Programming Assignment 2
使用平面单应矩阵的增强现实技术

在这项任务中，你将使用平面单应矩阵一步一步地实现一个AR应用。在我们开始实施之前，我们将引导你了解平面单应矩阵的理论。在编程部分，你将首先学习如何找到两幅图像之间的点对应关系，并利用这些点来估计它们之间的匹配关系。利用这种匹配关系，你将对图像进行扭曲，最后实现你自己的AR应用。

1 说明

1. **诚信与协作：**我们鼓励学生以小组形式工作，但每个学生必须提交自己的作品。如果你以小组形式工作，请在你的报告中写明合作者的名字。代码不应该被分享或复制。除非允许，否则请不要使用外部代码。抄袭是被强烈禁止的，并可能导致本课程的不及格。

2. **尽早开始!** 这是个比任务1难得多的任务。

3. **问题：**如果你有任何问题，请先在Piazza上查看。其他学生可能也遇到过同样的问题，而且可能已经解决了。如果没有，请在讨论区发表你的问题。教学人员会尽快回复。

4. **写作：**你的文章应该主要由三部分组成：（i）你对理论问题的回答，（ii）每个步骤的结果图像，以及（iii）对实验的讨论。请注意，我们不接受你在本作业中的手写扫描件。请将你的理论问题的答案和实验的讨论打成电子版。

5. **提交：**创建一个压缩文件，<andrew-id>.zip，由你的文章、你的Python实现（包括辅助函数）和你的实现、结果组成。请确保删除data/文件夹、loadVid.py、helper.py和任何其他你生成的临时文件。你最终上传的文件应该按照这个布局排列：

- <AndrewId>.zip
  - <AndrewId>.pdf
  - python/
    - ar.py
    - briefRotTest.py
    - HarryPotterize.py
    - matchPics.py
    - planarH.py
    - yourHelperFunctions.py
  - result/
    - ar.avi
  - ec/
    - ar_ec.py
    - panorama.py
    - the images required for generating the results.

请确保你在将压缩文件上传到Canvas之前遵守上述的提交规则。违反这一提交规则的作业将被扣去总分的10%。

6. **文件路径：** 请确保你使用的任何文件路径是相对的，而不是绝对的。不是 `cv2.imread('/name/Documents/subdirectory/hw2/data/xyz.jpg')` 而是 `cv2.imread('../data/xyz.jpg')`.

2 单应矩阵

使用平面单应矩阵进行形变

回顾一下，平面单应矩阵是一种形变操作（它是像素坐标从一个相机帧到另一个相机帧的映射），它对现实世界中位于平面上的点做了一个基本假设。在这个特定的假设下，平面上的点在一个视图中的像素坐标可以通过单应矩阵**H**直接映射到相同点在另一个相机视图中的像素坐标。

$$\mathbf{x_1} \equiv \mathbf{H}\mathbf{x_2} \tag{1}$$

$\equiv$ 符号代表恒等于。点 $\mathbf{x}_1$ 和 $\mathbf{x}_2$ 处于齐次坐标中，这意味着它们有一个额外的维度。如果 $\mathbf{x}_1$ 是一个 3D 向量 $\begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T$, 它表示的是一个 2D point $\begin{bmatrix} \frac{x_i}{z_i} & \frac{y_i}{z_i} \end{bmatrix}^T$ (称为非齐次坐标或异构坐标)。这个额外的维度是一个数学上的便利，可以用一个简洁的矩阵形式来表示变换（如平移、旋转、缩放等）。$\equiv$ 表示该方程对一个尺度系数是正确的。

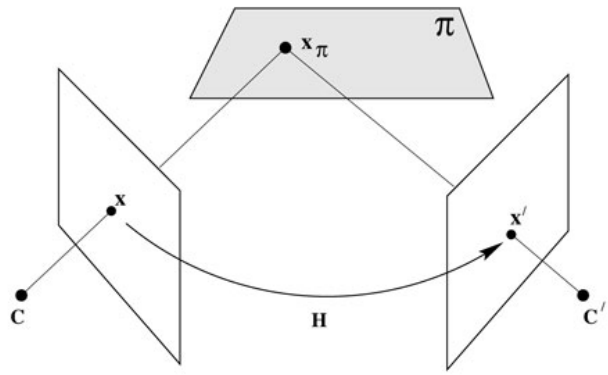

图1：一个单应性矩阵 $\mathbf{H}$ 连接相机 C 和 C' 在平面 π 上所有点 $\mathbf{x}_\pi$ 的视图 $\mathbf{x}$ 和 $\mathbf{x}'$，分别满足 $\mathbf{x}' = \mathbf{H}\mathbf{x}$.
[From Hartley and Zisserman]

直接线性变换

投影几何中一个非常常见的问题通常是 $\mathbf{x} \equiv \mathbf{A}\mathbf{y}$, 其中 $\mathbf{x}$ 和 $\mathbf{y}$ 已知向量, 而 $\mathbf{A}$是一个包含未知数的矩阵，需要解决。给定两幅图像中的匹配点，我们的单应关系显然就是这样一个问 题 的 例 子 。注意，等式仅在一定比例下成立 (这意味着方程组的形式为 $\mathbf{x} = \lambda\mathbf{H}\mathbf{x}'$),这就是为什么我们不能使用普通的最小二乘解，比如你过去可能用过的解联立方程。解决这类问题的标准方法称为直接线性变换，我们将方程改写为适当的齐次方程，然后在标准最小二乘意义下求解。因为这个过程涉及到分解H矩阵的结构，所以它是将问题转化为一组线性方程组，从而使其得名。

Q2.1对应 (15 points)

假设 $\mathbf{x_1}$为图像中的一组点， $\mathbf{x_2}$ 为另一摄像机拍摄的图像中的对应点。假设存在单应性 $\mathbf{H}$ 使得：

$$\mathbf{x_1^i} \equiv \mathbf{H}\mathbf{x_2^i} \quad (i \in \{1 \dots N\})$$

其中 $\mathbf{x^i}_{\mathbf{1}} = \begin{bmatrix} x_1^i & y_1^i & 1 \end{bmatrix}^T$ 在齐次坐标系中, $\mathbf{x^i_1} \in \mathbf{x_1}$ ， $\mathbf{H}$ 是一个 3×3 的矩阵
对于每个点对，这种关系可以改写为

$$\mathbf{A_i}\mathbf{h} = 0$$

其中 $\mathbf{h}$ 是一个从 $\mathbf{H}$ 重塑的列向量， $\mathbf{A_i}$ 是由点 $\mathbf{x_1^i}$ 和 $\mathbf{x_2^i}$ 得出的矩阵这可以帮助从给定的点对应关系中计算出 $\mathbf{H}$ 。

1. $\mathbf{h}$ 有多少个自由度? (3 points)
2. 求解 $\mathbf{h}$ 需要多少个点对? (2 points)

3. 推导出 $\mathbf{A_i}$. (5 points)
4. 当求解 $\mathbf{Ah} = 0$时, 实质上你是在试图找出存在于 $\mathbf{A}$ 的无效空间中的 $\mathbf{h}$ 。这意味着对于 $\mathbf{h}$ 会有一些非平凡的解使得乘积 $\mathbf{Ah}$ 结果为 0。
   $\mathbf{h}$ 的平凡解是什么？$\mathbf{A}$ 矩阵是满秩吗？为什么是/为什么不是？它对奇异值有什么影响？它对奇异向量有什么影响？ (5 points)

使用矩阵分解计算单应矩阵

单应矩阵$\mathbf{H}$可以将一组点（在齐次坐标中）转换为另一组点。在本项目中，我们将使用特征匹配获得相应的点坐标，然后需要计算单应性矩阵。你已经在问题2中导出了 $\mathbf{Ax} = \mathbf{0}$ 。在本节中，我们将研究如何使用两种方法求解此类方程，其中任何一种都可以用于后续的赋值问题。

特征值分解

求解 $\mathbf{Ax} = \mathbf{0}$ 的一种方法是计算 $\mathbf{A}$ 的特征值和特征向量。0对应的特征向量就是这个问题的答案。举个例子:

$$\mathbf{A} = \begin{bmatrix} 3 & 6 & -8 \\ 0 & 0 & 6 \\ 0 & 0 & 2 \end{bmatrix}$$

使用 `numpy.linalg` 的函数 `eig`, 我们得到以下特征值和特征向量:

$$V = \begin{bmatrix} 1.0000 & -0.8944 & -0.9535 \\ 0 & 0.4472 & 0.2860 \\ 0 & 0 & 0.0953 \end{bmatrix}$$
$$D = \begin{bmatrix} 3 & 0 & 2 \end{bmatrix}$$

这里，$\mathbf{V}$ 的列是特征向量，$\mathbf{D}$ 中的每个对应元素都是特征值。我们注意到有一个0的特征值，与之对应的特征向量就是方程 $\mathbf{Ax} = \mathbf{0}$的解。

$$\mathbf{Ax} = \begin{bmatrix} 3 & 6 & -8 \\ 0 & 0 & 6 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} -0.8944 \\ 0.4472 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

奇异值分解

矩阵 $\mathbf{A}$ 的奇异值分解（SVD）表示为:

$$\mathbf{A} = U\Sigma V^T$$

这里，U 是一个列向量的矩阵，称为 "左奇异向量"。同样地，V 被称为 "右奇异向量"。矩阵 Σ 是一个对角线矩阵。每个对角线元素 σ_i 被称为 "奇异值"，这些奇异值按大小顺序排列。在我们的例子中，它是一个 9×9 的矩阵。

- 如果 $\sigma_9 = 0$，则系统精确确定，存在单应性，所有点精确匹配。
- 如果 $\sigma_9 \geq 0$，系统是超定的。 存在单应性，但并非所有点都完全匹配（它们符合最小二乘误差）。 该值代表拟合优度。
- 通常情况下，你至少会有四个对应关系。如果没有，那么这个系统是不确定的。我们不会在这里处理这些问题。

U 的列是 $\mathbf{A}\mathbf{A}^T$ 的特征向量。V 的列是 $\mathbf{A}^T\mathbf{A}$. 的特征向量，我们可以利用这一事实来求解方程 $\mathbf{Ah} = \mathbf{0}$ 中的 $\mathbf{h}$ 。利用这一知识，让我们重新表述求解 $\mathbf{Ax} = \mathbf{0}$的问题。我们希望在最小二乘意义上最小化解的误差。理想情况下，乘积 $\mathbf{Ah}$ 应为0。因此，平方和误差可写成:

$$
\begin{aligned}
f(\mathbf{h}) &= \frac{1}{2}(\mathbf{Ah} - \mathbf{0})^T(\mathbf{Ah} - \mathbf{0}) \\
&= \frac{1}{2}(\mathbf{Ah})^T(\mathbf{Ah}) \\
&= \frac{1}{2}\mathbf{h}^T\mathbf{A}^T\mathbf{Ah}
\end{aligned}
$$

使该误差相对于 $\mathbf{h}$ 最小化，我们得到:

$$
\begin{aligned}
\frac{d}{d\mathbf{h}} f &= 0 \\
\implies \frac{1}{2}(\mathbf{A}^T\mathbf{A} + (\mathbf{A}^T\mathbf{A})^T)\mathbf{h} &= 0 \\
\mathbf{A}^T\mathbf{Ah} &= 0
\end{aligned}
$$

这意味着$\mathbf{h}$的值等于对应于零特征值的特征向量（或在噪声情况下最接近零）。因此，我们选择 $\mathbf{A}^T\mathbf{A}$的最小特征值，即 Σ 中的 σ_9 的最小二乘解是相应的特征向量（在矩阵 $\mathbf{V}$的第9列中）。

3 计算平面单应性矩阵

特征检测与匹配

在确定图像对之间的单应性矩阵之前，我们需要在两个图像之间找到对应的点对。但我们如何得到这些点呢？一种方法是手动选择它们，这既繁琐又无效。CV方法是在图像对中

查找兴趣点并自动匹配它们。为了能够做一些很酷的事情，我们不会在这里重新实现特征检测器或描述符，而是使用 python 模块。兴趣点检测器（例如 Harris、SIFT、SURF 等）的目的是在我们提取特征描述符（例如 MOPS 等）的图像中找到特定的显着点。这些描述符试图以尽可能简洁但描述性的方式总结特征点周围的图像内容（对于许多计算机视觉任务，通常在表示和计算复杂性之间进行权衡；你可以拥有一个非常高维的特征描述符，将确保你获得良好的匹配，但计算它可能会非常昂贵）。因此，匹配是一项尝试在与当前描述符最匹配的新图像上计算得到的描述符列表中找到描述符的任务。这可能像两个描述符之间的欧几里得距离一样简单，也可能更复杂，具体取决于描述符的组成方式。出于本练习的目的，我们将使用广泛使用的 FAST 检测器和 Brief 描述符。

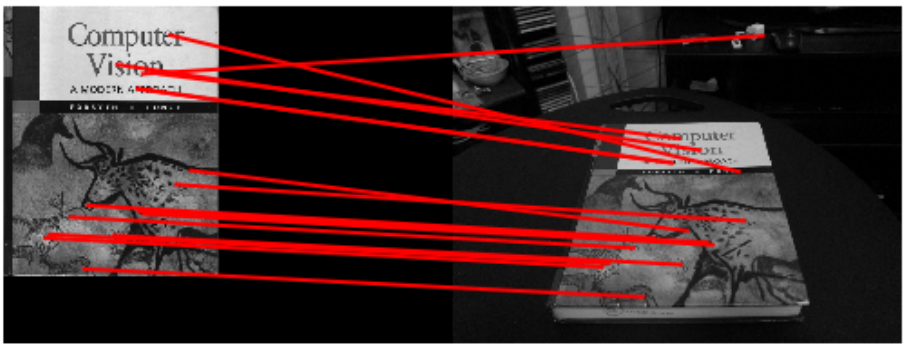

图 2：一些匹配的 FAST 特征点与BRIEF描述符

Q3.1 FAST 检测器 (5 points)

FAST检测器与你在课堂上看到的Harris角点检测器有什么不同？(你可能需要在网上查找FAST检测器。)你能评论一下它与Harris角点检测器相比的计算性能吗？

Q3.2 BRIEF 描述符 (5 points)

Brief 描述符与你在课堂中看到的滤波器组有何不同？你能使用其中任何一个滤波器组作为描述符吗？

Q3.3 匹配方法 (5 points)

Brief 描述符属于二进制描述符的类别。 在这种描述符中，对应于检测到的特征点的图像区域被表示为 1 和 0 的二进制字符串。 用于此类描述符的常用度量称为汉明距离。 请在线搜索以了解汉明距离和最近邻，并描述如何使用它们将兴趣点与简要描述符进行匹配。 在我们的设置中，汉明距离距离比更传统的欧几里得距离测量有什么好处？

有什么好处？

Q3.4 特征匹配 (10 points)

请实现一个函数：

```
matches, locs1, locs2 = matchPics(I1, I2)
```

其中 I1 和 I2 是你想要匹配的图像。locs1 和 locs2 是包含特征点的 x 和 y 坐标的 $N \times 2$ 矩阵。 matches 是一个 $p \times 2$ 矩阵，其中第一列是 I1 中特征的索引，类似地，第二列包含与I2相关的索引。使用提供的辅助函数 corner detection 来计算特征，然后使用提供的辅助函数 computeBrief来建立描述符，最后使用提供的辅助函数 briefMatch 来比较它们。
使用所提供的辅助函数 plotMatches 可视化匹配点，并将结果图像包含在你的记录中。图2中展示了一个示例。

两幅图像之间的匹配数量根据角检测中使用的参数 sigma 以及 briefMatch中的值比率而变化。你应该改变这些以获得最佳结果。图2所示的例子是sigma $= 0.15$，ratio $= 0.65$。

我们为你提供以下辅助函数:

```
locs = corner detection(img, sigma)
desc, locs = computeBrief(img, locs)
matches = briefMatch(desc1, desc2)
plotMatches(im1, im2, matches, locs1, locs2)
```

locs 是一个 $N \times 2$ 的矩阵，其中每一行代表一个特征点的位置 (x, y) 。请注意，有效输出特征点的数量可以少于输入特征点的数量。 desc 是兴趣点的 BRIEF 描述符的相应矩阵。

<u>在你的文章中：</u>如上所述，使用所提供的辅助函数 plotMatches 来可视化你的匹配点，并在你的文章中包括所产生的图像。

Q3.5 BRIEF 和 Rotations (10 points)

让我们研究一下 BRIEF 如何与 rotations一起工作。写一个脚本 briefRotTest.py:

- 获取 cv cover.jpg 并以10度为增量将其与自身旋转匹配。 [提示：使用 scipy.ndimage.rotate]

- 存储每个方向的匹配计数直方图
- 使用 `matplotlib.pyplot.bar` 绘制直方图

<u>在你的文章中：</u> 包括在三个不同方向上的特征匹配结果的可视化。解释为什么你认为简要描述符的行为方式是这样的。

单应矩阵计算

Q3.6 计算单应矩阵 (15 points)

编写一个函数 `computeH`，从一组匹配的点对中估计平面单应矩阵。

```
H2to1 = computeH(x1, x2)
```

`x1` 和 `x2` 是 $N \times 2$ 矩阵，包含两幅图像之间点对的坐标 (x, y)。 `H2to1` 应该是从图像 2 到图像 1 的最小平方意义上的最佳单应性的 3×3 矩阵。 `numpy.linalg` 函数 `eig` 或 `svd` 将有助于获得特征向量（详见本讲义第二节）。

单应矩阵归一化

归一化提高了解决方案的数值稳定性，你应该始终将你的坐标数据归一化。归一化有两个步骤：

1. 将各点的平均值平移到原点。
2. 缩放各点，使其与原点的最大距离为 $\sqrt{2}$。

这是一个线性变换，可以写成如下：

$$\widetilde{\mathbf{x}}_1 = \mathbf{T}_1\mathbf{x}_1$$
$$\widetilde{\mathbf{x}}_2 = \mathbf{T}_2\mathbf{x}_2$$

其中 $\widetilde{\mathbf{x}}_1$ 和 $\widetilde{\mathbf{x}}_2$ 是 $\mathbf{x_1}$ 和 $\mathbf{x_2}$ 归一化后的齐次坐标。$\mathbf{T}_1$ 和 $\mathbf{T}_2$ 是 3×3 矩阵。

通过 computeH 从 $\widetilde{\mathbf{x}}_2$ 到 $\widetilde{\mathbf{x}}_1$ 计算出的单应矩阵 $\mathbf{H}$，满足：

$$\widetilde{\mathbf{x}}_1 = \mathbf{H}\widetilde{\mathbf{x}}_2$$

通过用 $\mathbf{T}_1\mathbf{x}_1$ 和 $\mathbf{T}_2\mathbf{x}_2$ 来替换 $\widetilde{\mathbf{x}}_1$ 和 $\widetilde{\mathbf{x}}_2$ 我们可以得到：

$$\mathbf{T}_1\mathbf{x}_1 = \mathbf{H}\mathbf{T}_2\mathbf{x}_2$$
$$\mathbf{x}_1 = \mathbf{T}_1^{-1}\mathbf{H}\mathbf{T}_2\mathbf{x}_2$$

Q3.7 归一化后的单应性矩阵 (10 points)

实现函数 `computeH_norm`:

```
H2to1 = computeH_norm(x1, x2)
```

该函数应归一化 `x1` 和 `x2` 中的坐标，并调用如上所述的 `computeH(x1, x2)` 。

RANSAC

算法通常可以将任何模型适用于噪声数据。你将为求解图像之间的（平面）单应矩阵来使用它。计算单应矩阵至少需要4个点对。

Q3.8 实现用于计算单应矩阵的RANSAC算法 (25 points)

编写函数:

```
bestH2to1, inliers = computeH_ransac(x1, x2)
```

其中 `bestH2to1` 应该是在RANSAC时找到的内点最多的单应矩阵 $\mathbf{H}$ 。$\mathbf{H}$ 是个单应矩阵，因此假如 $\mathbf{x}_2$ 是 `x2` 中的一个点并且 $\mathbf{x}_1$ 是 `x1`中的对应点，那么 $\mathbf{x}_1 \equiv \mathbf{H}\mathbf{x}_2$。`x1` 和 `x2` 是包含匹配点的 $N \times 2$ 矩阵。`inliers` 是一个长度为N 的向量，在作为共识集一部分的匹配处为 1，在其他地方为 0。 使用`computeH_norm` 计算单应矩阵。

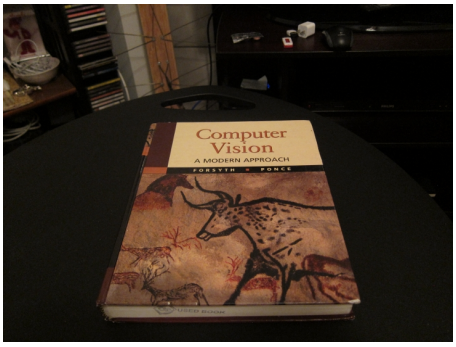

图 3：测试书

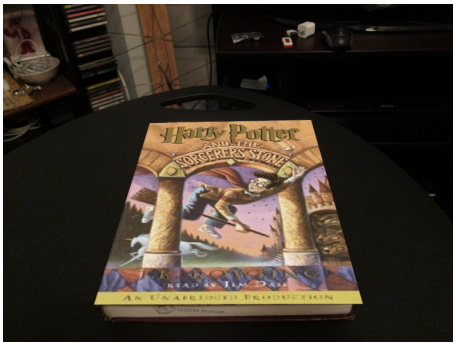

图 4：哈利波特化的测试书

自动单应矩阵估计和形变

Q3.9 将它们放在一起 (10 points)

写一个 `HarryPotterize.py` 脚本:

1. 读取 `cv_cover.jpg`，`cv_desk.png` 和 `hp_cover.jpg`。
2. 使用 `MatchPics` 和 `computeH ransac` 自动计算单应矩阵。
3. 使用计算的单应矩阵来使 `hp_cover.jpg` 形变为 `cv_desk.png` 的图片尺寸，你可以使用 `skimage` 的函数 `skimage.transform.warp` 或者 OpenCV 的函数 `cv2.warpPerspective`。
4. 此时你应该注意到，尽管图像被扭曲到正确的位置，但它并没有填满与书相同的空间。你认为为什么会发生这种情况？你将如何修改 `hp_cover.jpg`来解决这个问题？
5. 实现函数：

```
composite_img = compositeH( H2to1, template, img )
```

现在，将此形变图像与桌面的图像合成，如图4所示

<u>在你的文章中:</u> 展示最终的图片，即 `composite_img` 。由你的脚本`HarryPotterize.py` 生成。此外，讨论 4. 中提出的问题。

4 创建增强现实应用程序

Q4.1 合并视频 (20 points)

现在有了这些代码，你就可以创建你自己的增强现实应用了。你要做的是将视频 `ar _ source.mov` 在视频`book.mov`上进行哈利波特化。具体来说，你要在`book.mov` 的每一帧中追踪测试书本，并将`ar _ source.mov` 的每一帧叠加到`book.mov`中的书上。请写一个脚本 `ar.py` 来实现这个 AR 应用，并将你的结果视频作为 `ar.avi` 在 `result/` 目录中。你可以使用我们提供的函数 `loadVid` 来加载视频。如果你能正确地把视频拼接起来，你将得到全部分数。参见图5，了解最终视频应该是什么样子的一个示例帧。

请注意，这本书和我们提供的视频有着不同的长宽比（图像宽度与图像高度的比例）。你必须裁剪每一帧，以适应书的封面。还应该裁剪每一帧，以便在最后的输出中只使用其中心区域。见图6的例子。

另外，视频 `book.mov` 中只有物体的平移。如果你想考虑物体的旋转、缩放等，你必须选择一个更好的特征点表示（如ORB）。

图5：在一个移动目标上渲染视频

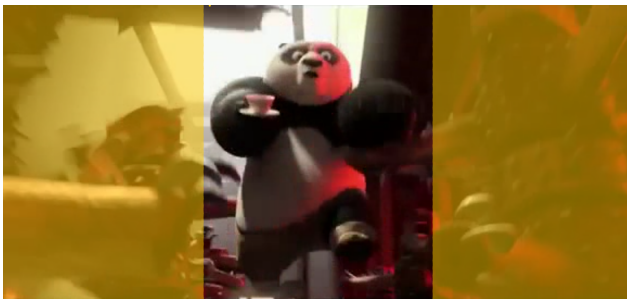

图6：裁剪掉每一帧的黄色区域，以符合书的长宽比。

5 全景图

Q5.1x: 创建一个简单的全景图 (10 points)

使用自己的相机拍摄两张照片，尽可能用纯旋转分开，然后使用`panorama.py`.构建全景。请确保图像中的对象距离足够远，以便不存在视差影响。您可以使用python模块`cpselect`来选择每张图片上的匹配点，或者使用一些自动方法。提交原始图像、最终的全景图，以及加载图像并组装全景图的脚本`panorama.py`。我们提供了两张图片供你开始使用(`data/pano_left.png`和`data/pano_right.png`)。在提交这个作业时，请使用你自己的图片。

<u>在你的文章中：</u> 包含你的原始图像和全景图的结果图像。见下面的图7-9为例。

图7：原始图像1（左）

图8：原始图像2（右）

图9：全景图