# 计算机视觉实践报告（四）

## 1.实验目的

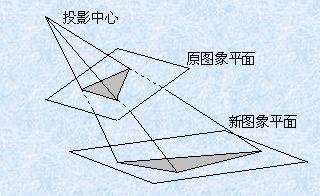
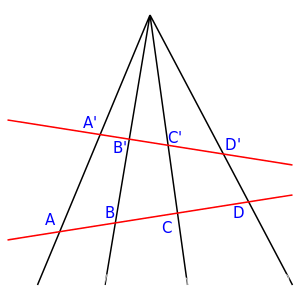
* 单应性变换，计算图片之间的单应性变换，并对其单应性变换进行分析。

### 2.实验原理

#### 2.1单应性变换

两个不同视角的图像上的点对的homogeneous coordinate可以用一个射影变换（projective transformation）表述，即：x1 = H\*x2

二维和三维的图示如下：

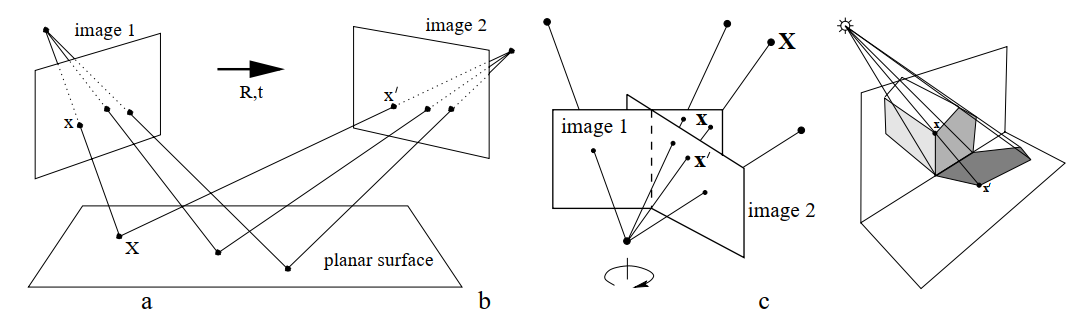


射影变换也叫单应Homography，“Homo”前缀就是same的意思，表示“同”，homography就是用同一个源产生的graphy，中文译过来大概就是“单应”。

#### 2.2单应性矩阵

单应矩阵描述两个平面上的对应点之间的变换关系，同一个平面在任意坐标系之间都可以建立单应性变换关系，因此矩阵H就叫单应性矩阵。x1和x2都是3\*1的齐次坐标，因此H是一个3\*3的矩阵：

如（a）：plannar surface上的X点可以通过单应性矩阵H1和H2变换到image1和image2，（b）和（c）同理。

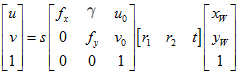


##### 2.3单应性矩阵的自由度

单应矩阵是3\*3的矩阵，为什么它的自由度是8呢？公式x1 = H\*x2，如果给定一个单应矩阵H={h\_ij}，给它每个元素乘上同一个数a，得到的单应a\*H和H作用相同。这是因为新单应a\*H无非是把齐次点x1变成了齐次点a\*x1，因此可以把a换成1/h22，那么H就变成了只有8个自由元素的矩阵。

##### 2.4单应性矩阵的求解

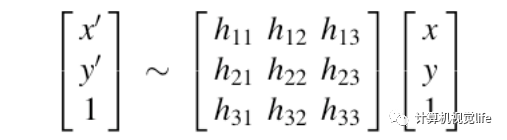
我们已经得到了像素坐标系和世界坐标系下的坐标映射关系：



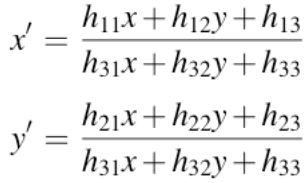
其中，u、v表示像素坐标系中的坐标，s表示尺度因子，fx、fy、u0、v0、γ（由于制造误差产生的两个坐标轴偏斜参数，通常很小）表示5个相机内参，R，t表示相机外参，Xw、Yw、Zw（假设标定棋盘位于世界坐标系中Zw=0的平面）表示世界坐标系中的坐标。

首先，我们假设两张图像中的对应点对齐次坐标为(x',y',1)和(x,y,1)，单应矩阵H定义为：

则有：

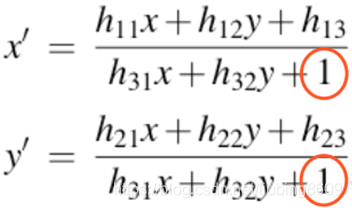


矩阵展开后有3个等式，将第3个等式代入前两个等式中可得：

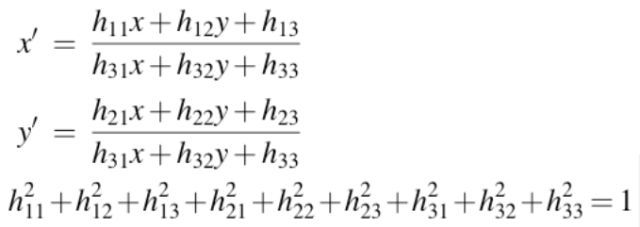


8自由度下H计算过程有两种方法。

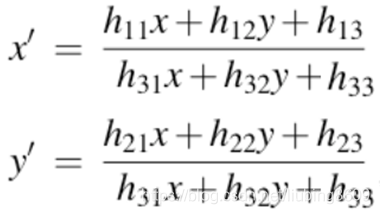
第一种方法：直接设置h33=1，那么上述等式变为：



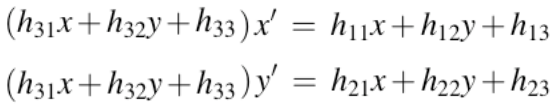
第二种方法：将H添加约束条件，将H矩阵模变为1，如下：



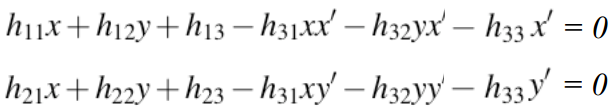
以第2种方法（用第1种也类似）为例继续推导，我们将如下等式（包含||H||=1约束）:



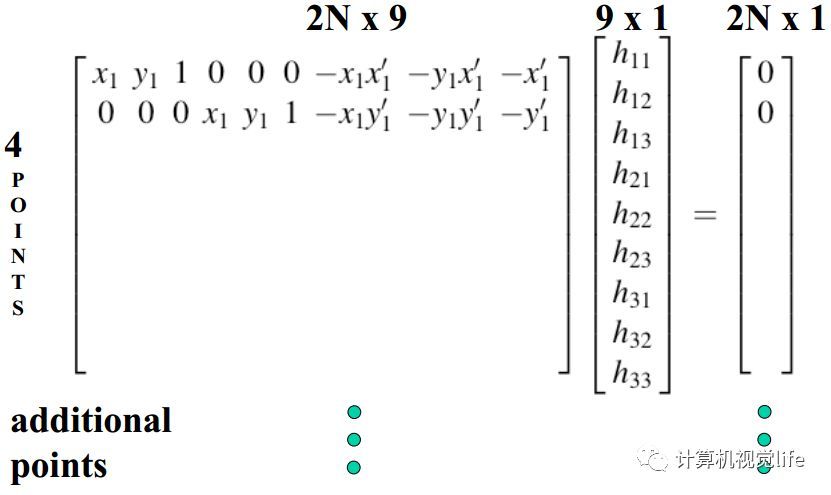
乘以分母展开，得到：



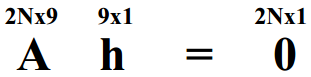
整理，得到：



假如我们得到了两幅图片中对应的N个点对（特征点匹配对），那么可以得到如下线性方程组：



写成矩阵形式：



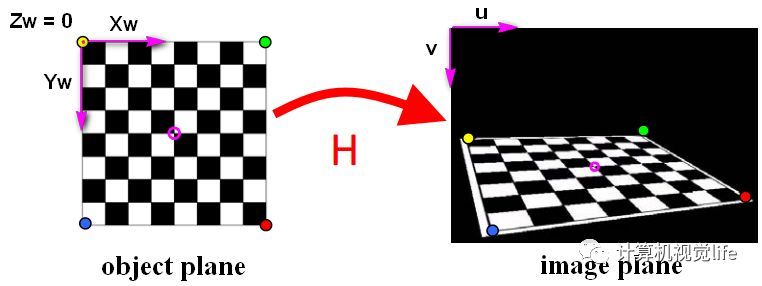
由于单应矩阵H包含了||H||=1约束，因此根据上图的线性方程组，8自由度的H我们至少需要4对对应的点才能计算出单应矩阵。这也回答了前面图像校正中提到的为何至少需要4个点对的根本原因。

但是，以上只是理论推导，在真实的应用场景中，我们计算的点对中都会包含噪声。比如点的位置偏差几个像素，甚至出现特征点对误匹配的现象，如果只使用4个点对来计算单应矩阵，那会出现很大的误差。因此，为了使得计算更精确，一般都会使用远大于4个点对来计算单应矩阵。另外上述方程组采用直接线性解法通常很难得到最优解，所以实际使用中一般会用其他优化方法，如奇异值分解、Levenberg-Marquarat（LM）算法（后续文章会介绍）等进行求解。

如何根据标定图得到单应矩阵？

经过前面一系列的介绍，我们应该大致明白如何根据打印的棋盘标定图和拍摄的照片来计算单应矩阵H。我们来总结一下大致过程。

1. 打印一张棋盘格标定图纸，将其贴在平面物体的表面。
2. 拍摄一组不同方向棋盘格的图片，可以通过移动相机来实现，也可以移动标定图片来实现。
3. 对于每张拍摄的棋盘图片，检测图片中所有棋盘格的特征点（角点，也就是下图中黑白棋盘交叉点，中间品红色的圆圈内就是一个角点）。我们定义打印的棋盘图纸位于世界坐标系Zw=0的平面上，世界坐标系的原点位于棋盘图纸的固定一角（比如下图中黄色点）。像素坐标系原点位于图片左上角。



1. 因为棋盘标定图纸中所有角点的空间坐标是已知的，这些角点对应在拍摄的标定图片中的角点的像素坐标也是已知的，如果我们得到这样的N>=4个匹配点对（越多计算结果越鲁棒），就可以根据LM等优化方法得到其单应矩阵H。

## 3单应变换的应用

#### 3.1 相机中的应用

在相机的内参数标定过程中会用到求解单应性矩阵对一个棋盘格图像，棋盘格的世界坐标系是用户任意设定的，标定的时候，默认世界坐标系就是以标定板左上角点为原点，z轴垂直于标定板，xoy面与标定板重合的三维直角坐标系。棋盘格的格子长度已知，因此可以知道各个角点的世界坐标系坐标XYZ（Z=0）。由于Z=0，因此可以忽略掉Z这个维度，世界坐标系中某个坐标(X,Y,Z,1)到图像坐标(x,y,1)的变换就等价于(X,Y,1)到图像坐标(x,y,1)的变换。此时的变换矩阵就由3\*4变为3\*3，成为了单应性矩阵。

因此用4个角点就可以计算H的8个参数。以不同的角度对棋盘格拍摄3张就可以得到3个H，如果用张正友标定法，就可以得到6个约束方程，可以求解B矩阵（对称矩阵）的6个未知参数，进而通过Cholesky分解求解出内参矩阵A的参数。

相机下的单应性（Homography）变换，可以简单的理解为它用来描述物体在世界坐标系和像素坐标系之间的位置映射关系。对应的变换矩阵称为单应性矩阵。

#### 3.2图像校正

用单应矩阵进行图像矫正的例子如下图所示，最少需要四个对应点对（后面会给出原因）就可以实现。

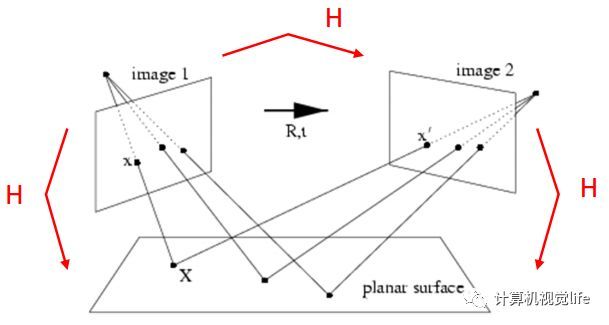


##### 3.3视角变换

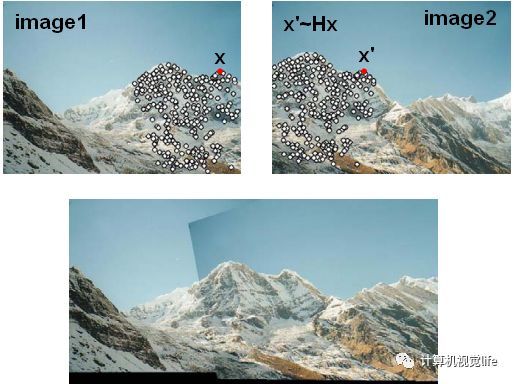
##### 单应矩阵用于视角变换的例子如下图所示，可以方便地将左边普通视图转换为右图的鸟瞰图。

##### 3.4图像拼接

既然单应矩阵可以进行视角转换，那我们把不同角度拍摄的图像都转换到同样的视角下，就可以实现图像拼接了。如下图所示，通过单应矩阵H可以将image1和image2都变换到同一个平面。

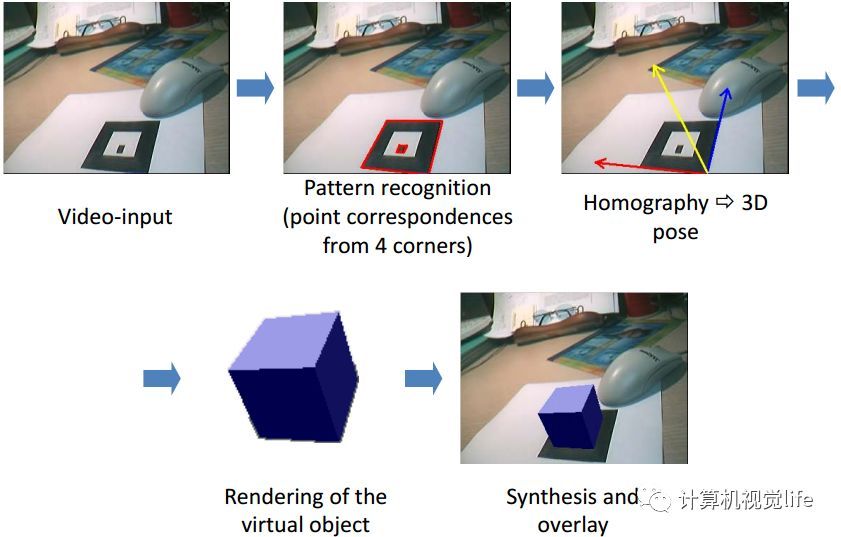


单应矩阵用于图像拼接的例子如下所示。



#### 3.5 增强现实（AR）

平面二维标记图案（marker）经常用来做AR展示。根据marker不同视角下的图像可以方便的得到虚拟物体的位置姿态并进行显示，如下图所示。



因此生成器的训练步骤如下：

## 4.代码应用

h, status = cv2.findHomography(pts\_src, pts\_dst)

im\_dst = cv2.warpPerspective(im\_src, h, size)

import cv2

import numpy as np

if \_\_name\_\_ == '\_\_main\_\_' :

# Read source image.

im\_src = cv2.imread('book2.jpg')

# Four corners of the book in source image

pts\_src = np.array([[141, 131], [480, 159], [493, 630],[64, 601]])

# Read destination image.

im\_dst = cv2.imread('book1.jpg')

# Four corners of the book in destination image.

pts\_dst = np.array([[318, 256],[534, 372],[316, 670],[73, 473]])

# Calculate Homography

h, status = cv2.findHomography(pts\_src, pts\_dst)

# Warp source image to destination based on homography

im\_out = cv2.warpPerspective(im\_src, h, (im\_dst.shape[1],im\_dst.shape[0]))

# Display images

cv2.imshow("Source Image", im\_src)

cv2.imshow("Destination Image", im\_dst)

cv2.imshow("Warped Source Image", im\_out)

cv2.waitKey(0)