

# 计算机视觉 实验四 报告

张俊华 16030199025

郁张超 16030140077

## 一、实验内容

本实验将运用平面扫描立体视觉与光度测量立体视觉的方法，来恢复图像深度，并建立立体图。实验包含三个部分：

1. 光度测量立体视觉（详见讲义第18讲）：给定在不同的已知光照方向下从相同视角拍摄的一组图像，从中恢复物体表面的反照率(albedo)和法线方向(normals)。
2. 平面扫描立体视觉（详见讲义第16讲）：给定同一场景从不同的视角拍摄的两幅校准图像，从中恢复出粗略的深度图。
3. 基于泊松方程重建深度图（详见讲义第18讲）：根据法线图及粗略深度图，恢复出物体每个点的深度，并重建3D网格。

实验分工：

- 光度测量立体视觉：郁张超
- 平面扫描立体视觉：张俊华
- 基于泊松方程重建深度图：张俊华、郁张超

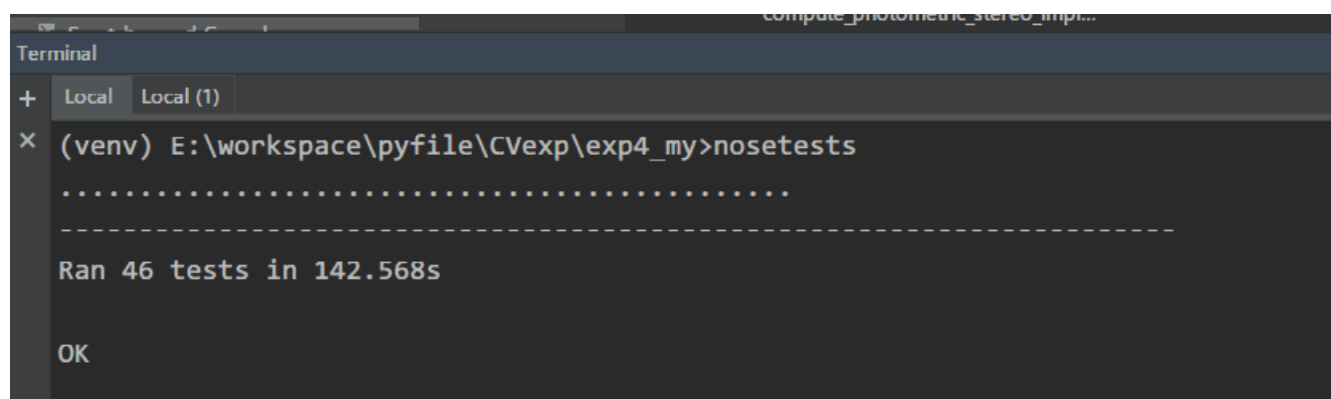
## 二、实验环境

Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:22:17) [MSC v.1500 32 bit (Intel)] on win32

PyCharm 2018.2.4 Build #PY-182.4505.26, built on September 19, 2018 Windows 10 10.0

## 三、实验过程

首先对 `students.py` 中的函数完成实现



```
Terminal
+ Local Local (1)
x (venv) E:\workspace\pyfile\CVexp\exp4_my>nosetests
.....
-----
Ran 46 tests in 142.568s

OK
```

优化：

将函数中部分矩阵运算重构，采用 numpy 的矩阵运算实现，可见有显著的性能提升

```
Local Local (1)
(venv) E:\workspace\pyfile\CVexp\exp4_my>nosetests
.....
-----
Ran 46 tests in 19.277s

OK
```

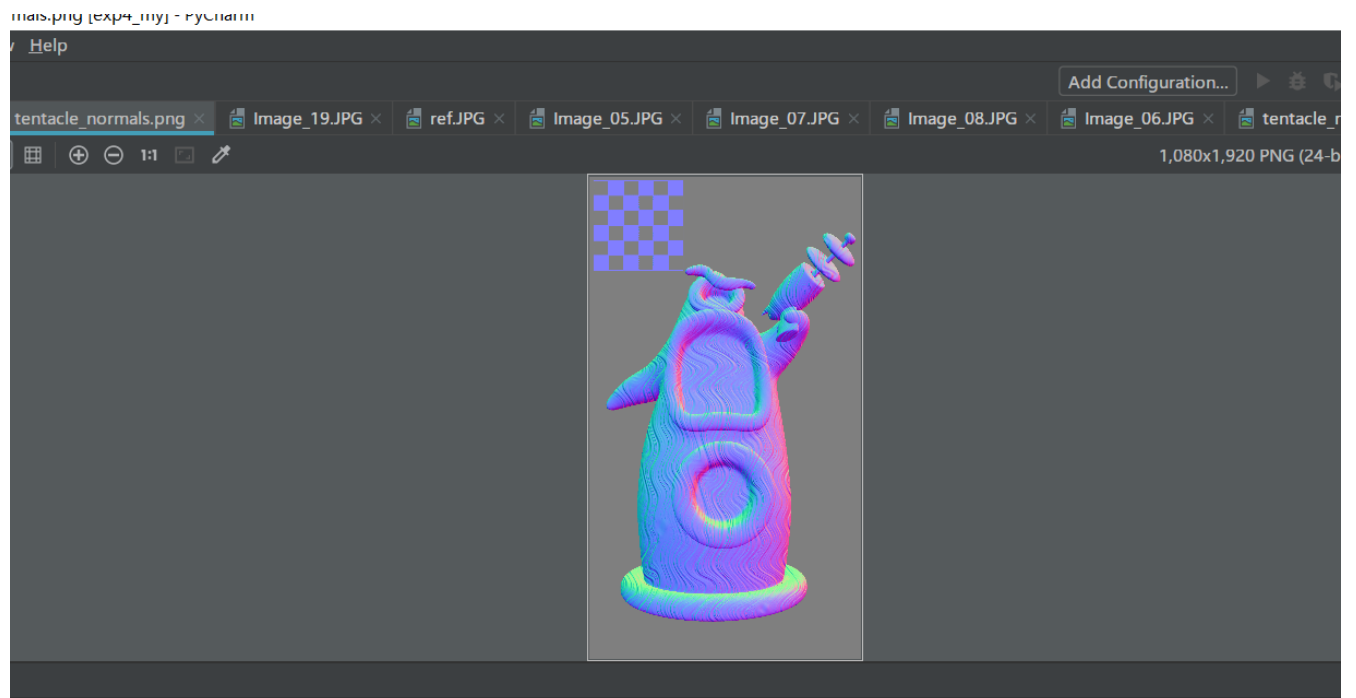
## 光度测量立体视觉测试

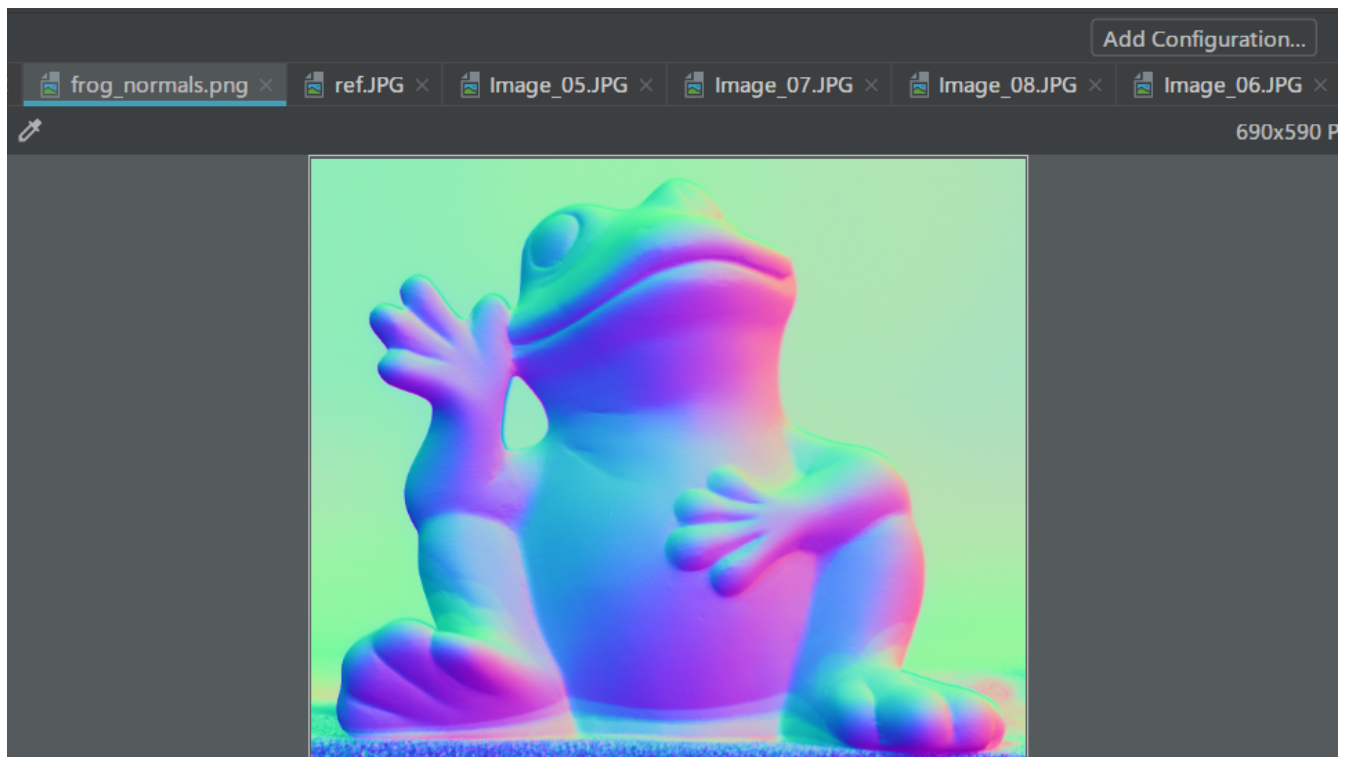
执行

```
1 | python photometric_stereo.py tentacle
```

```
(venv) E:\workspace\pyfile\CVexp\exp4_my>python photometric_stereo.py tentacle
Average RMSE of rerendered image is 0.0200623523872
Saving albedo to output/tentacle_albedo.png
Saving normals visualization to output/tentacle_normals.png
Saving normals to output/tentacle_normals.npy
```

观察生成结果：可见对表面法向量的计算满足测试要求





## 平面扫描立体视觉测试

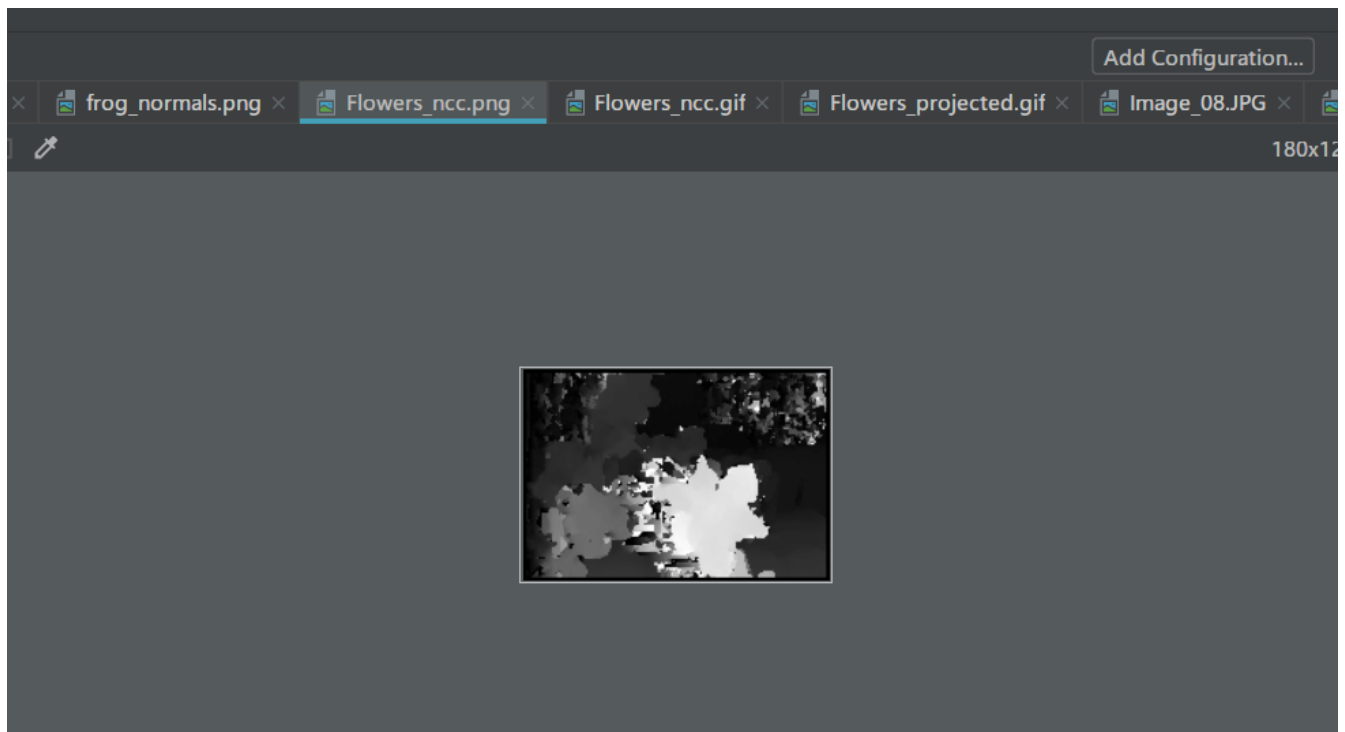
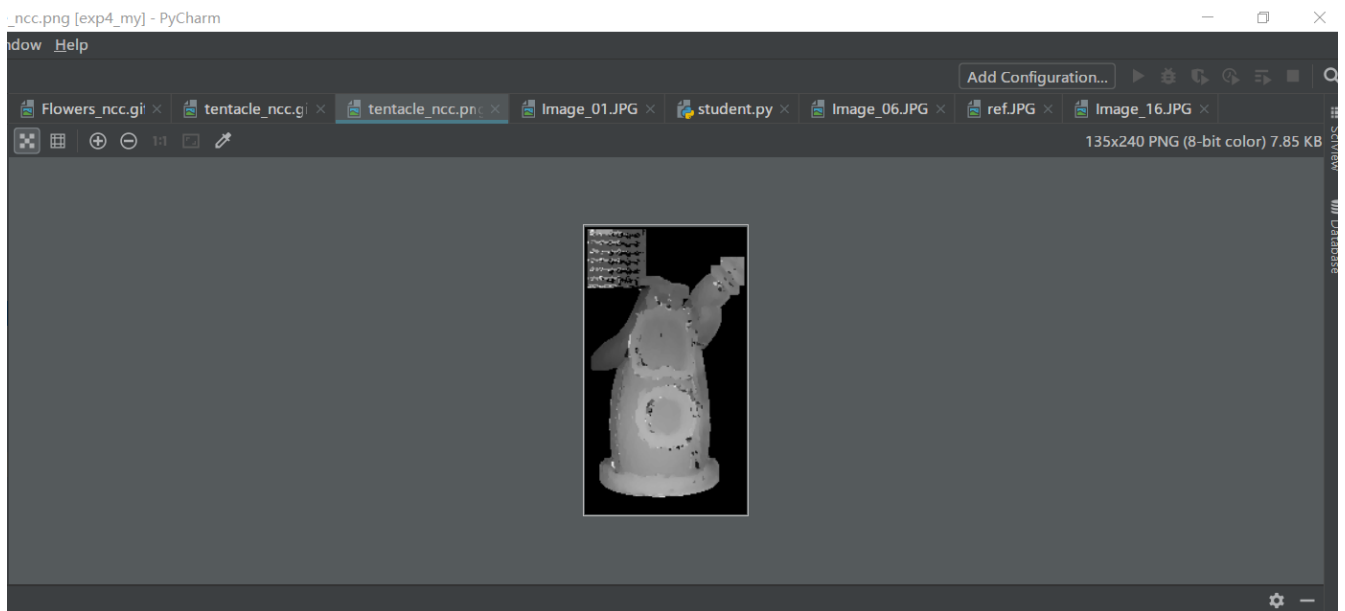
执行

```
1 | > python plane_sweep_stereo.py tentacle
```

运行后观察生成的 gif 深度图：

```
(venv) E:\workspace\pyfile\CVexp\exp4_my>python plane_sweep_stereo.py Flowers
Progress: 99
Plane sweep took 26.3250000477 seconds
Saving NCC to output/Flowers_ncc.png
Saving depth to output/Flowers_depth.npy
```

```
(venv) E:\workspace\pyfile\CVexp\exp4_my>python plane_sweep_stereo.py tentacle
Progress: 99
Plane sweep took 37.0290000439 seconds
Saving NCC to output/tentacle_ncc.png
Saving depth to output/tentacle_depth.npy
```



## 基于泊松方程重建深度图

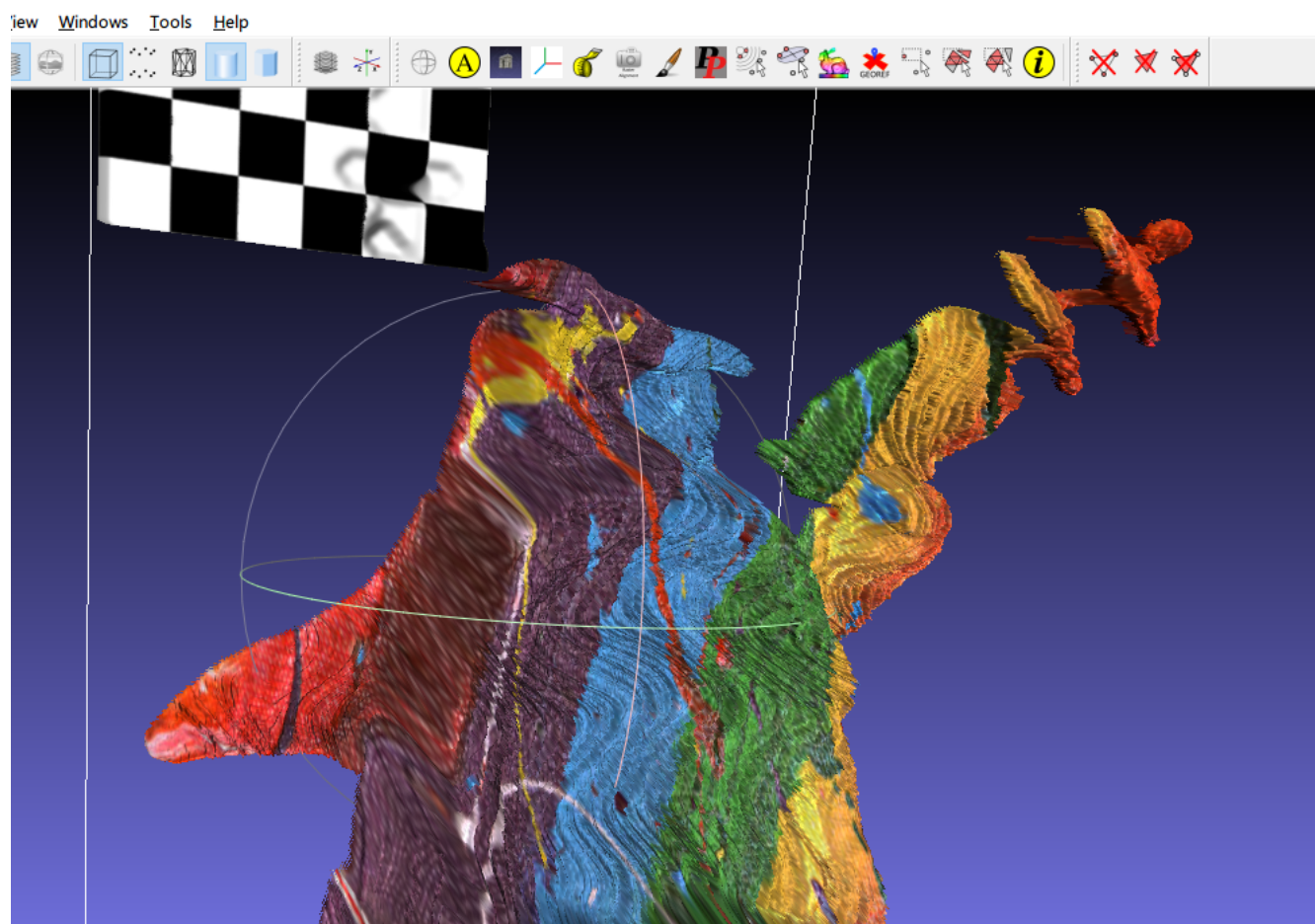
执行

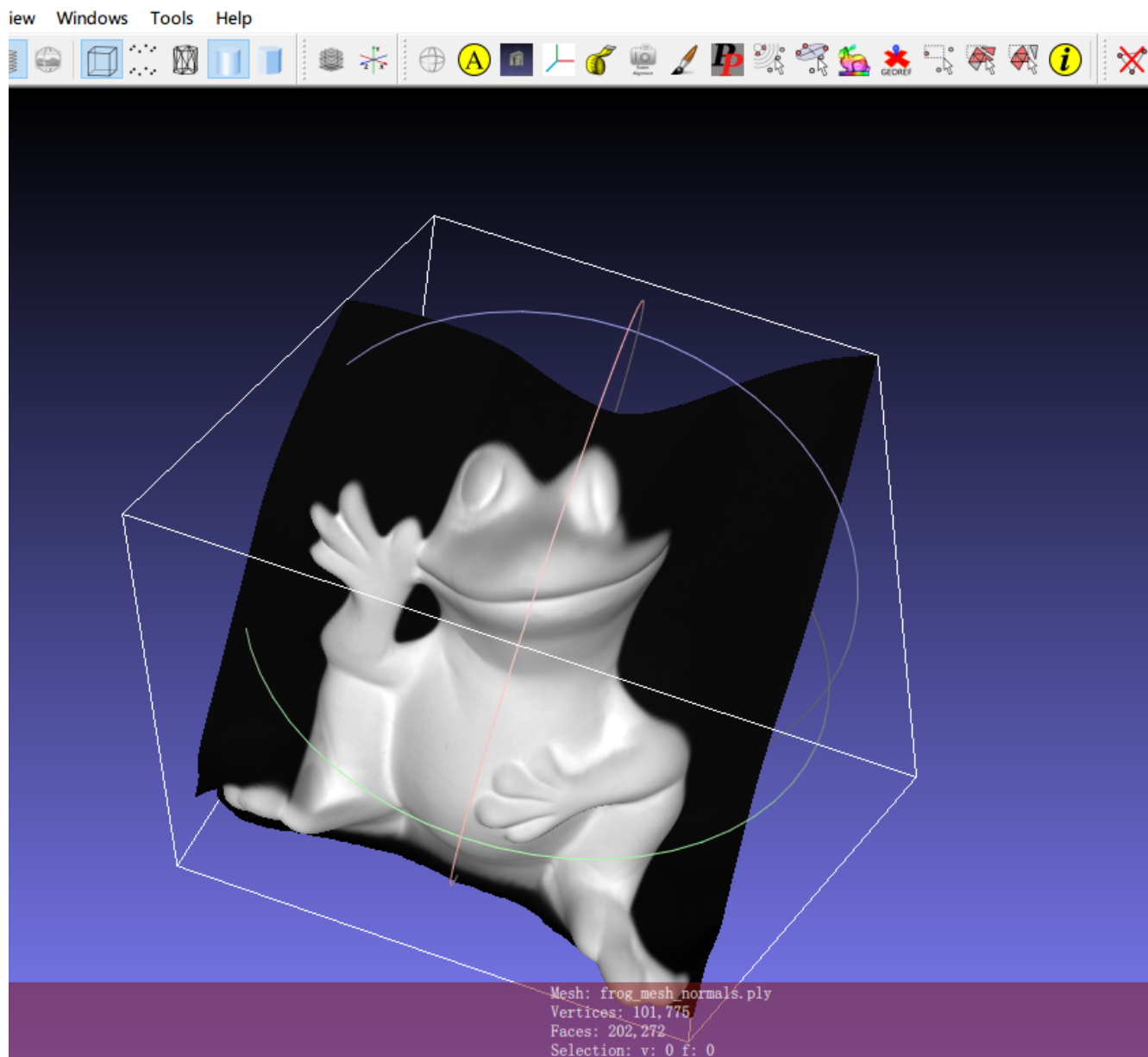
```
1 | >python combine.py tentacle both
```

运行后会产生 .ply 格式的 mesh 文件

```
(venv) E:\workspace\pyfile\CVexp\exp4_my>python combine.py tentacle both
Initialized data in 0.129999876022 seconds
Set up linear system in 19.1040000916 seconds
Solving...
Solve complete in 3.132999897 seconds
Save mesh to output/tentacle_mesh_both.ply
done :)
```

在 MeshLab 里观察产生的点云





## 三维点云中出现的问题

可见，光度测量立体视觉的重建结果，较平面扫描立体视觉重建结果更平滑，平面扫描立体视觉因为涉及到相机位置的移动，图像校准时会产生更大的误差，表面会有毛刺，不平滑。但光度测量立体视觉的结果对于彩色物体的成像结果不够好。不同颜色可能会对反照率的计算产生干扰。

另外，重建结果在边缘处质量下降严重。可能是由于背景对主体边缘产生了较大的干扰，影响结果的生成。