第一章　基本原理

1.1 特征点提取和匹配

特征点提取方法有很多，主流的有 Harris，SIFT，FAST 等，在图像上提取出有辨识度的点，通常是角点。对于每个角点生成一个特征描述子来区分该角点与其他角点，常用的描述子也有很多：Brief、SIFT、ORB、SURF 等等。在不同视角下如果角点特征值相似度高，则可认为两个角点相匹配，此时可得到同一个空间点在两个不同的图像平面下的坐标，知道这些坐标对于后续参数的求解至关重要。

1.2 求解本质矩阵 E

本质矩阵约束两个图像平面的映射关系，而且可从 E 分解得到两相机间的旋转和平移的变换关系，对于恢复运动参数必不可少，所以第一步先求解本质矩阵 E。空间点投影到两图像平面的情形如图一所示，相机O_r坐标系中任意一点P_r可通过旋转平移转换到相机O_l的坐标系。

$$P_l = RP_r + T$$

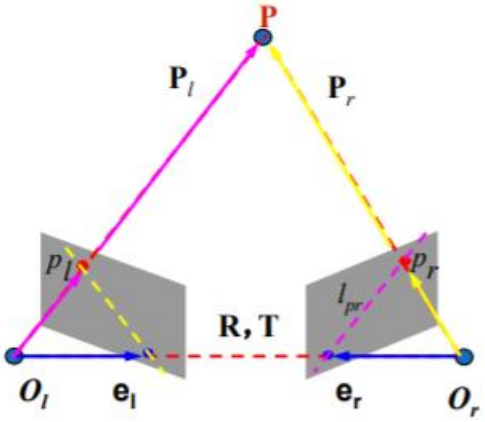
显然O_lP，O_rP，O_lO_r共面，通过极几何约束可以得到：

$$P_l^T(T \times PP_r) = 0$$

令$E = T \times R$，因为在齐次坐标系下的$p_r \cong P_r$，$p_l \cong P_l$，有：

$$p_l^T E p_r = 0$$

至此得到足够简洁的约束。试想当p_l^T确定，E已知是，p_r满足一个线性约束，即一个平面的点坐标已知，则另一个平面上的匹配点必然在一条直线上。可见E确实约束了两图像平面的映射关系。



图一 空间点投影到两图像平面

1.3 恢复运动参数

SFM 的一个难点在于相机的运动参数未知，索性可以从本质矩阵 E 中通过 SVD 分解得到两个相机视角相对的旋转矩阵 R 和平移向量 T，不过由于未知空间坐标之间的实际距离，重建出来的模型只有形状，没有尺寸。

恢复空间点坐标

一个空间点投影到图像平面是有一个投影矩阵$P_{3\times4}$的(X与x均为齐次坐标)

$$X = Px$$

而P可通过相机本身自带的内部参数K和运动参数R、T求得。

$$P = K[R|T]$$

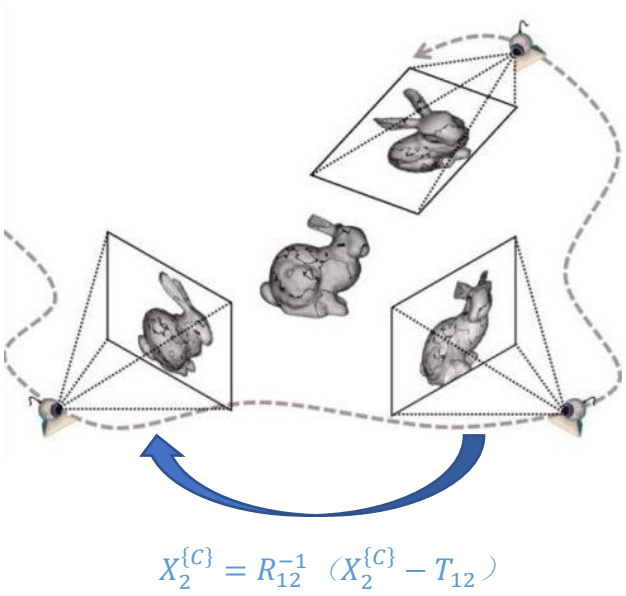
所以对于两个平面之间匹配的点，每一组对应的平面坐标$x_1 = \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix}$，$x_2 = \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}$可列出方程

$$\begin{cases} (u_1p_1^{31} - p_1^{11})X + (u_1p_1^{32} - p_1^{12})Y + (u_1p_1^{33} - p_1^{13})Z = p_1^{14} - u_1p_1^{34} \\ (v_1p_1^{31} - p_1^{21})X + (v_1p_1^{32} - p_1^{22})Y + (v_1p_1^{33} - p_1^{23})Z = p_1^{24} - v_1p_1^{34} \\ (u_2p_2^{31} - p_2^{11})X + (u_2p_2^{32} - p_2^{12})Y + (u_2p_2^{33} - p_2^{13})Z = p_2^{14} - u_2p_2^{34} \\ (v_2p_2^{31} - p_2^{21})X + (v_2p_2^{32} - p_2^{22})Y + (v_2p_2^{33} - p_2^{23})Z = p_2^{24} - v_2p_2^{34} \end{cases}$$

显然，借助最小二乘法，每对匹配点都能解出一个对应的空间坐标。而对于一个图像序列，往往多幅图像中都有同一个空间点，所以能得到更多方程，当有l张图像有同一

匹配点时，可列出$2l$个方程。

至此还有一个问题，运动参数R、T是图片序列两两为一组求解出来的，每组求解过程中假设了其中一张相机坐标系为世界坐标系，所以此时上述方程求解出的空间点并不是在同一坐标系下，此时应该利用图像序列连续的性质，将坐标转换回同一个世界坐标系。示意图如图二。



图二 坐标变换示意图

第二章　设计过程

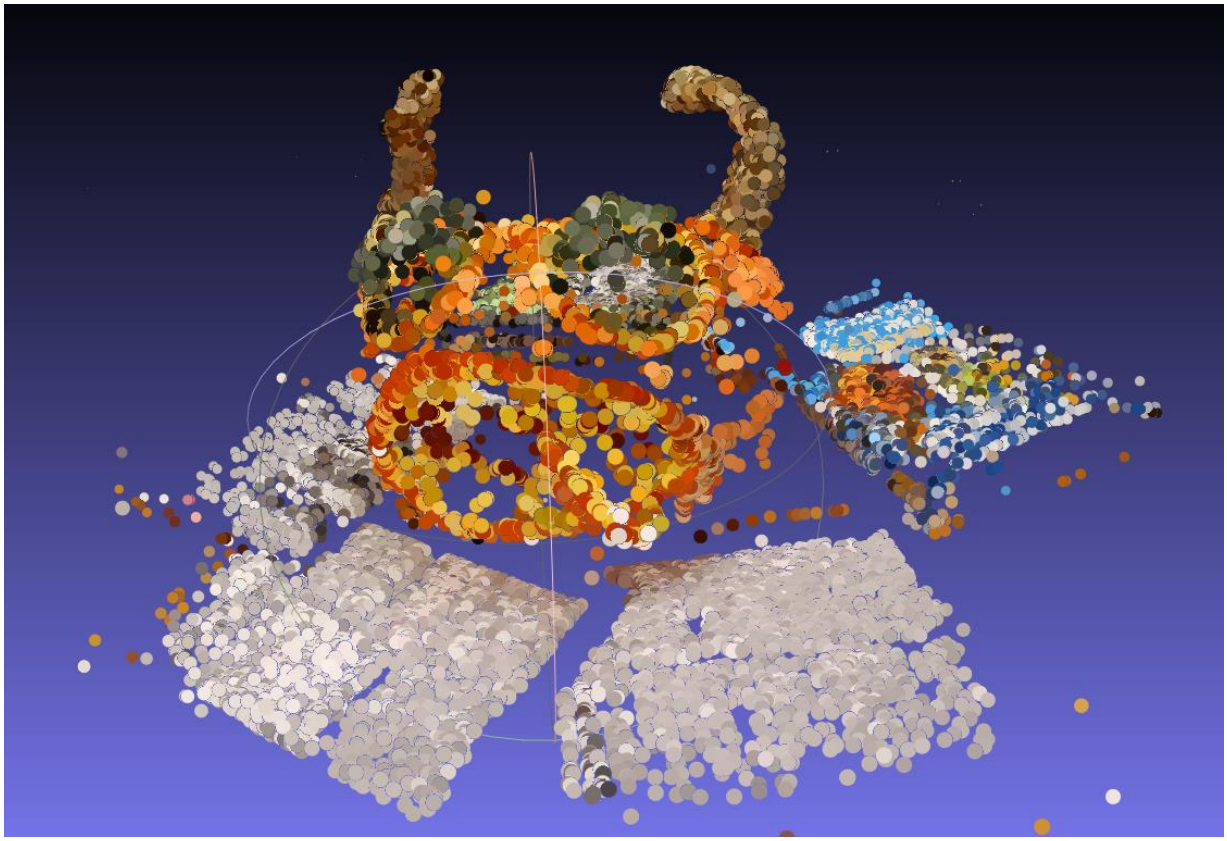
2.1 特征匹配和特征匹配选择

特征点检测选用了 SIFT，因为 SIFT 具有很强的鲁棒性，在不同的尺度、平移、旋转、噪声甚至仿射变换下都有稳定且良好的检测性能。能够保证图像序列视角变换的情况下产生足够的角点。

特征匹配选择的是 SIFT 特征，主要出于计算量考虑，因为在 Opencv 的 SIFT 角点检测函数中会默认输出 SIFT 特征，方便起见直接用该特征进行匹配。

2.2　生成三维点云

将图像序列想象为一个串，每相邻两幅图像生成三维点云，沿着串通过R、T不断转换坐标系，最后转换到的世界坐标系（即第一张图片的相机坐标系）下，如图四所示。



图四 测试图像序列上生成的三维点云（图中所示为部分图像，实际采用了 78 张图像）

重建后首先发现点云有很多噪声，用基于密度的聚类方法去除噪点（孤立点），齐次发现点云是倾斜的，因为点云全融合到第一个相机坐标系下了，而第一个相机是以斜拍的角度拍摄的。这一步用了半人方式，找了同一个平面上的三个点，计算出罗德里斯公式，将点云转正。

2.3 路径规划

2.3.1 A^*算法

对于二维路径规划问题，我们采取了A^*算法来求解最优路径。A^*算法从起点开始遍历四周的位置，并通过构造启发式的代价函数来找到最优位置做为下一个移动的位置，并重复上述步骤直至达到终点。

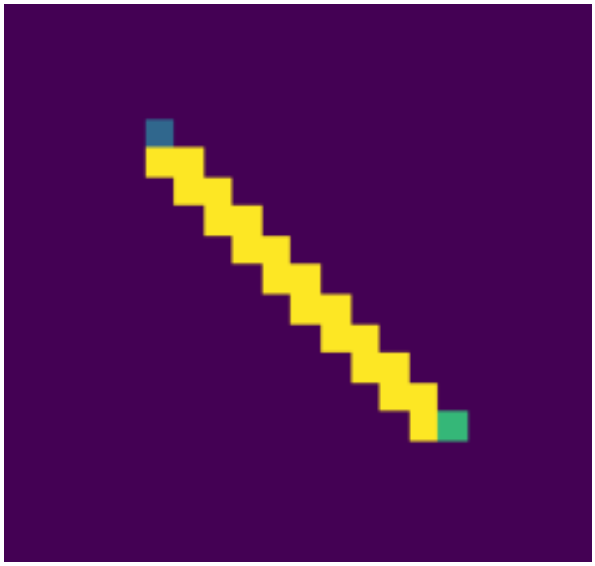


图 5 A^*算法搜索的最短路径

如图所示，限定移动的方向为当前点的上、下、左、右四个方向，图中蓝色与绿色的点分别表示给定的起点与终点，而黄色的点序列则为从起点到终点的最短路径。每到达一个点，A^*算法都会遍历当前点的上下左右四点，找到其中所有未曾访问的点做为候选点，并为每个候选点计算代价函数$f(n)$来确定最优的移动位置。$f(n)$应当使得使算法在保证路径最短的前提下尽可能地以最快的速度找到目的地，具体构造如下：

$$f(n) = g(n) + h(n)$$

$$g(n) = \mathrm{distance}(P_{start}, P_n)$$

$$h(n) = \mathrm{distance}(P_{end}, P_n)$$

其中，P_{start}，P_{end}，P_n分别为起点，终点，与当前点的位置，distance()计算两位置间的距离。在使用A^*算法进行路径规划时，常用的距离为曼哈顿距离，即：

$$distance(P_1, P_2) = |x_1 - x_2| + |y_1 - y_2|$$

当存在障碍物时，A^*算法在遍历可移动点时还会考虑该点是否触碰到了障碍物，并在绕开障碍物的情况下找到最短的路径，如下图所示：

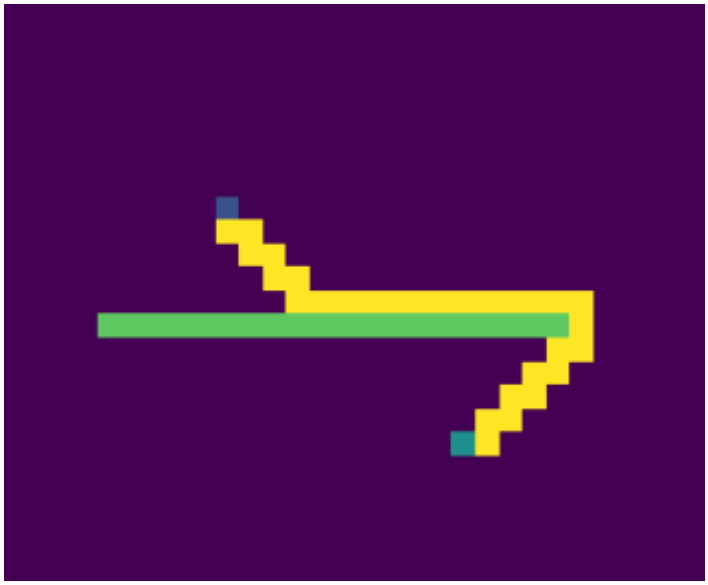
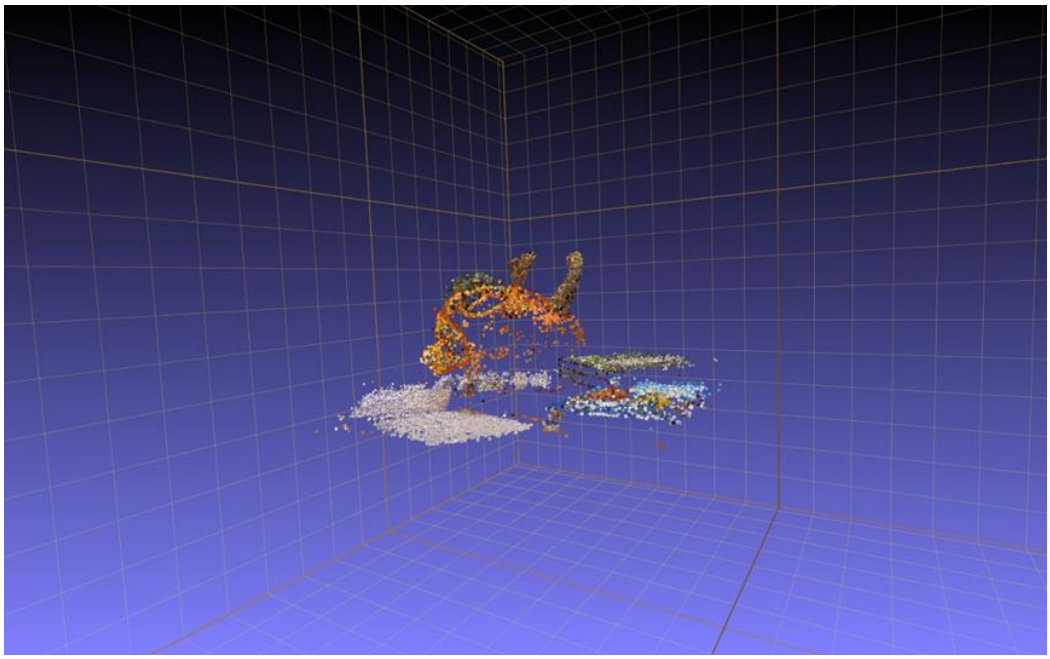
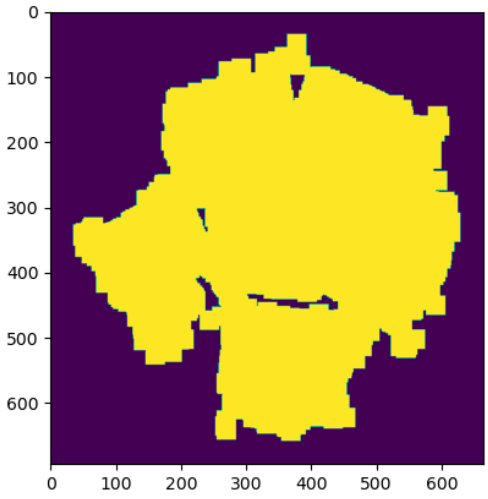


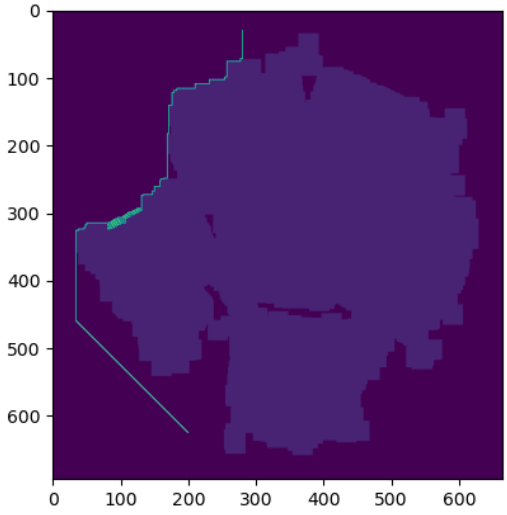
图 5 A^*算法的自动避障

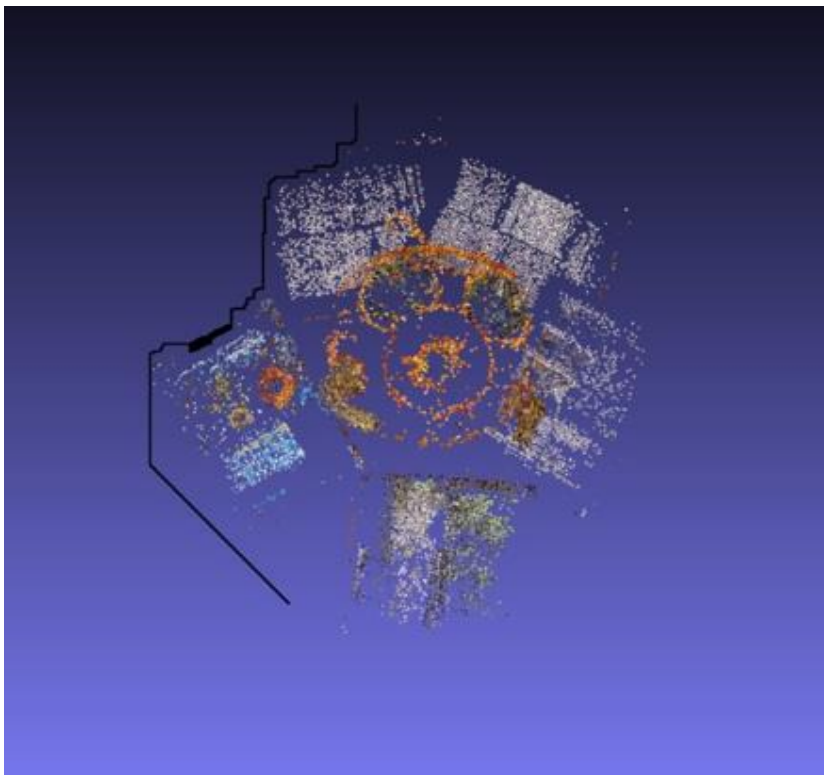
将三维重建后的三维点云投射到二维空间，给定起点与终点后利用A^*算法进行路径规划，即可得到在任意高度均可行的移动路径。由于三维重建与机器的运动均存在一定误差，必须要给路径与障碍物之间留出余量以保证安全。因此，在规划前，可先将形态学中的膨胀处理应用于投影后的二维图像上，膨胀障碍的边界，以便得到更加安全的路径。

第三章　结果展示




0
100
200
300
400
500
600
0
100
200
300
400
500
600


0
100
200
300
400
500
600
0
100
200
300
400
500
600



第四章　改进方向特征点匹配

如果采用逐点特征点匹配，那时间复杂度为$o(N^2)$，耗时随角点数程平方增长，可通过 k-means 方法聚类，建立搜索树，在近邻区域优先匹配，可将时间复杂度降到$o(kN)$。其他方法还有借助哈希表优化匹配。相机旋转矩阵

本文将图像序列看作一个有序的串，相邻两图像之间两两计算运动参数R、T，这称为增量式重建，除此之外还有全局式重建。全局式将图像序列看作一个全连接的网，每张图像对其余所有图像计算R、T，相对于增量式的优点在于：恢复出的空间坐标坐标转换到世界坐标系下时只需进行一次坐标转换，效率高，且不用按顺序添加图片。后续应用

得到小天体的三维模型后，可用最小二乘法解出近似的椭圆方程，用椭圆模拟小天体最大内接椭球体，假设小天体质量均匀分布，则质心与形心重合，从椭球体行心出发的射线介于椭球面与三维重建表面的线段长度为地面高度，由此绘制得高程图。根据高程图可对探测器落点进行选择