点的选取；轨迹的描述

特征点的选取：比如最小包围框；手的质心；头和脚的特征点；运动前景

怎样通过特征点集合描述运动轨迹？

1. 直接按照时间顺序连接相邻帧之间的特征点。
2. 将特征点集合拟合成不同的多项式曲线。
3. 主曲线：一条空间曲线，从数据的中部光滑地通过，且不限制于对数据的光滑线性平均，甚至不限制于数据的中部是直线，只使得数据点集合到该曲线的正交距离最小。

运动轨迹的应用场合：交通监控，表述车辆、行人行动路线；动作、手势识别，表述运动物体或肢体局部的简单运动；人机接口。

运动轨迹的不足：

- 只能粗略地表述物体全局的运动信息；
- 无法描述运动细节；
- 没有有效地体现时间信息。

运动表达——时空图表达

原理：将图像序列的前景运动信息和时间信息用一张图表述出来。

- 运动能量图（Motion Energy Image——MEI）：帧间差分得到前景的二值图像，然后将$[1,t]$的所有前景二值图像求并集得到t时的能量图。问题：区分不了可逆的运动（坐下和站起）。
- 运动历史图（Motion History Image——MHI）：改进MEI，更近的运动像素更亮。
- 其它“运动图”。

主要应用场合：行为、动作、手势识别；人机接口。 优点：较好地包含了全局运动、形状、时间信息。 不足之处：缺少局部运动信息，不动有效地区分局部变化的动作；不动有效地区分速度的变化。

行为识别

行为识别可以看作是时变特征数据的分类问题，即将待识别的行为序列（测试序列）与预先标记好的代表典型行为的参考序列进行匹配。

基于模板匹配的方法

思想：用输入图像序列提取的特征与在训练阶段预先保存好的模板进行相似性度量，选择与测试序列距离最小的已知模板的所属类别作为被测试序列的识别结果。 步骤：

1. Temporal Templates (Bobickand Davis PAMI 2001) 将图像序列目标的运动信息转化为运动能量图像(MEI)和运动历史图像(MHI)；在图像上提取基于不变矩的运动特征(具有平移、旋转和尺度不变性)，并采用马氏距离度量测试序列和模板之间的相似性。

2. 动态时间归整(DTW)：既使测试序列模式与参考序列模式的时间尺度不完全一致，只要时间次序约束存在，DTW就能较好地完成测试序列和参考序列之间的模式匹配。基于动态规划（dynamic programming）思想

假设两个向量C和Q，长度分别为m和n。那么DTW的目标就是要找到一组路径：

$$W = w_1, w_2, \dots, w_K \quad \max(m,n) \le K \le m+n-1$$

$$w_k = (c_i, q_j)_k$$

使得经由上述路径的“点对点”对应距离之和为最小：

$$DTW(C,Q) = \min\{\frac{1}{K}\sum_k \|c_i - q_j\|\}$$

此路径必须满足以下条件：

1 首尾对齐： $w_1 = (c_1, q_1) \quad w_K = (c_m, q_n)$

2 单调性： $w_k = (a,b), w_{k-1} = (a',b')$

$$0 \le a - a' \le 1, \quad 0 \le b - b' \le 1$$

基于状态转移图模型的方法

思想：定义每个静态姿势作为一个状态，这些状态之间通过某种概率联系起来。任何运动序列可以看作为这些静态姿势的不同状态之间的一次遍历过程，在这些遍历期间计算联合概率，其最大值被选择作为分类行为的标准。 HMM是一个双重随机过程，两个组成部分：马尔可夫链：描述状态的转移，用转移概率描述。特定状态下可观察值，用观察概率描述。 估值问题：给定模型参数（A, B, π）的情况下，求某种观测序列 O 出现的概率。

Forward Algorithm:

$$P(O|\lambda)=\sum_{S} P(O|S,\lambda)P(S|\lambda)$$

•前向变量

$$\alpha_t(i)=P(O_1,O_2,\cdots,O_t,q_t=\theta_i|\lambda)$$

•则有：

$$\alpha_1(i)=\pi_i b_i(O_1)$$

$$\alpha_{t+1}(j)=[\sum_{i=1}^{N}\alpha_t(i)a_{ij}]b_j(O_{t+1})$$

算法步骤

I. 初始化： $\alpha_1(i)=\pi_i b_i(o_1)$

II. 迭代： $\alpha_{t+1}(j)=\left[\sum_{i=1}^{N}\alpha_t(i)a_{ij}\right]b_j(o_{t+1})$

III. 终止 $P(O|\lambda)=\sum_{i=1}^{N}\alpha_T(i)$

解码问题：给定模型参数（A,B,π）、观测序列 O，让我们求出造成这个观测序列最有可能对应的状态序列 X 。 Viterbi Algorithm:

- 初始化：
$$\delta_1(i)=\pi_i b_i(O_1),\quad 1\le i\le N$$
$$\varphi_1(i)=0,\quad 1\le i\le N$$
- 递归：
$$\delta_t(j)=\max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}]b_j(O_t),\quad 2\le t\le T,1\le j\le N$$
$$\varphi_t(j)=\arg\max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}],\quad 2\le t\le T,1\le j\le N$$
- 终结：
$$P^*=\max_{1\le i\le N}[\delta_T(i)])$$
$$q_T^*=\arg\max_{1\le i\le N}[\delta_T(i)]$$
- 求S序列：
$$q_t^*=\varphi_{t+1}(q_{t+1}^*),\qquad t=T-1,T-2,\ldots,1$$

学习问题：给定观测序列 O，用极大似然估计求模型参数（A,B,π）。 Baum-Welch Algorithm:

•基本思路与步骤：

1. 初始模型（待训练模型） λ_0,
2. 基于 λ_0以及观察值序列O，训练新模型 λ；
3. 如果 log(P(O|λ)) - log(P(O|λ_0) < Delta，
说明训练已经达到预期效果， 算法结束。
4. 否则，令 $\lambda_0=\lambda$ ，继续第2步工作

物体表达和场景表达

物体表达

计算机视觉中的物体表达（object representation）：计算机对目标物体的存储形式。 著名的物体表达理论：马尔的三维物体表达；基于二维图像的物体表达；逆生成模型表达。

在计算机视觉领域中，深度神经网络（DNN）被广泛地认为能够学习到高效的物体表达，且在大部分研究主题中取得了压倒性的优势（除三维计算机视觉相关主题之外）。

传统识别

物体识别：从物体图像或物体点云中对物体类别进行自动识别

- 图像物体识别：AlexNet(2012),VGG(2014),GoogleNet(2014),ResNet(2015),
- 人脸识别：DeepID,VS2VI
- 三维物体识别: PointNet(2017)

小样本识别

小样本（FewShot）识别：给定少量的有标注数据，学习一个可以识别这些类别数据的模型。

分类：

- 基于度量学习的方法
- 基于元学习的方法
- 基于数据增强的方法

零样本识别

零样本（Zero Shot）识别：给定有标注的已见类别数据和语义特征，学习一个可以识别未见类别数据的模型（但测试时知道其语义特征！比如颜色、名字、条纹）。

方法：使用有标注的已见类别数据和语义特征建立一个视觉和语义特征之间的映射模型，根据语义特征推断未见类别。

分类I：

- 归纳式（Inductive）：训练过程只使用训练数据；面临域漂移问题。
- 直推式（Transductive）：训练过程既使用训练数据也使用没有标记的测试数据；由于使用了未标记的不可见类样本，能够在一定程度上减轻域漂移问题的影响。

分类II：

- 基于表征学习的零样本学习方法：学习一个具有判别性的特征空间，并在此空间建立视觉-语义映射模型。DeViSE(2013)
- 基于生成模型的零样本学习方法：利用生成模型生成未见类别数据，然后用生成的未见类别数据训练分类器。GAN(2014)，Conditional GAN(2014, 带条件约束的生成模型，在生成器和判别器中均引入了条件变量y, y可以是任何的辅助信息，如类别标签、其它模态的数据等）)，f-CLSWGAN(2018, 基于给定的语义特征来合成视觉特征),

零样本识别存在的问题： 域漂移问题：利用有标注的已见类别数据学习得到的视觉-语义映射模型不能很好地适用于未见类别域。 域偏置问题：利用有标注的已见类别数据学习得到的视觉-语义映射模型倾向于将模型输入分类为一个已见类别。

关键：学习出更具泛化能力与判别能力的视觉图像表达

开集识别

以上的识别都是闭集识别，即模型知道所有的类别。零样本识别中即使训练集和测试集都没有斑马类别的样本，但测试集中包含了斑马类别的描述，这是有用的语义信息。 然而开集识别要实现的是训练集和测试集的类别可以有一样，也可以有不一样的，但模型最终能在不提供任何辅助信息的情况下区分出已知类别，同时也能分辨出未知类别并作相应处理。

开集识别：既要正确地对已知类别样本进行分类，又要识别出未知类别的样本。 难点：特征混淆问题（已知类别样本之间的特征混淆、已知类别样本与未知类别样本的特征混淆） 两种基本思路： 1、判别式方法：直接训练一个具有良好泛化性能的开集识别神经网络，将原有的k类分类问题转化为(k+1)类分类问题。 2、生成式方法：对已知类别样本的分布进行建模，根据测试样本是否符合建模的分布、符合哪一类的分布判断属于未知类别或属于某种已知类别。

神经场景表达

通过生成式查询和绘制网络来学习场景表达的框架性算法。 基本思想：在无监督学习框架下，通过学习输入图像的表达和控制在新视点下图像的生成质量，来学习场景物体形状、姿态、颜色、纹理和光照的表达分布。

NeRF(Neural Radiance Fields, 2020)

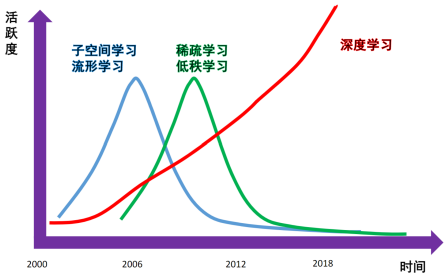
目标：对目标场景进行隐式建模，使得该模型能自动合成任意视角的场景图像。 模型训练过程中的输入：针对目标场景，拍摄得到的不同视角图像（相机参数已知）。

模型测试过程中的输出：任意视角下的目标场景图像。 神经辐射场（Neural Radiance Fields）完成场景表达；基于辐射场的体素渲染（volume rendering）完成渲染。

NeRF优点：能够较好地表达复杂场景。令人振奋的一篇论文。NeRF不足：场景可表示的大小受限制、不能跨场景泛化、场景必须静止且光照不变、需要位姿信息等。

发展现状：大量的基于NeRF框架的方法，Mip-NeRF、Point-NeRF、Human-NeRF、Urban-NeRF、Block-NeRF、Raw-NeRF…

计算机视觉中机器学习方法

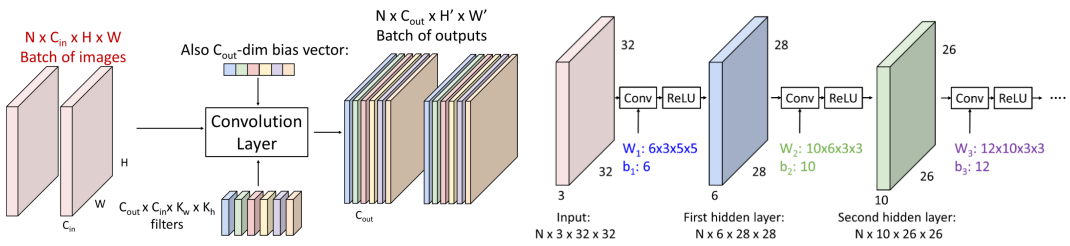


深度学习

1. 神经网络：Kolmogorov理论指出：双隐层感知器就足以解决任何复杂的分类问题；无隐藏层的神经网络就是线性的，将超平面分成两个；单隐藏层的神经网络可以形成凸域（解决异或问题）。

2. CNN：卷积核（filter）是对连续卷积核函数的离散化近似（比如由高斯核函数在每个位置算出值就得到高斯卷积核）；感受野(Receptive Field)，即某个神经元能看到的输入图像的区域（由局部连接的特性而来）；CNN设计思想：局部连接、权值共享、多核卷积。

3. CNN参数的计算：$3*32*32$的图片（此处的3表示图像RGB的3个维度），通过$3*5*5$的filter（此处的3表示卷积核有3个channel，这个要和输入的channel数保持一致），得到$1*28*28$（no padding, stride=1）；更一般的，channel就是矩阵的深度，filter通过定义input channel数和output channel数来定义这种深度的变化；用公式来说，图像为$w*h*c_{in}$，想得到$w'*h'*c_{out}$，那么filter就应该是$f*f*c_{in}*c_{out}$，这样通过定义filter的input channel数和output channel数来满足矩阵深度的要求，然后通过定义f的大小、padding、stride的大小满足矩阵长宽的要求。output channel数实际上就是filter的个数，前面例子$3*32*32$的图片通过1个$3*5*5$的filter得到$1*28*28$，那么这个图片通过6个参数不同的$3*5*5$的filter，再将结果stack便得到$6*28*28$。下面示意图还给卷积结果加了个偏置。



Convolution Summary

Input: C_{in} x H x W
Hyperparameters:
- **Kernel size**: K_H x K_W
- **Number filters**: C_{out}
- **Padding**: P
- **Stride**: S

Weight matrix: C_{out} x C_{in} x K_H x K_W
giving C_{out} filters of size C_{in} x K_H x K_W
Bias vector: C_{out}
Output size: C_{out} x H' x W' where:
- H' = (H – K + 2P) / S + 1
- W' = (W – K + 2P) / S + 1

Common settings:
$K_H = K_W$ (Small square filters)
P = (K – 1) / 2 ("Same" padding)
C_{in}, C_{out} = 32, 64, 128, 256 (powers of 2)
K = 3, P = 1, S = 1 (3x3 conv)
K = 5, P = 2, S = 1 (5x5 conv)
K = 1, P = 0, S = 1 (1x1 conv)
K = 3, P = 1, S = 2 (Downsample by 2)

Pooling Summary

Input: C x H x W
Hyperparameters:
- Kernel size: K
- Stride: S
- Pooling function (max, avg)

Output: C x H' x W' where
- H' = (H – K) / S + 1
- W' = (W – K) / S + 1

Learnable parameters: None!

Common settings:
max, K = 2, S = 2
max, K = 3, S = 2 (AlexNet)

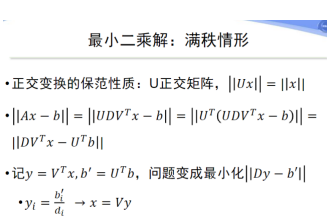
稀疏捆绑调整（**Sparse Bundle Adjustment**）

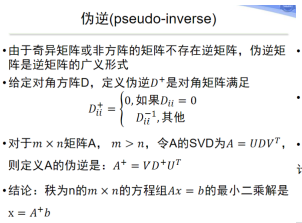
考虑Ax=b的方程组，令A为m×n矩阵，则存在三种可能性：

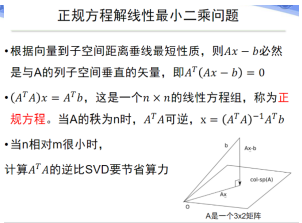
- m<n 解不唯一，有一个解矢量空间
- m=n 只要A可逆，有唯一解
- m>n 方程组一般没有解，除非b属于由A的列所生成的子空间。

最小二乘解：满秩情形 考虑m>n并且A的秩为n的情形：如果解不存在，那么寻找一个向量x，使得|Ax−b|最小，x称为该超定方程的一个最小二乘解；

求解方法：1. SVD。2. 伪逆 3. 正规方程



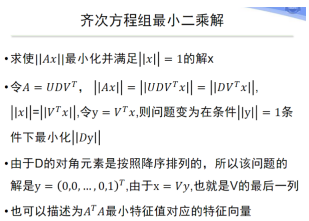




齐次方程组最小二乘解：Ax=0

- R(A)<n, 有非零解，解不唯一
- R(A)=n，平凡解x=0，目标寻找方程组的非零解

如果x是方程组的解，那么kx也是其解。求使|Ax|最小化并满足x=1的解x



迭代方法非线性最小问题 求解X=f(P)，X是测量向量(y的意思，已知)，P是参数向量(x的意思)。转换为给定X，寻找向量$\hat{P}$,满足$\epsilon = f(\hat{P}) - X$并使得$\epsilon$|最小。 Δ就是算法得到的下一个点和当前点的向量$P_{t+1} = P_t + \Delta$。Newton方法（$H\Delta = -J^T\epsilon$）；高斯牛顿法（$J^TJ\Delta = -J^T\epsilon$）；梯度下降法（$\lambda\Delta = -J\epsilon$）；Levenberg-Marquardt iteration（$(J^TJ + \lambda I)\Delta = -J^T\epsilon$）；Sparse Levenberg-Marquardt。

Levenberg-Marquardt iteration

- $(J^TJ+\lambda I)\Delta=-J^T\epsilon$ 增广正规方程组
- λ的初始值通常取$N=J^TJ$对角线元素的平均值除以10^3
- 如果求解的Δ使得ϵ减小，则该增量被接受，并在下一次迭代之前将λ除以10
- 反之，如果Δ使得ϵ增加，则将λ乘以10并重解Δ，直到求解到使得ϵ减小为止
- 另一种变形N'：$N=J^TJ$, $N'_{ii}=(1+\lambda)N_{ii}$, $N'_{ij}=N_{ij}$
- N'的奇异问题

Sparse Levenberg-Marquardt

- LM算法只适合参数较少的最小化问题，因为LM解正规方程组的复杂度是N^3，并且还有多次迭代
- 考虑m幅图像的三维重构问题，需要估计所有相机的参数和三维点。对应射影重构问题，则参数个数为11m+3n
- 可以利用正规方程组的稀疏分块结构节约计算时间

子空间分析

原理：降维后再对低维数据进行处理

主成分分析方法（**Principal Component Analysis**，**PCA,1901**） 基本思想：寻找投影映射P，使得样本从N维降到m维（$m<N$），同时最小化样本和从低维重构的样本的平方误差。几何上来说：PCA是一个正交化线性变换，把数据变换到一个新的坐标系统中，使得这一数据的任何投影的第一大方差在第一个坐标（称为第一主成分）上，第二大方差在第二个坐标（第二主成分）上，依次类推。

算法：

计算步骤

- 对于给定的样本矩阵$X=[x_1,x_2,\dots,x_n]$, 其中x_i是N维向量

1. 计算样本均值 $\mu=\frac{1}{n}\sum_{i=1}^{n}x_i$
2. 计算离散度矩阵 $S=\sum_{i=1}^{n}(x_i-\mu)(x_i-\mu)^T$
3. 将离散度矩阵进行特征值分解，取最大的m个特征值和相应的特征向量$P=[v_1,v_2,\dots,v_m]$，其中P是$N\times m$维矩阵。
4. 实现降维 $y_i=P^T(x_i-\mu)$

推导：

- 变量定义：
- 训练样本集：$X=[x_1,x_2,\dots,x_n]$，其中x_i是N维向量
- 样本均值：$\mu=\frac{1}{n}\sum_{i=1}^{n}x_i$
- 离散度矩阵：$S=\sum_{i=1}^{n}(x_i-\mu)(x_i-\mu)^T$
- 投影后的低维数据：$y_i=P^T(x_i-\mu)$，其中P是$N\times m$维投影矩阵

怎样使新变量尽可能地保持原有的信息呢？

重构误差最小（least squares reconstruction）

样本：$X=[x_1,x_2,\dots,x_n]$

投影：$y_i=P^T(x_i-\mu)$，其中P是$N\times m$维矩阵

样本重构：$\hat{x}_i=Py_i$

重构误差：$e=\sum_{i=1}^{n}\|(x_i-\mu)-\hat{x}_i\|^2$

- 最小化重构误差，等价于如下优化问题：

$$P=\arg\max_P|P^TSP|$$

- 特征值分解：$Sv_i=\lambda_i v_i, i=1,2,\dots,n$

其中$\lambda_1\geq\lambda_2\geq\cdots\geq\lambda_n$，$\lambda_i$与$v_i$分别是特征值和对应的特征向量

- 最大的一组特征值对应的特征向量组成的子空间即为所求：

$$P=[v_1,v_2,\dots,v_m]$$

Eigenfaces,1991: 思考：PCA是无监督的降维方法，如何用到识别这种有类别的任务中？在子空间进行欧氏距离度量！

人脸表述与识别

- 人脸表述：基于得到的投影矩阵P，每张人脸可以由一个m维向量表示

$$y_i=P^T(x_i-\mu)$$

- 人脸识别：
  - 对于一张输入的人脸图像，计算它的低维表述

$$y_{input}=P^T(x_{input}-\mu)$$

  - 计算 $k=\arg\min_i\|y_{input}-y_i\|$

评价：

PCA对于椭球状分布的样本集有很好的效果, 学习所得的主方向就是椭球的主轴。

PCA 是一种非监督的算法, 能找到很好代表所有样本的方向, 但这个方向对于分类未必是最有利的。

独立成分分析方法（**Independent Component Analysis**，**ICA**）：InfoMax 算法（信息极大化算法,1995），FastICA 算法（固定点算法,1997）

- ICA，独立成分分析：指从多个源信号的线性混合信号中分离出源信号的技术。
- ICA假设：源信号统计独立。

- 模型： $x = A * s$ 或 $x = \sum_{i=1}^{n} a_i s_i$

- 求解： $u = W * x = W * A * s$

线性判别分析方法（**Linear DiscriminatAnalysis**，**LDA**）

LDA（Linear Discriminant Analysis），又称Fisher Discriminant Analysis ，是一种监督的维数约简方法

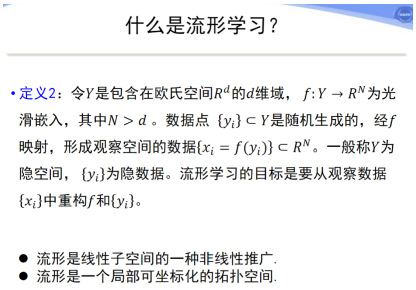
Fisher判别原则：寻找投影W，使得投影后的样本类内散度最小，而类间散度最大。

线性方法的不足：现实中数据的有用特性往往不是特征的线性组合。

流形学习

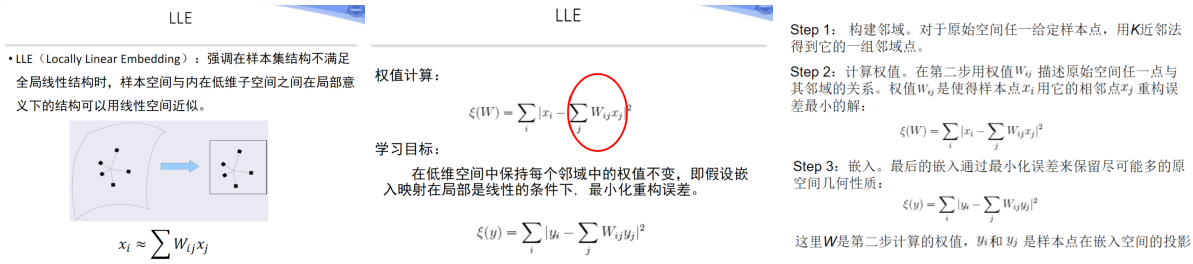
流形：如果一个N维的拓扑空间M内的任意一点都存在一个邻域$U \in M$使得该邻域是N维欧氏空间的同胚，则这个拓扑空间M被称为流形。

流形学习认为高维数据都是有结构的，可以由低维数据进行变化得来。我们就是要从高维数据求低维数据和这个变化。



LLE(Locally Linear Embedding,2000):

想法：在原空间确定小范围的权值，在低维空间保持这些权值（W）不变来求样本点的低维映射（y）。



优点：

- LLE算法可以学习任意维数的低维流形.
- LLE算法中的待定参数很少, K（邻居个数）和d（低维的维度）.
- LLE算法中每个点的近邻权值在平移, 旋转,伸缩变换下是保持不变的.
- LLE算法有解析的整体最优解,不需迭代.
- LLE算法归结为稀疏矩阵特征值计算, 计算复杂度相对较小, 容易执行.

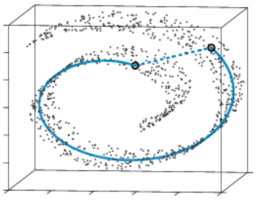
缺点：

- LLE算法要求所学习的流形只能是不闭合的且在局部是线性的.
- LLE算法要求样本在流形上是稠密采样的.
- LLE算法中的参数K, d 有过多的选择.
- LLE算法对样本中的噪音很敏感

MDS:

MDS的基本思想: 约简后低维空间中任意两点间的距离应该与它们在原始空间中的距离相同. MDS的求解: 通过适当定义准则函数来体现在低维空间中对高维距离的重建误差, 对准则函数用梯度下降法求解,对于某些特殊的距离可以推导出解析解法.

MDS的失效



Isomap（等距映射，**2000**）= MDS + 测地距离： 主要思想：建立在多维尺度变换(MDS)的基础上，力求保持数据点的内在几何性质，即保持两点间的测地距离，不是欧氏距离。

Step 1：在样本集上构建近邻图*G*。如果样本*i*和*j*之间距离小于某个阈值,或者他们为k-近邻，则连接*i*和*j*

Step 2：计算样本两两之间测地距离（用Dijkstra算法），建立测地距离矩阵 $D_G = d_G(x_i, x_j)$

Step 3：利用MDS算法构造内在*d*维子空间，最小化下式

$$E = \| \tau(D_G) - \tau(D_Y) \|_{L^2}$$

矩阵变换算子$\tau(D) = -HSH/2$将距离转换成MDS所需内积形式，其中*S*是平方距离矩阵 $\{S_{x_i x_j} = D^2_{x_i x_j}\}$是集中矩阵 $\{H_{x_i x_j} = \delta_{x_i x_j} - 1/N\}$

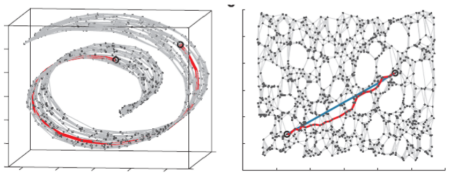
上式的最小值可以通过求矩阵$\tau(D_G)$的*d*个最大特征值对应的特征向量来实现

缺点：

- Isomap是非线性的, 适用于学习内部平坦的低维流形, 不适于学习有较大内在曲率的流形.
- Isomap算法中有两个待定参数K, d.

效果

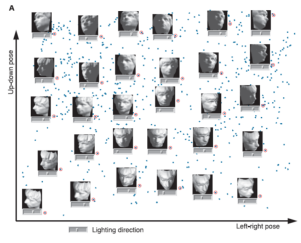
Swiss Roll在2D流形空间的投影



3维数据集　　2维投影

人脸图像在2D流形空间的投影

横坐标反映了光照变化，纵坐标反映姿态变化



Laplacian Eigenmap:

主要思想：在高维空间中离得很近的点投影到低维空间中的像也应该离得很近。

主要思想的数学表达：

令样本集：$X = (x_1, x_2, \cdots, x_n)$　投影后样本：$\mathbf{y} = (y_1, y_2, \cdots, y_n)$

LE的目标是最小化目标函数：

$$\sum_{i,j=1}^{n} \|y_i - y_j\|^2 W_{ij}$$

●这里权值反映样本之间的关系，一般用热核表示：

$$W_{ij} = e^{-\frac{\|x_i - x_j\|^2}{t}}$$

也可以简单定义成1（节点i和j相邻）或0（不相邻）

●为了使最小化问题解唯一，必须加上尺度归一的限制条件，目标函数变为:

$$\arg \min_{\mathbf{y}^T D\mathbf{y}=1} \mathbf{y}^T L\mathbf{y}$$

这里L=D-W被称为Laplacian矩阵

$D_{ii} = \sum_j W_{ji}$ 是对角矩阵

● 可以转化成广义特征值问题求解：

$$L\mathbf{y} = \lambda D\mathbf{y}$$

LaplacianEigenmap算法的特点

- 算法是局部的非线性方法。
- 算法与谱图理论有很紧密的联系。
- 算法中有两个参数k、d.
- 算法通过求解稀疏矩阵的特征值问题解析地求出整体最优解。
- 算法使原空间中离得很近的点在低维空间也离得很近, 可以用于聚类。
- 没有给出显式的投影映射，即，对于新样本（out-of-sample）无法直接得到其在低维子流形上的投影。

总结： LLE, Isomap, LaplacianEigenmap有效的原因

1. 它们都是非参数的方法，不需要对流形的很多的参数假设。
2. 它们是非线性的方法，都基于流形的内在几何结构，更能体现现实中数据的本质。
3. 它们的求解简单，都转化为求解特征值问题，而不需要用迭代算法。

流形学习研究的常规模式：

1 对嵌入映射或者低维流形作出某种特定的假设，或者以保持高维数据的某种性质不变为目标。

2 将问题转化为求解优化问题。

3 提供有效的算法。

流形学习中存在的问题：

1 如何确定低维目标空间的维数?

2 当采样数据很稀疏时，怎样进行有效的学习?

3 将统计学习理论引入流形学习对其泛化性能进行研究。

稀疏表达

定义：Sparse representations are the representations that account for most or all information of a signal with a linear combination of a small number of elementary signals.

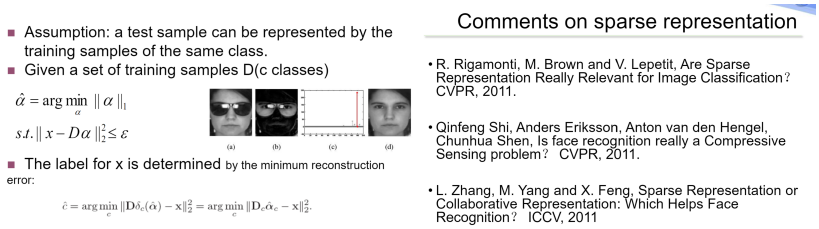
最开始的优化目标：在保证完美恢复信号的约束下最小化系数数量（0范数）。这是一个NP-hard问题，但是Donoho(Stanford) and Elad: there exists a unique solution under some condition.

- Candes (Stanford) and Tao(UCLA): Under the RIP(Restricted Isometry Property) condition, the solution to the original L0-norm problem is the same as the one to the corresponding L1-norm problem:

$$\min_{\alpha} \|\alpha\|_0 \quad s.t. \quad x = D\alpha \;\Rightarrow\; \min_{\alpha} \|\alpha\|_1 \quad s.t. \quad x = D\alpha$$

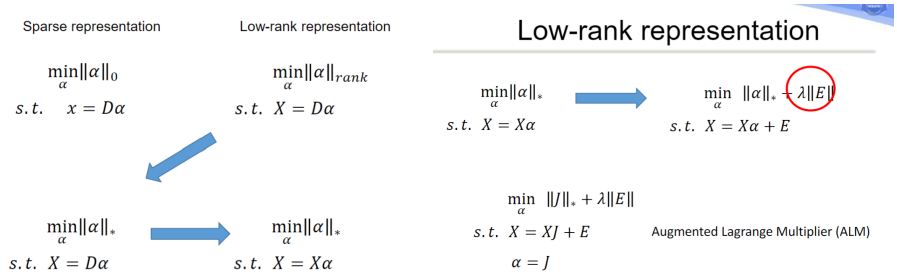
Convex!

Sparse Representation for Recognition(2009)：最后识别是通过在全部样本上的系数的同个类别的系数重建结果最好的。



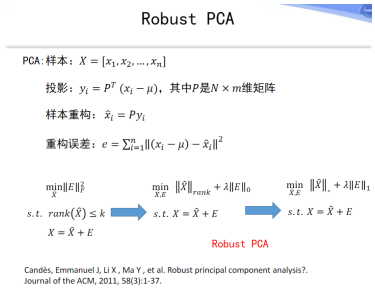
低秩表达

左侧图左上是系数表达，右上是低秩表达（注意α, X是矩阵了），秩不好求就用核函数，在很多应用情况下字典就是样本集（$D = X$）；右侧图是因为约束不一定能够满足，于是加一个误差矩阵E。这样还有个好处，我们可以进行图像去噪这种任务了



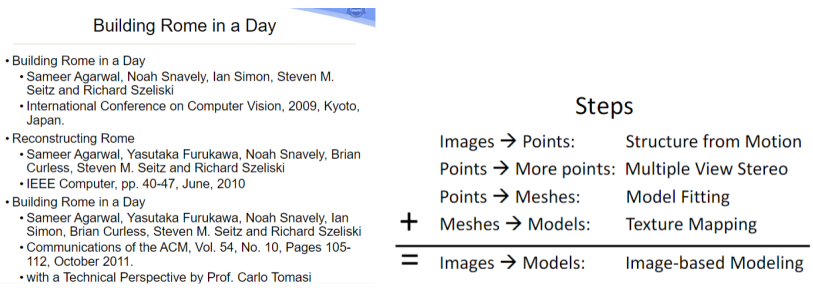
Robust PCA(2011)

研究动机：数据矩阵X既包含结构信息，也包含噪声，因此可以将矩阵X分解为两个矩阵的和$\hat{X} + E$，一个是低秩的，另一个是稀疏的



典型计算机视觉系统

基于图像的大规模场景建模系统**(SFM,MVS)**：构建物理世界地图



机器人视觉导航与定位系统**(SLAM)**：局部位姿估计

视觉定位与增强现实系统**(Localization, AR)**：绝对的位姿估计

乘积量化（Product Quantization）

- 乘积量化（PQ）算法是一种加快图像检索速度的检索算法
- 例子：假设有1M的图像表达向量，向量的维度为D=128，则对于一幅查询图像来说，需要计算1M个余弦距离。如何加快查询速度？KDTree？LSH(Locality Sensitive Hashing)？这里采用乘积量化。
- 方法：
    1. 空间切分：PQ先将D维空间切分成M份：即将128维空间切分成M个D/M维的子空间，如M=8，则分成8个16维子空间
    2. 量化：每个子空间内都会有1M个16维短向量，为每个子空间单独训练一本码本，码本大小k=256，依次量化每个子空间的数据（计算每个短向量距离最近的聚类中心）
    3. 压缩：每一个原始向量可以压缩成M个索引值构成的压缩向量，可以表示k的M次方个图像，空间大小也由原来的128x4 bytes -> 8 bytes
    4. 距离计算： 对称距离计算：直接使用两个压缩向量x，y的索引值所对应的码字q(x),q(y)之间的距离代替，而q(x),q(y)之间的距离可以离线计算，因此可以把q(x),q(y)之间的距离制作成查找表，只要按照压缩向量的索引值进行对应的查找就可以了，所以速度非常快 非对称距离计算：使用x，q(y)之间的距离代替x，y之间的距离，其中x是查询向量。虽然y的个数可能有上百万个，但是q(y)的个数只有k个，对于每个x，我们只需要在输入x之后先计算一遍x和k个q(y)的距离，制成查找表（因为只有k个，所以速度是非常快的），然后按照y对应的压缩向量索引值进行取值操作