

多视角图像序列的三维重建 MATLAB 工具箱

目录

摘要.....	3.
1.总览	
1.1 简介.....	3.
1.2 两个简例.....	5.
2.用户手册	
2.1 相机的标定.....	10.
2.2 立体几何函数库.....	13.
2.3 相机模型函数库.....	16.
2.4 三维重建基本函数库.....	20.
3. 更多示例.....	31.
附录	
参考文献.....	34

摘要

三维重建是机器视觉的重要组成部分，也是近年来图像信息技术方面的热门课题之一。目前基于 C/C++语言的平台有 OpenCV 库(Open Source Computer Vision Library)等。MATLAB 作为一种广泛使用的优秀的科学计算编程语言，近些年来也在图像处理和机器视觉上有所发展，但其中三维重建的部分还很不全面。该工具箱主要在 MATLAB 上提供一个针对三维重建的开发平台(包括相机标定工具箱、立体几何函数库、相机模型函数库、三维重建基本函数库、可视化函数库等)，用于进行相关课题的学习、教学演示、程序开发、高级算法开发等。工具箱中还包含了利用该平台初步开发出的几种不同的三维重建解决方案(已知控制点、未知控制点、以曲线为对象的情况)，适用于不同场合的三维重建。

关键字： 三维重建；机器视觉；MATLAB；工具箱

1. 总览

1.1 简介

什么是多视角图像序列的三维重建？

多视角图像序列的三维重建是指根据一个物体的不同角度的数字图像还原出物体的三维模型，这类课题是机器视觉中重要的组成部分。三维重建技术可从平面图像中提取空间信息，补充平面图像的不足。

该工具箱解决了什么问题？能在哪些方面使用？

该工具箱主要在 MATLAB 上提供一个三维重建的开发平台（丰富的函数库、数据库、可视化等功能），用于进行相关课题的学习、教学演示、程序开发、高级算法开发等。工具箱中还包含了已利用该平台初步开发出的几种不同的三维重建解决方案，适用于不同对象的三维重建。

该工具箱作为开发平台具体有那些功能？

开发平台的功能包括

- (1) 改进的相机标定工具箱(包含 GUI 界面)——对普通的非测量图像设备进行标定，矫正图像扭曲、中心偏移等误差；
- (2) 立体几何函数库——处理空间点、矢量、射线、直线、平面、曲面等基本的几何对象之间的各种复杂关系(如改变其位置、求交点、求距离、求位置关系等)；

(3) 相机模型函数库——建立相机的投影模型，对各种坐标系（本地、相机、图像）之间的基本几何对象进行转换及处理。

(4) 三维重建基本函数库——提取特征点；根据已知坐标的控制点定位相机；管理原始数据及中间步骤产生的数据并进行存档、覆盖询问等。

(5) 可视化函数库——对前面的功能进行大量的 3D 可视化实现，几乎每个计算过程的可视化都能完美呈现。

利用这些现成的功能，用户可以方便地实现三维重建相关的程序的调试、修改、演示，也可以用于该方面课题的学习、教学等环节。

已利用该平台开发出的重建程序有哪些功能？

作者利用该平台已初步开发出两类可实际应用的三维重建程序。

(1) 当图像中含有空间坐标已知的控制点时(对于小物体的三维重建，通常用一张棋盘作为背景，在棋盘上建立空间坐标系)，可利用这些控制点先精确定位相机（每个图像对应相机的一个位置和朝向），再进一步根据图像上的对应特征点逆向求解出每个特征点的空间坐标，最后根据算出的空间点进行模型的美化。

经过实际试验，这种方法的精度非常高，仅一个 848*480 分辨率的摄像头就可以把小物体上一点确定到 0.01mm。

(2) 在 (1) 的条件下，进行了相机定位以后，如果由于物体表面的性质（例如非常光滑）只能分辨出曲线，可以在各张图上选取对应的曲线，生成 3D 曲线以确定模型的形状，再对模型进行美化。

(3) 当图像中没有空间坐标已知的控制点(例如建筑物等)时，可只利用图片上的对应特征点求出物体的形状(特征点的空间相对位置)。

该工具箱能完美实现三维重建的所有过程吗？

目前该工具箱侧重于数据处理方面的算法（二维数据还原为三维数据的过程）。至于图像识别方面的算法，如特征点识别和配对、边界和角点检测、运动检测等图像（视频）识别方面的功能，不在该工具箱的目标范围。然而近几年来 MATLAB 在这些方面已经逐步增加了新的工具箱（例如 Computer Vision Toolbox 和 Image Processing Toolbox），由于作者能力有限，未能及时将这些工具箱的内容整合到该工具箱中来，但程序中提供了数据接口和手动获取图像特征的友好界面。在本文的示例中，特征点（或曲线）由手动获取而得。

1.2 简例

下面给出两个简例，例 1 是使用开发平台的简单实例，例 2 是简单的三维重建实例。

简例 1

已标定好的 H50(极速牌)摄像头在位置坐标为 $(x, y, z)=(10, 10, 5)$ 水平俯视放置，主光轴通过原点，已知水平面上一点 P 在图像上的坐标 $(m, n)=(60, 30)$ ，求点 P 的坐标，并且把结果可视化。计算过程如下：

```
1 %simple example1
2
3 %计算
4 cam=setcam(10,10,5,'carl','H50'); %根据标定结果生成H50摄像头的
5 %参数数组以及位置(摄像头默认指向原点)
6 mn=[60,30]; %相片上的坐标用mn表示(单位是像素)
7 L=mn2Lloc(mn,cam); %'Lloc'代表L-local, '2'代表to; L是点法式直线
8 Plane=[0 0 0, 0 0 1]; %Plane是点法式平面,前三个是一点,后三个是法向量
9 P=LPlane2P(L,Plane); %生成直线与平面的交点
10 disp(['P=' P]); %在控制窗显示结果
```

可视化编程如下：

```
11
12 %可视化
13 setfigure H50 %生成空白的H50的相片(注意摄像头的光心并不一定是图心);
14 Scatter(mn); Text(mn,'相片坐标(60,30)');%在相片上标记点(m,n)
15
16 setfigure 3D %生成中心投影的空间坐标系
17 showcam(cam,3); %在坐标系中,画出相机,大小为3个单位
18 plotL3(L,20); %画出射线,长度为20单位
19 plotboard; %在水平面上画出一块标定板
20 Scatter3(P); %描出点P
21 Text(P,[' P=(' num2str(P) ') ']); %标注点P的坐标
```

说明：

以上的程序中，除了 `disp` 函数，使用的都是工具箱中的函数。在工具箱中，经常用到的数据已经过命名和规格标准化，例如 `cam` 是 6×3 的数组，前 3 行分别是相机坐标系的 3 个单位向量，第 4 行是相机位置，第 5, 6 行是中心偏移、图像大小、焦距等。

另外，函数的命名在工具箱中也遵循很好的规律。例如空间点表示为 `P`，矢量为 `v`，图像坐标为 `mn`，相机为 `cam`，本地坐标系为 `loc` 等。“2”通常用来表示转换关系。例如函数 `mn2Lloc` 可根据图像上的点生成对应的空间射线，又如 `vcam2vloc` 把相机坐标系中的矢量转换为本地坐标系中的矢量。

为了维持 MATLAB 可进行数组运算的优势，几乎所有的工具箱函数都支持数组运算。例如 $m \times n$ 坐标的数组规格为 $N \times 2$ ，直线(射线) L 为 $N \times 6$ 。

运行结果：

$P = [0.28339 \ 1.4169 \ 0]$

生成的图像如下

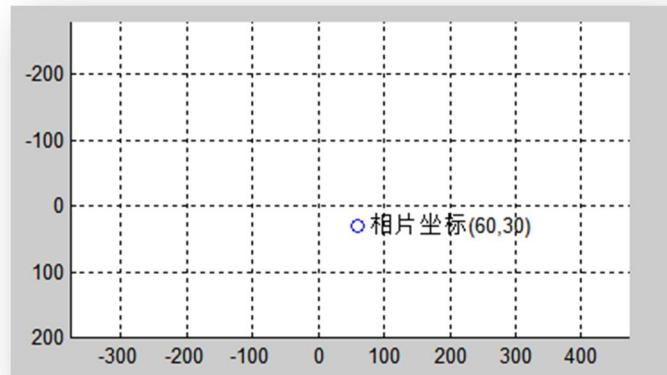


图 1

生成的 3D 视图如下

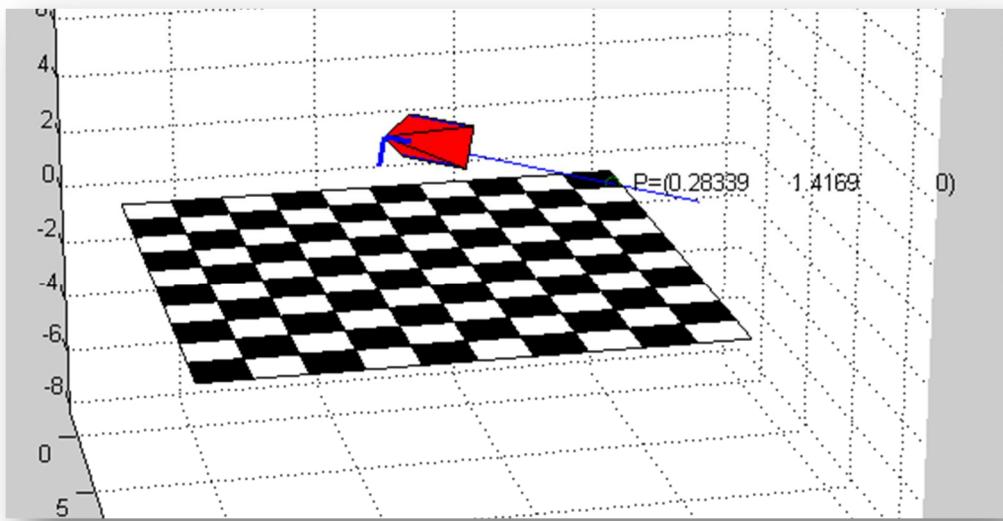


图 2

简例 2：积木的三维重建

1. 标定相机(详见相关章节)
2. 在标定板上任意摆放积木并拍照(共七张)



图 3

3. 特征点的选取 (featurepoints 函数)

该工具箱提供外部的数据接口(可利用 MATLAB 的 Computer Vision Toolbox 或者 Image Processing Toolbox 的相关功能)和方便的手动选取界面。选取完以后，程序会自动在文件夹中生成相关的数据文件供以后调用，查看等。

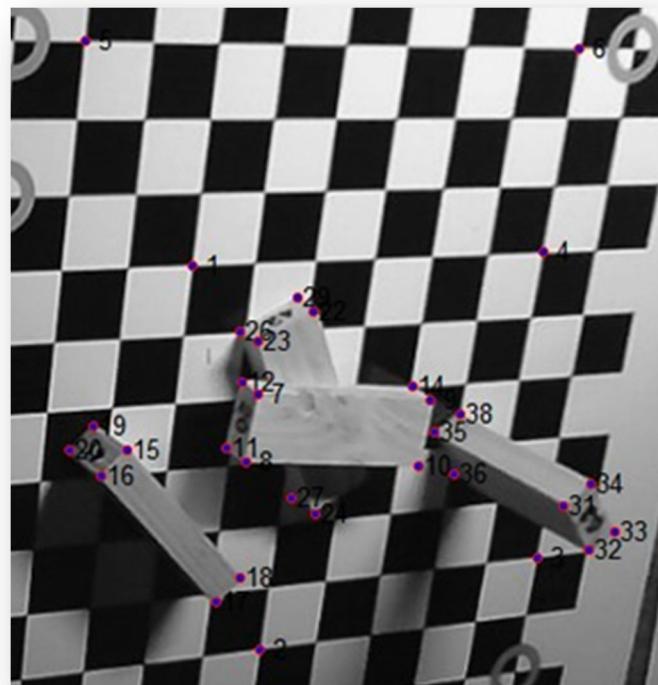


图 4. 特征点选取情况显示

4. 运行程序 next.m 和 next_check。

运行结果：

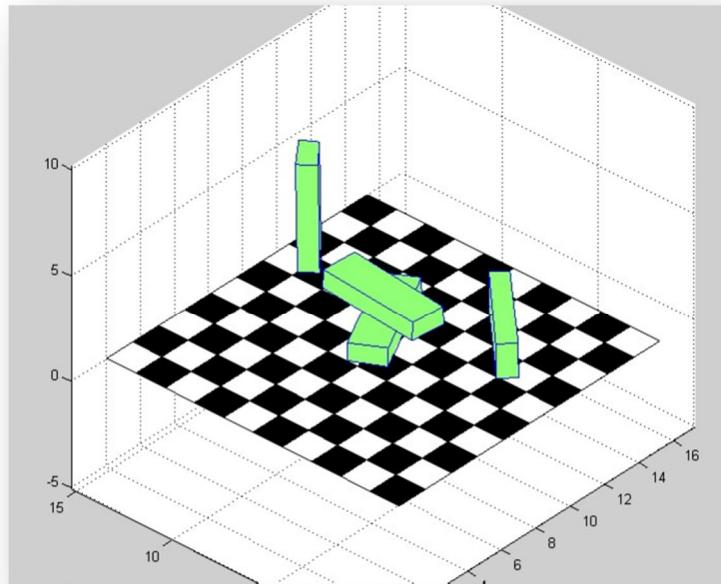


图 5

交叉检验(next_check 函数) (用于误差分析, 每个空间点放大后会显示详细数据) :

运行结果：

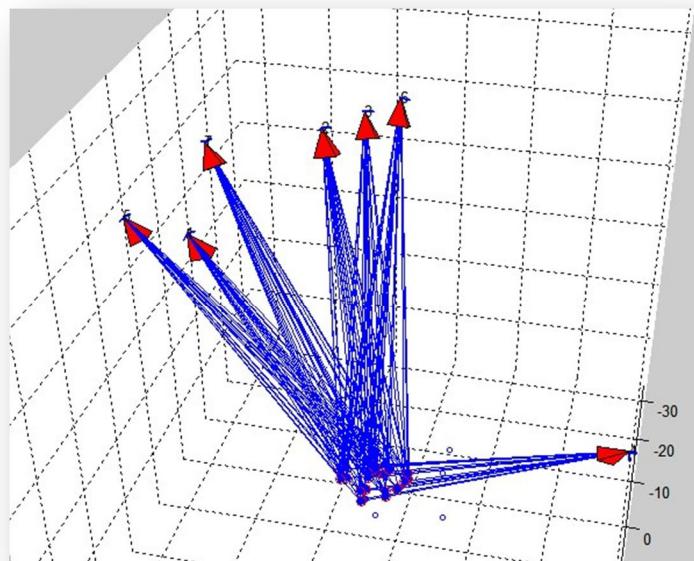


图 6

重叠投影检验(transparent_photo 函数):把生成的 3D 模型在各个对应的相机位置观测，并透过半透明图像重叠观察，以检验结果的正确性。

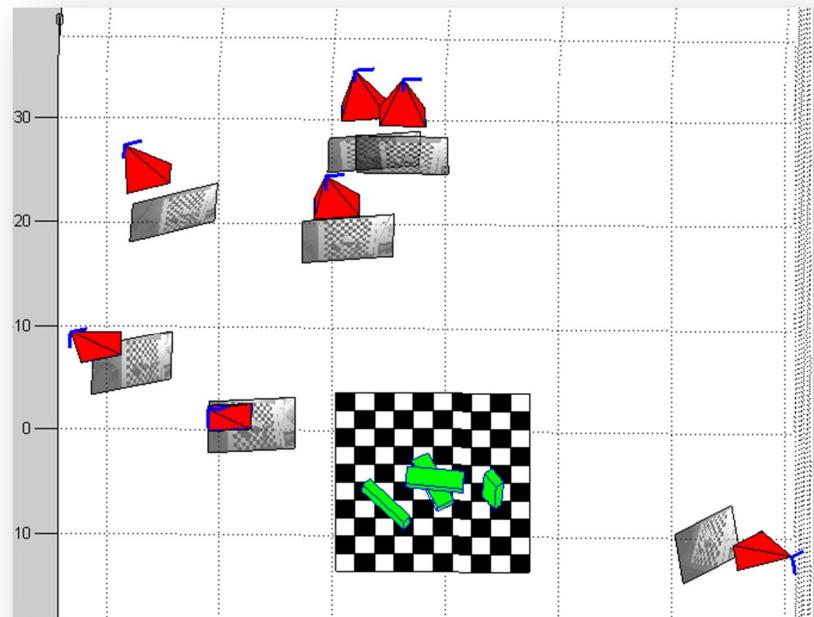


图 7

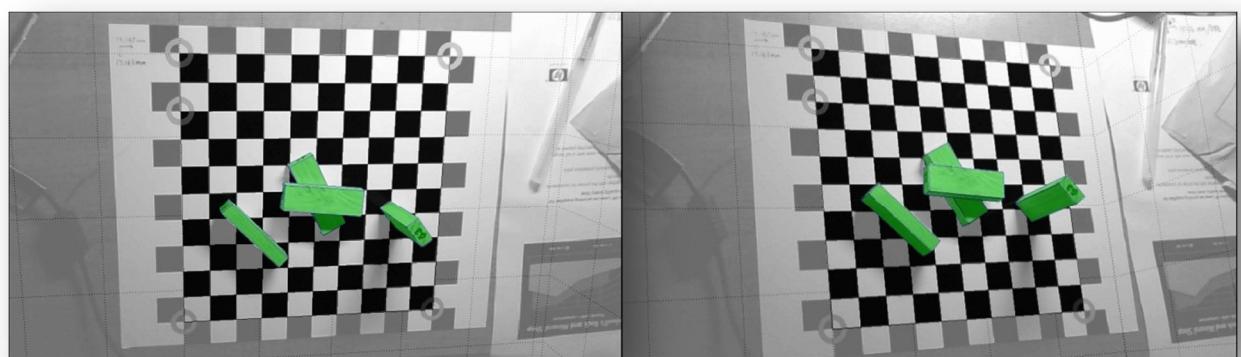


图 8

2. 用户手册

2.1 相机的标定

2.1.1 改进的相机标定工具箱

1. 改进的相机标定工具箱

相机标定工具箱是根据网上的一个开源工具箱修改整合而来，改进的部分实现了跟该作品之间很好的兼容。该工具箱可以根据一张打印的格子纸来标定相机，修正图像的形变，同时获取相机的参数(如视角，焦距等)。

原工具箱网址(<http://robots.stanford.edu/cs223b04/JeanYvesCalib/>)

2.1.2 标定步骤

1. 打印格子纸，用摄像头拍照(建议 10 张左右，如右图)

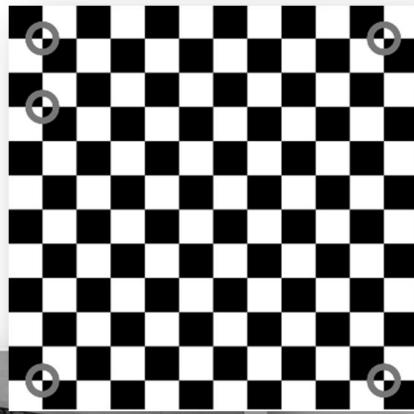


图 9

2. 对标定板拍照

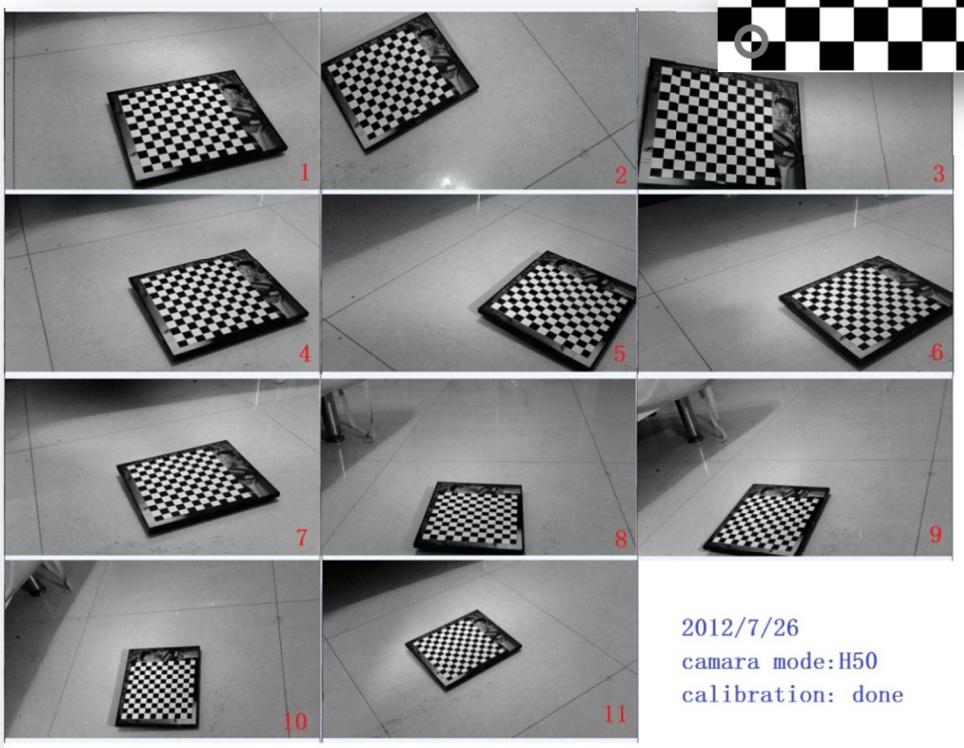


图 10

把图像按照[名字+序号]的方式命名，存入一个文件夹。例如：“calib1.bmp”，“calib2.bmp”等等(图像格式支持“.bmp”、“.jpg”、“.tif”、“.ras”、“.pgm”、“.ppm”)

3. 运行工具箱

在 MATLAB 命令窗输入 calib，启动工具箱

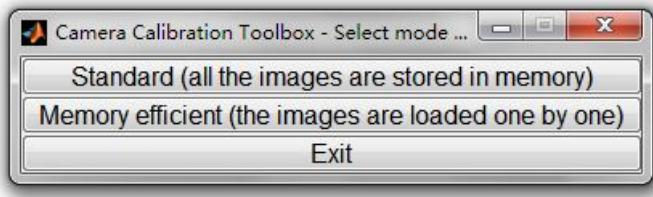


图 11

选择 Standard

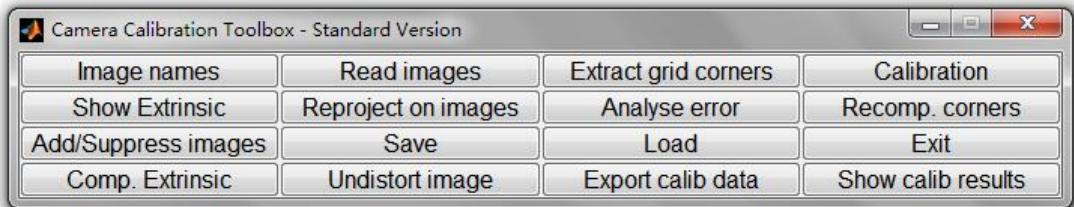


图 12

选择 Image names，按提示输入图像名，后缀名

```
Basename camera calibration images (without number nor suffix): calib
Image format: (l)=l='ras', 'b'=bmp', 't'=tif', 'p'=pgm', 'j'=jpg', 'm'=ppm') b
Loading image 1...2...3...4...5...6...7...8...9...10...11...12...13...
done
>> |
```

继续选择 Read images，结果如下

```
Loading image 1...2...3...4...5...6...7...8...9...10...11...12...13...
done
```

选择 Extract grid corners，手动或者自动在图上选取标定板的四个角(若有个别无法自动识别的将提示手动选取)，每一张图选完四个角以后，会自动选取剩下的角点。

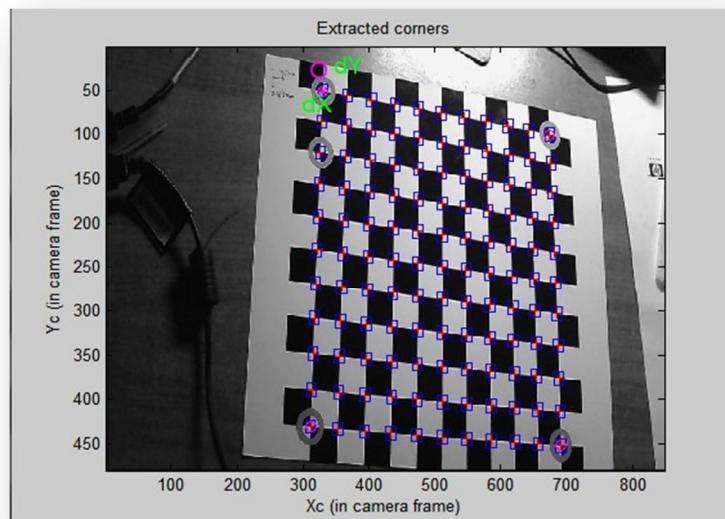


图 13

在界面中继续点击 Calibration，标定程序将自动运行。

运行结果：

分别在界面中选择 Show Extrinsic 可显示标定板与相机的位置关系

(左图为相机参考系中标定板的位置，右图为标定板参考系中相机的位置)

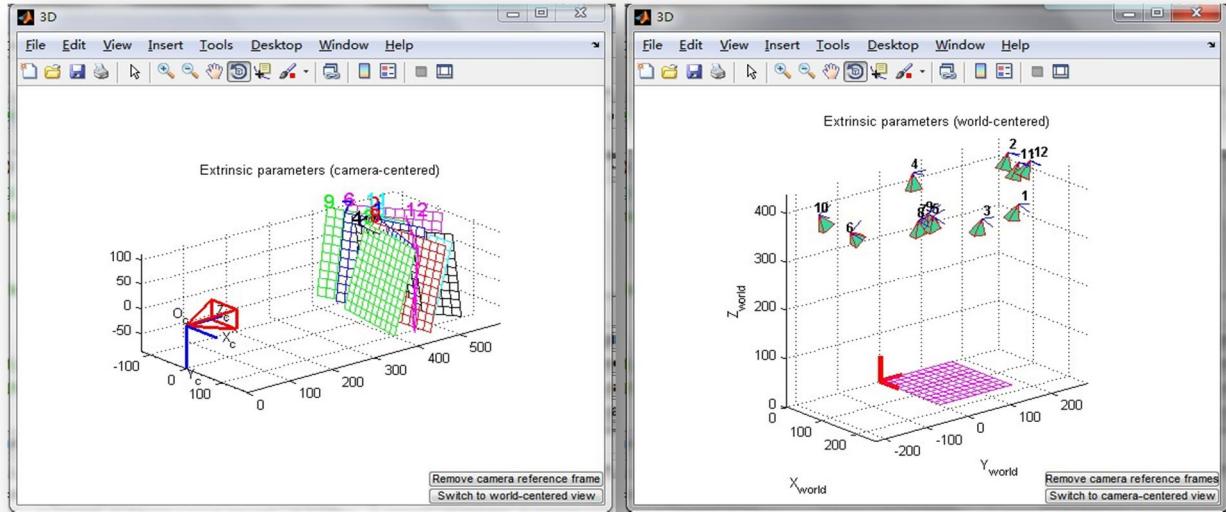


图 14

选择 Analysis Error 进行误差分析

误差结果如右图(单位:像素)

对任意一点点击左键，可在控制窗口显示该点所在的图像以及该点的坐标，以便查看该点的选取是否有误。

标定完相机以后，程序会把相机的各种参数(如焦距，视角，中心等)以及误差分布保存至 Calib_Result.mat 文件，以后每一张图像都可以根据结果矫正，生成矫正后的图像。

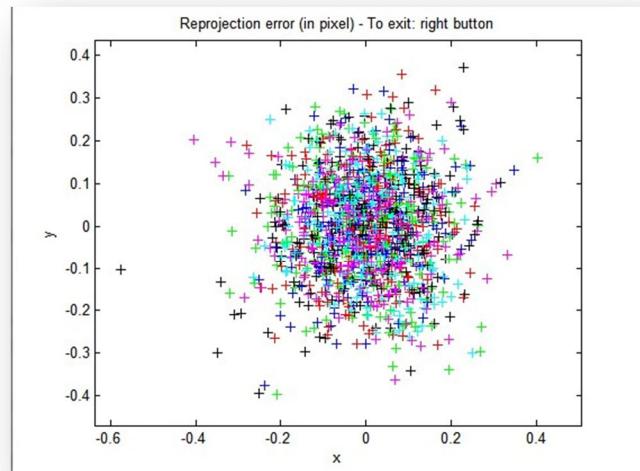


图 15

2.2 立体几何函数库

2.2.1 几何变量的格式标准

矢量 v size(v)=[N, 3]

每一行为一个矢量的三个直角坐标(x, y, z)。

空间点 P size(P)=[N, 3]

每一行为一个点的三个直角坐标(x, y, z)。

射线(或直线) L size(L)=[N, 6]

每一行的前3列为射线的端点(直线上一点), 后3列为射线的单位方向向量。

带误差的射线 Lerr2 size(Lerr2)=[N, 8]

每一行的前6列为L, 第7列为端点的误差估计(Perr), 第8列为方向向量的角误差估计(angerr)

平面 Plane size(Plane)=[N, 6]

每一行的前三列为平面上一点, 后三列为平面的单位法向量。

下面是一些常用函数的详细说明

2.2.2 关于矢量的常用函数

PlotV3

格式

PlotV3(V)

size(V)=[N, 3];

功能

在空间坐标系中画出矢量 V

vang

格式

angle=vang(V1, V2)

size(V1)=size(V2)=[N, 3]; size(angle)=[N, 1]

功能

求出两个矢量的夹角(弧度)。区间[0 pi]。支持数组运算

vlaw

格式

V=vlaw(V1, V2)
size(V1)=size(V2)=size(V)=[N, 3]
功能
求出两个矢量确定的平面的单位法向量(根据右手定则)。支持数组运算

vmag
格式
A=vmag(V)
size(V)=[N, 3]; size(A)=[N, 1];
功能
求出矢量的模长。支持数组运算

vunit
格式
Vu=vunit(V)
size(V)=size(Vu)=[N, 3]
功能
把矢量化为单位矢量

vturn
格式
(1) Vt=vturn(V, A1, A2); (1)
(2) V1=turn(V, A, theta); (2)
size(V)=size(Vt)=[N, 3]; size(A)=size(A1)=size(A2)=[1 3];
size(theta)=[1 1];
功能
(1) 把空间矢量 V 从矢量 A1 方向转到矢量 A2 方向 (绕 A1 叉乘 A2 的方向)。
(2) 把 V 以 A 为轴, 以右手定则旋转 theta 角(单位弧度)。

2. 2. 2 关于射线(直线)的常用函数

L22h
格式
h=L22h(L1, L2)
size(L1)=size(L2)=[N, 6]; h=[N, 1]
功能
计算两条直线的距离

L22P

格式

[P, h, t1, t2]=L22P(L1, L2)

size(L1)=size(L2)=[N, 6]; size(t1)=size(t2)=size(h)=[N, 1]; size(P)=[N, 6];

功能

计算两条直线的近似交点(最短的连线的中点)

Ln2P

格式

[P, Hs]=Ln2P(Lerr2)

size(Lerr2)=[N, 8]; size(P)=[N, 3]; size(Hs)=[N, 1]

功能

求出n条带误差射线的加权最优交点。根据误差决定权重，具体表达式可在文件中修改。“最优”是指选择交点P，使P到每条射线的距离(Hs)的平方和最小，用最小二乘法进行计算。

LP2H

格式

H=LP2H(L, P)

size(L)=[N, 6]; size(P)=[N, 3]; size(H)=[N, 1]

功能

给定一条射线L和一点P，求出距离H

LP2P

格式

[P1, R]=LP2P(L, P)

size(L)=[N, 6]; size(P)=size(P1)=[N, 3]; size(R)=[N, 1]

功能

给定一条射线L和一点P；求L上的一点P1使P1离P最近，并且求出P1离射线端点的距离R

P22L

格式

L=P22L(P1, P2)

size(P1)=size(P2)=[N, 3]; size(L)=[N, 6];

功能

给定两点P1, P2，求出以P1为端点，过P2的射线L

plotL3

格式

plotL3(L, Len, varargin)

size(L)=[N, 6]; size(Len)=[1, 1]或[N, 1];

varargin是([属性名], [属性值], ...)的参数对，用于控制线的属性，用法与MATLAB的plot函数一致

功能

画出指定长度(Len)的射线L

LPlane2P

格式

P=LPlane2P(L, Plane) ;

size(L)=size(Plane)=[N, 6]; size(P)=[N, 3];

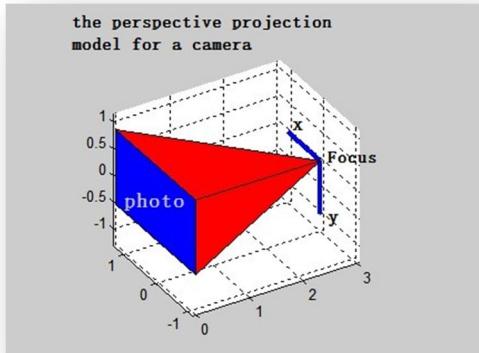
功能

求出射线L和平面Plane的交点P。

2.3 相机模型函数库

2.3.1 相机模型

相机模型如下方左图所示，蓝色的平面是图像位置，四棱锥的顶点是相机焦点，空间中的一点与焦点的连线经过图像上的对应点(右图)，这种投影方式成为“中心投影”。



工具箱中的相机模型

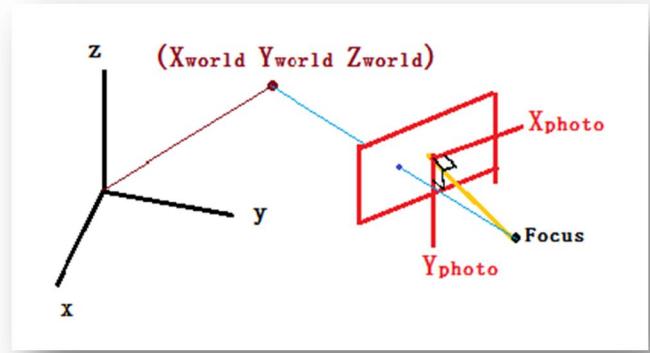


图 16

相机成像的数学模型（中心投影）

2.3.2 程序中用到的坐标系

本地坐标系:与物体保持相对静止的坐标系（在其他文献中也称世界坐标系）

相机坐标系:与相机保持静止，以主光轴为z轴，相机右方为x轴，相机下方为y轴的坐标系(如图)。

图像坐标系:光轴上的一点在图像上的投影点作为原点(而不是图像的中心)，图像右方为x轴，图像下方为y轴的平面坐标系。图像中的坐标在程序中以'mn'表示 (size(mn)=[N, 2])。原点的位置在标定相机时自动算出，储存在 Calib_Results.mat 的 cc 变量中。

spatial 坐标系:由于图像在 MATLAB 中通常以矩阵的形式保存，所以还有一种与矩阵行标和列标一致的坐标系(但允许使用小数)，在程序中通常用'xy'表示，示意图如下(图中的每个方格代表

一个像素)。

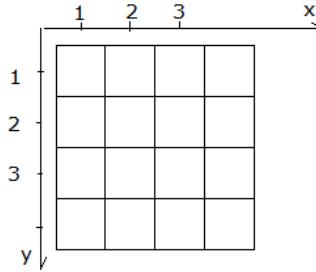


图 17

2.3.3 程序中对特定数组的格式规定

相机基本数组 cam size(cam)=[N, 6];

cam(1:3, 1:3) 中的每一行分别是相机坐标系的三个正交单位矢量在本地坐标系中的坐标。

cam(4, :) 是相机在本地坐标系中为位置

cam(5, :) 暂时闲置

cam(6, 1) 是拍摄图片的宽度(单位:像素)

cam(6, 2) 是拍摄图片的高度(单位:像素)

cam(6, 3) 是相机的焦距

图片坐标数组 mn size(mn)=[N, 2];

mn 数组的每一行都是图片坐标系中的一个点

spatial 坐标数组 xy size(xy)=[N, 2];

xy 数组的每一行都是spatial 坐标系中的一个点

矢量数组 V size(V)=[N, 3];

V 数组中每一行都是一个矢量, 坐标系视情况而定, 相机坐标系中的 V 也命名为 vcam, 本地坐标系中的 V 也命名为 vloc

2.3.4 常用函数详细说明

setcam

格式

(1) cam=setcam(camera)

(2) cam=setcam(x, y, z, 'cart', camera)

(3) cam=setcam(phi, theta, r, sph_cart, camera)

功能

生成参数组, 用相机标定的结果 Calib_Results.mat 设定 cam 数组中的相机参数, 设置相机的 z 轴 (光轴) 经过原点, 根据不同的格式设定相机的位置。

(1) 用默认参数设定相机位置 (默认值可在程序中自定义)

- (2) 用直角坐标指定相机的位置为 [x y z]
(3) 用球坐标指定相机的位置为 [phi, theta, r]

setcamP2P

格式

cam=setcamP2P (Pcam, targ, camera)

size(cam)=[6, 3]; size(Pcam)=[1 3]; size(targ)=[1 3]; size(camera)=[1 N]

class(camera)=char;

功能

通过指定相机位置Pcam和相机目标targ(落在光轴上, 投影到图片上mn=[0, 0])来确定cam, 其他参数从Calib_Results.mat中提取。

axes2cam

格式

(1) cam=axes2cam

(2) cam=axes2cam(ha)

size(ha)=[1, 1]; size(cam)=[6, 3];

功能

当前的图像窗口中(axes对象)的属性projection(投影方式)的值为"perspective"(中心投影)时, 可根据axes对象的相关属性生成cam数组。相机其他参数从Calib_Results.mat提取。

cam2axes

(类比) axes2cam

mn2spatial

格式

(1) xy=mn2spatial(mn)

size(mn)=size(xy)=[N, 2];

(2) [x, y]=mn2spatial(m, n)

size(m)=size(x)=size(n)=size(y)

功能

图坐标系的坐标转换为spatial坐标系的坐标

spatial2mn

(类比mn2spatial)

vcam2vloc

格式

vloc=vcam2vloc(vcam, cam)

size(vloc)=[N, 3]; size(vcam)=[N, 3]; size(cam)=[6, 3];

功能

把相机坐标系中的矢量转换成本地坐标系中的矢量。

vloc2vcam

(类比 vcam2vloc)

mn2vcam

格式

$\text{vcam} = \text{mn2vcam}(\text{mn}, \text{cam})$

$\text{size}(\text{vcam}) = [N, 3]; \text{size}(\text{cam}) = [6, 3];$

功能

找到图片坐标系中的点在相机坐标系中对应的矢量

mn2vloc

(类比 mn2vcam)

mn2Lloc

格式

$\text{Lloc} = \text{mn2Lloc}(\text{mn}, \text{cam})$

$\text{size}(\text{mn}) = [N, 2]; \text{size}(\text{cam}) = [6, 3]; \text{size}(\text{Lloc}) = [N, 6];$

功能

找出图片上一点 mn 对应的本地坐标系的射线

camturn

格式

$\text{V1} = \text{camturn}(\text{V}, \text{m}, \text{n}, \text{theta}, \text{cam})$

$\text{size}(\text{x}) = \text{size}(\text{y}) = \text{size}(\text{theta}) = [1, 1]; \text{size}(\text{cam}) = [6, 3]; \text{size}(\text{V1}) = [6, 3];$

功能

旋转相机，使相机的 z 轴(光轴)由原来的 $\text{mn2vloc}([0 \ 0], \text{cam})$ 方向摆到 $\text{mn2vloc}([\text{m} \ \text{n}], \text{cam})$ 的方向，V 也随相机一起摆到 V1 方向。如果不输入 cam，则默认相机坐标系与本地坐标系重合

aCamTurn

格式

$\text{V} = \text{camturn}(\text{V1}, \text{m}, \text{n}, \text{theta}, \text{cam});$

功能

参考 camturn 中的参数，在 m, n, theta, cam 参数相同时，可把 V1 逆向摆动到 V。

Ploc2mn

格式

$\text{mn} = \text{Ploc2mn}(\text{Ploc}, \text{cam})$

功能

本地坐标系中的一点 P 投影到图像上的坐标

showcam

格式

```

showcam(cam, f)
size(cam)=[6, 3]; size(f)=[1, 1]

```

功能

在当前的图形窗中画出参数数组为 cam 的相机模型(包括位置, 朝向, 视野范围等信息), f 为相机模型的焦距在图中的长度, 用于控制模型的大小。

2.4 三维重建基本函数库

该部分主要实现三维重建中一些基本的问题, 如相机定位的几种方式, 相机朝向的最优化等

2.4.1 三棱锥相机定位法(原创)

三棱锥相机定位法可根据图片上空间坐标已知的极少控制点(大于 3)实现相机的定位。

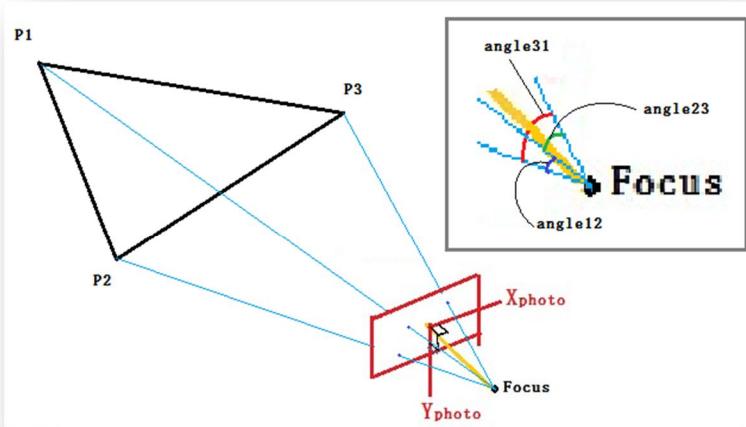


图 18-1
数值解三棱
锥原理

所谓“三棱锥”是指在本地坐标系中由相机坐标 P 和和三个已知的空间 P1、P2、P3 构成的三棱锥 $P-P_1P_2P_3$ 。若三点都出现在同一张图像中(坐标为三行的 mn 数组), 则可根据相机模型求出三条棱两两之间的夹角 (ang12、ang23 和 ang31)。这些已知数据足以确定点 P 的解。但事实上点 P 的解一般有 n 个 ($n \leq 4$) 所以需要另一个三棱锥来筛选出相机的位置, 所以“三棱锥定位法”至少需要四个点。

程序中可分别用牛顿法迭代法和遍历法解三棱锥, 前者速度较快, 但有可能解不出, 每一个解三棱锥的步骤都配有相应的可视化函数(如下图)。

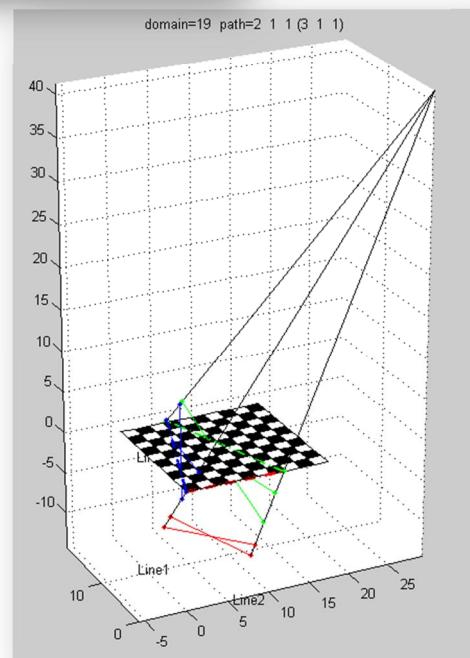


图 18-2
数
值
解
三
棱
锥
过
程

2.4.2 相机位置及朝向的统计最优化

区别于三棱锥定位法，当图片上有很多空间坐标已知的控制点（图像中的棋盘格点），可先选取其中四个用三棱锥法初步定位，再把所有点输入最优化函数，精确定位相机，使图像上的点方差最小。

具体算法（cam_modi 函数）：

设图像中的 N 点 mn ($size(mn)=[N, 2]$) 所对应的空间点为 $Ploc$ ($size(Ploc)=[N, 3]$)。程序先用三棱锥法求出 cam 数组（以确定相机位置初值），利用函数 $mn1=Ploc2mn(Ploc, cam)$ ；把空间点投影到图片坐标系中，与 mn 进行比较，算出所有点的偏移方差，再用 MATLAB 自带的局部最优化函数 fminsearch 把相机位置 $P=cam(4, :)$ 和朝向 $cam(1:3, :)$ 调整到使方差最小的值。

统计最优化模块同样配有可视化函数用于检验和调试

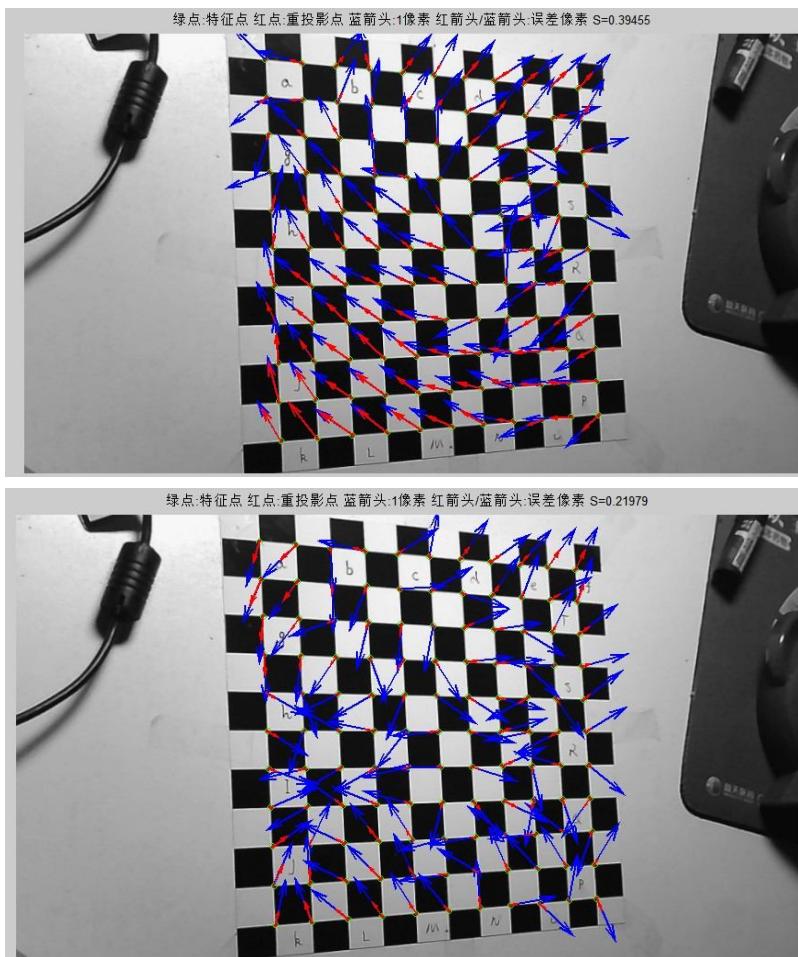


图 19

矫正前投影标准偏差为 0.39 像素，矫正后为 0.21 像素（图中红色箭头是该点的偏差，蓝色箭头代表一个像素的偏差，作为对比）。

2.4.3 空间点的确定

若已知若干张图像上一点（图中为三点）和每张图像所对应的相机位置，就可以用每张图像逆向投影出一条射线（函数 $mn2Lloc$ ），再求其最优近似交点。每个相机在进行位置和朝向最优化的时候会估计位置误差（Perr）和朝向误差（angerr），生成带误差的空间直线（Lerr2 数组），立体

几何工具箱中的 Ln2P 函数可根据这些误差决定每条射线的权重，再用最小二乘法求出最优交点，并估计交点的误差。

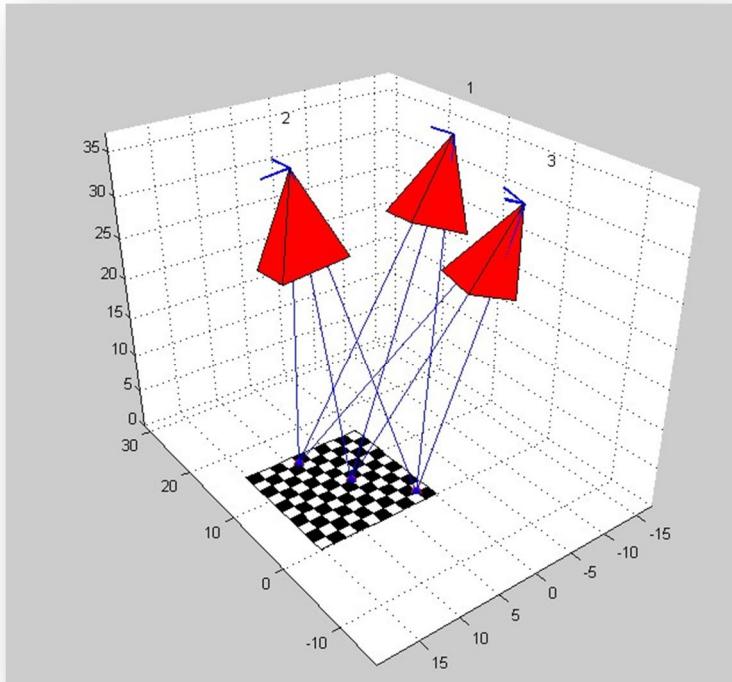


图 20

2.4.4 空间曲线的确定

当被拍摄的物体表面非常光滑时，无法识别出不同图片中对应的特征点（无论是肉眼还是自动识别程序）。此时 2.4.2 中确定空间点的方法失效。所以程序中开发了一种新的方法，在只能识别不同图像中物体上对应的特征曲线时，还原出该曲线的空间位置。

相应的可视化程序效果如右图所示。在两张图片中选取对应特征曲线上的若干点（ $mn1$ 和 $mn2$ 数组），点的数量和位置不必一一对应，落在同一条特征曲线上即可。利用 $mn2Lloc$ 函数可将 $mn1$ 和 $mn2$ 还原成空间的两张曲面，进而算出两曲面的交线，即所求的空间曲线。

关于特征曲线的还原实例，参见第三章 (3. 更多示例)。

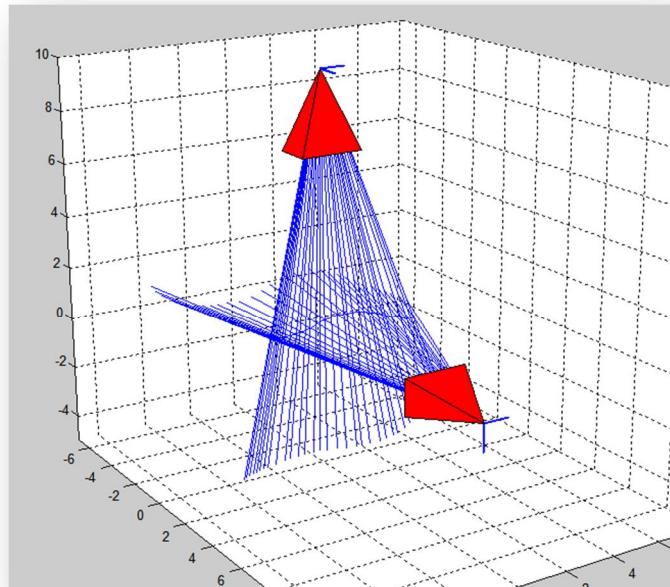


图 21

2.4.5 没有已知控制点的情况——deturn 算法

如果不同图像中可以找到对应的特征点，却不知道任何特征点的空间坐标，可以通过该模块求解控制点的坐标。这是最高级的一种相机定位方法，但误差比上面的定位方法大（由于已知条件过少）。

对于这种情况下的三维重建，作者先自己发明了一种“去转动法”(deturn)，与随后在资料检索中发现的“核面几何”法（普遍使用的方法）原理相同，只是后者的操作对象是图片，对其进行平面的变换，前者的处理对象是空间矢量，对其进行平移与转动。由于前者较为直观，可画3D矢量图表示，下面介绍 deturn 法。

在两次拍照中，相机 1(cam1) 的位置为 Pcam1，相机 2(cam2) 的位置为 Pcam2，连接 Pcam1 与 Pcam2 之间的线段叫做 cam1 与 cam2 的基线。设两台相机同时拍摄到空间中的两点 P1 和 P2，两点投影到图像上的过程如下图所示。显然，平面 Plane1(过 P1, Pcam1, Pcam2) 与平面 Plane2(过 P2, Pcam1, Pcam2) 相交于基线。事实上，所有这样的平面都相交于基线，这样的性质在本程序中称为平行判据：

如果 cam2 只由 cam1 平移而来（而不发生转动或摆动），可把 L11, L21, L12, L22 的方向矢量(vcam 数组)都放在同一相机坐标系中(cam1 的相机坐标系)表示(矢量坐标不发生变化)，求出各平面，它们也会交于基线，满足平行判据。如果 cam2 除了平移，还加入了转动和摆动，当它们把各自的矢量放在同一坐标系中表示时，求出的平面不会交于同一条线，不满足平行判据。所以，根据平行判据可以断定两台相机拍照时是否朝向一致，并且求出基线的方向。

现实中大部分情况下，cam2 相对于 cam1 的三种运动都会发生。于是可以先求 cam2 的逆转动（和摆动），使所有的平面满足平行判据，这样，便可消去转动，使两个相机朝向相同，进而顺利求出基线，还原出 cam1 和 cam2 的相对位置。再用 2.4.3 中的方法求出 P1, P2 等点的空间坐标。

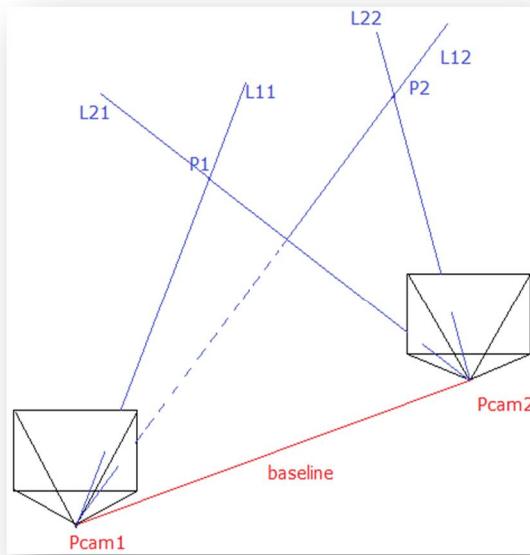


图 22

利用该模块中的 deturntrial 函数（自变量为转动参数 theta 和摆动的参数 [x, y]，详见立体相机模型函数库中的 acamturn 函数），将平行判据符合的程度用一个数 E ($E > 0$) 来表示，E 越小，说明平行判据满足得越好。实际实验中，theta=0 时，E 关于 x, y 的典型图像如下面左图所示。由于函数非常特殊，MATLAB 的自带 fminsearch 函数无法求出其最小值（处于以狭长的“峡

谷”中)。经过改进 E 的算法, 该函数变为右图, 解决了这一问题。

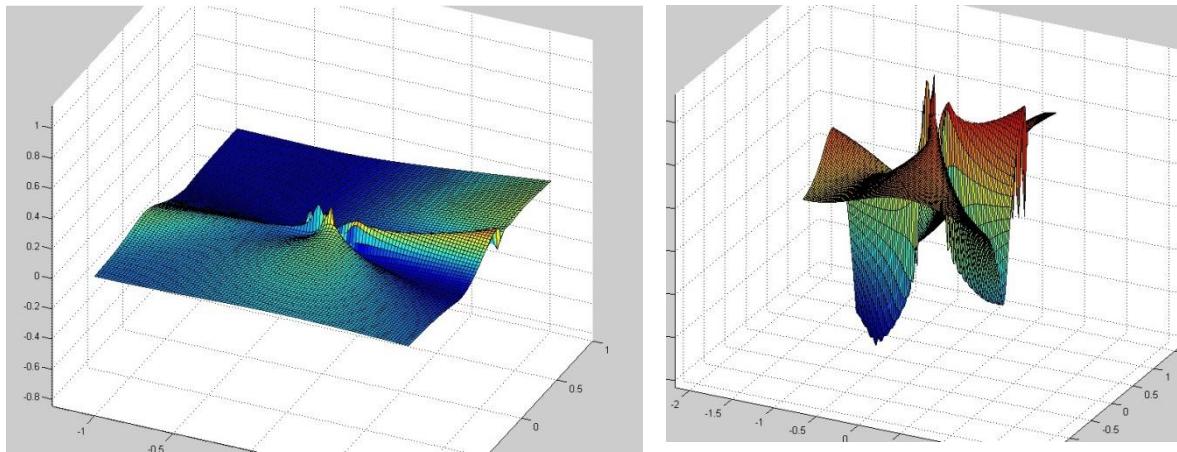


图 23

程序中对特定数组的格式规定

三棱锥 (pyramid) pmd size(pmd)=[4, 3]

pmd(1, 1:3) 分别是底面的三点 (P1, P2, P3)

pmd(4, :) 分别是对应的三个顶角 (ang1, ang2, ang3)

空间坐标与图像坐标的合并数组 Pmn size(Pmn)=[N, 5]

pmd(:, 1:3) 是空间点数组 P

pmd(:, 4:5) 是空间数组每个 P 对应的图像坐标 mn

程序中对存档文件的格式规定

主要的存档文件有 Lerr2.mat 以及 begin_cams.mat

begin_cams.mat 主要用于存档原始数据和相机数据

其中主要变量如下

mni (特征点及其编号)

class(mni)=cell; size(mni)=[N, 1]; size(mni{n})=[N, 3];

mni{n} 是第 n 张图像中的特征点原始数据 (如果没有特征点, 则为空)

mni{n}(:, 1:2) 是特征点的 mn 坐标, mni{n}(:, 3) 是每个对应特征点的序号。不同图片中的对应特征点有同样的序号。

mnc (特征曲线及其编号)

class(mnc)=cell; size(mnc)=[N, 1];

class(mnc{n})=cell; size(mnc{i})=[N, 2];
mnc{n} 是第 n 张图像中的特征曲线原始数据(如果没有, 则为空)
mnc{n}{m, 1} 为第 n 张图像中的一条特征曲线数据, size=[N, 3]

cams

每张图像对应的相机 cam 数组数据(由相机定位生成)
class(cams)=cell; size(cams)=cell;
class{n} 是第 n 张图像对应的 cam 数组
size(class{n})=[4, 3];

Perrs

每个 cam 数组对应的位置不确定度
size(Perrs)=[N, 1]

angerrs

每个 cam 数组对应的角度不确定度(弧度制)
size(angerrs)=[N, 1]

Lerr2.mat 主要储存计算中间数据及结果数据

包含的主要变量如下

DL

特征点的空间坐标
class(DL)=double; size(DL)=[N, 3]
DL{n} 是编号为 n 的特征点的本地坐标。

Lerr2

带误差的空间射线
class(Lerr2)=cell; size(Lerr2)=[N, 1]
Lerr2{n} 是编号为 n 的特征点对应的所有带误差的空间射线(来自所有包含该特征点的相机)

Curve

特征曲线的空间坐标
class(Curve)=cell; size(Curve)=[N, 1]
Curve{n} 是编号为 n 的特征曲线的空间坐标数组。
class(Curve{n})=double; size(Curve{n})=[N, 3];

Lcurve

图像中的特征曲线对应的空间曲面(由 N 条射线组成)
class(Lcurve)=cell; size(Lcurve)=[N, 1]
Lcurve{n} 是图像中编号为 n 的特征曲线对应的空间曲面
class(Lcurve{n})=double; size(Lcurve{n})=[N, 6];

2.4.5 常用函数表

sympmd

格式

P=sympmd (pmd0)

size (pmd0)=[4, 3];

功能

用 matlab 的符号计算工具箱解出三棱锥顶点的所有解。这个函数求出的结果是最权威最精确的，但速度也是最慢的，通常在调试三棱锥的数值解法是使用。

numpmd

格式

P=numpmd (pmd0)

size (pmd0)=[4, 3]; size (P)=[N, 3];

功能

遍历法求解三棱锥顶点(所有解)，只在极偶然的情况下会有漏解

ntpmd

格式

[Px, Py, Pz]=ntpmd (pmd0, P)

size (pmd0)=[4, 3]; size (P)=[1, 3];

功能

牛顿迭代法解三棱锥。虽然速度很快，但由于需要猜初值 P，且只能返回一个解，其他程序中基本已经用 numpmd 将其取代。

ori

格式

cam1=ori (Pmn2, cam)

size (Pmn2)=[2, 5]; size (cam)=size (cam1)=[6, 3]

功能

若已知相机位置，但是不知道相机朝向，可用两个空间坐标已知的特征点定位相机的朝向。

cam_modi

格式

[cam1, Perr, angerr, S2]=cam_modi (Pmn, cam)

size (Pmn)=[N, 5]; size (cam)=size (cam1)=[6, 3]; size (Perr)=size (angerr)=[N, 1];

size (S2)=[1, 1];

功能

相机位置及朝向最优化 cam 为三棱锥法解出的相机参数数组，cam1 为优化后的参数数组。S2 是

上文所述的重投影方差。

Ls22curve

格式

```
curve=ntpmd(Ls1,Ls2)  
size(Ls1)=[N1, 6]; size(Ls2)=[N2, 6]  
size(curve)=[N3, 3];
```

功能

Ls 的格式与直线 L 数组相同，但 Ls1 能张成一张放射状的曲面。curve 的格式与空间点 P 数组格式相同，但能排成一条空间曲线。

deturntrial

格式

```
E=deturntrial(V1,V2,x,y,theta);
```

功能

利用 x, y, theta 三个参数对向量 V2 进行摆动和旋转，然后估计 V2 与 V1 中对应矢量张成平面是否交于同一条直线，E 越好说明越精确地相交与同一条直线。

deturn

格式

```
[x,y,theta]=deturn(V1,V2,[x0,y0,theta0])
```

功能

根据 deturntrial 函数的返回值，求出 [x, y, theta]，使返回的 E 值最小。可以输入初值 [x0, y0, theta0]，若不输入，初值默认为 [0, 0, 0]。

featurepoints

格式

```
featurepoints(prenname,N)  
size(prenname)=[1,N1]; size(N)=[1,N2];  
class(prenname)=char
```

功能

手动取特征点的友好界面程序， prename 是同一组图像的文件名(除去后面的编号)， N 是要操作的图像序号数组。运行后根据控制窗的提示选择功能，可进行添加、删除、修改等操作。

featurelines

格式

```
featurelines(prenname,N)
```

功能

手动取特征曲线的友好界面程序， 类比 featurepoints。

begin格式

begin (N)

size(N)=[1, N]

功能

处理含有标定板的图片，生成每张图片对应的相机 cam 数组。运行前先用相机标定工具箱，获取每块标定板的图像坐标，自动生成**_corners.mat。begin 函数根据这些角点数据生成 cam 数组，以及相应的误差估计(perrs anerrs) 储存到 begin_cams.mat 文件的 cams、perrs、angerrs 变量中。程序的最后再调用 renewLerr 函数和 renewDLHs 函数更新 Lerr.mat 数据文件。

begin_check格式

begin_check(N)

功能

可视化检验 begin 函数运行的结果。

renewLerr2格式

renewLerr;

功能

被 begin 和 next 函数调用的子函数，用于更新 Lerr2.mat 文件中的 Lerr2 变量。

renewDLHs格式

renewDLHs;

功能

被 begin 和 next 函数调用的子函数，用于更新 Lerr.mat 文件中的 DL 和 Hs (DL 的误差估计) 变量。

renewLcurve格式

renewLcurve;

功能

更新 Lerr2.mat 中的 Lcurve 变量，重复则覆盖。调用 mn2Lloc 函数对 begin_cams.mat 中的特征点数据进行运算。

renewCurve格式

renewCurve;

功能

更新 Lerr2.mat 中的 Curve 变量，重复则覆盖。调用 Ls22curve 函数对该文件中的 Lerr2 变量进行运算。

next

格式

next(N)

size(N)=[1, N]

功能

处理 N 数组中指定序号的图像（N 是要操作的图像的序号的数组）。next 函数根据每张图像上所有中空间坐标已知的特征点，更新对应的 cam 数组（覆盖在 begin_cams.mat 文件的 cams 数组）。该函数起到传递作用，如果只有部分图像中有已知控制点的原始数据，可以先用 next 处理这些图像，求出更多的控制点，进而可以对其他包含了新求出的特征点的图片进行处理，定位相机。如此一直传递下去（各点的误差数据也会传递，且传递次数越多，误差增大）。

next_check

格式

next_check(N); size(N)=[1, N];

功能

可视化检验指序号的图像的 next 函数运行的结果（N 为序号数组）。

Curve_check

格式

Curve_check;

功能

可视化检验所有已生成的空间曲线，并在本地坐标系中标出。

reset3D

格式

reset3D;

功能

对存档文件 begin_cams.mat 和 Lerr2.mat 进行清零初始化，若没有存档文件，建立存档文件。

im4save

格式

im4save(filename)

功能

用于储存 Image Aquisition Toolbox 获取的图像。运行前先用 MATLAB 自带的图像获取工具箱（Image Aquisition Toolbox），连接摄像头，根据提示获取图像（操作详情参考 MATLAB “帮助”）

的相关章节）。获取图像后，储存为[filename].mat文件，运行该函数。函数读取mat文件，生成图像储存在MATLAB的当前路径中。图片的命名遵循[filename]+[序号]的格式，便于calibration toolbox读取，默认储存为bmp格式（可在函数文件中修改）。

transparent_photo

格式

```
h=transparent_photo(cam, photoname, Size, t)  
size(cam)=[6, 3]; class(photoname)=char; photoname=[1, n];  
size(Size)=size(t)=[1, 1];
```

功能

3D 模型与图像的“重投影检验”（见 1.2 中的简例 2）。

Size 是图像的宽度，t 是定时关闭的时间参数（可选择不输入），h 是透明图像的句柄（方便以后删除）。

imshow3

格式

```
imshow3(filename)  
class(filename)=char; size(filename)
```

功能

在空间坐标系中显示图像，可以用鼠标把图像在空间中旋转、平移、放大、缩小等。

3. 更多示例

3.1 建立线条模型(算法原理见 2.4.4)

该示例以四驱车模型的外壳为对象，建立其线条模型（美化渲染等过程略去）

(1) 用标定过的相机对车壳进行拍照(以标定板作为背景)



图 24

(2) 利用 featureline 函数根据提示获取特征曲线。

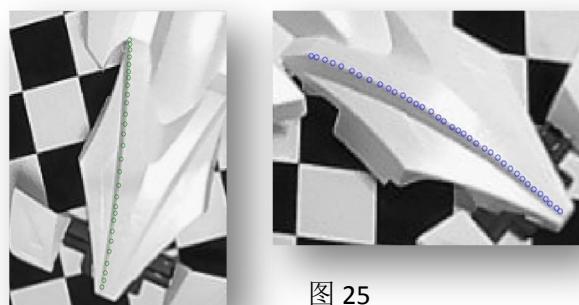


图 25

(3) 分别运行 renewLcurve 函数(生成对应的射线数组)和 renewCurve(生成对应的空间曲线)函数得到最后的结果。

用 Curve_check 函数显示曲线(只选取了部分主要特征曲线)

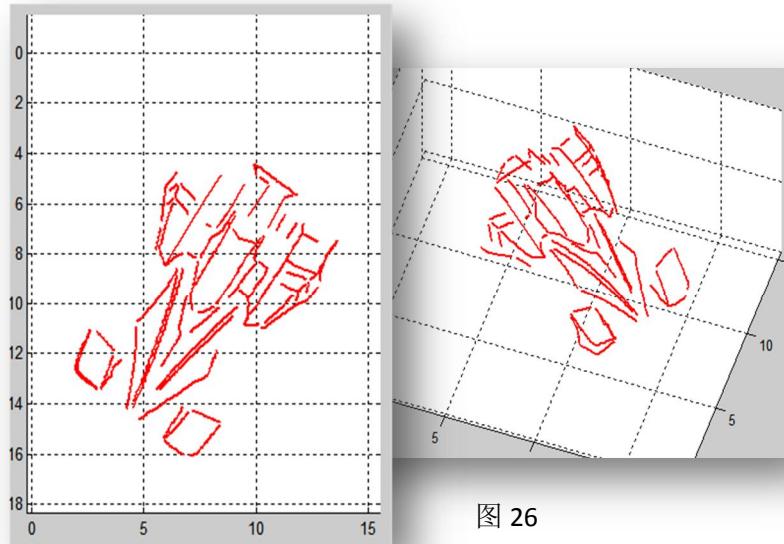


图 26

3.2 没有控制点情况下的模拟实验(2.4.5)

在该实验中，先设定好相机的位置（用 `setcam` 函数的球坐标模式），再用 `sinc` 函数生成的一张空间曲面网格。用 `Ploc2mn` 函数求出在两个相机中进行投影，用 `setfigure` 显示空白图像，再用 `plotnet` 函数画出模拟图像。

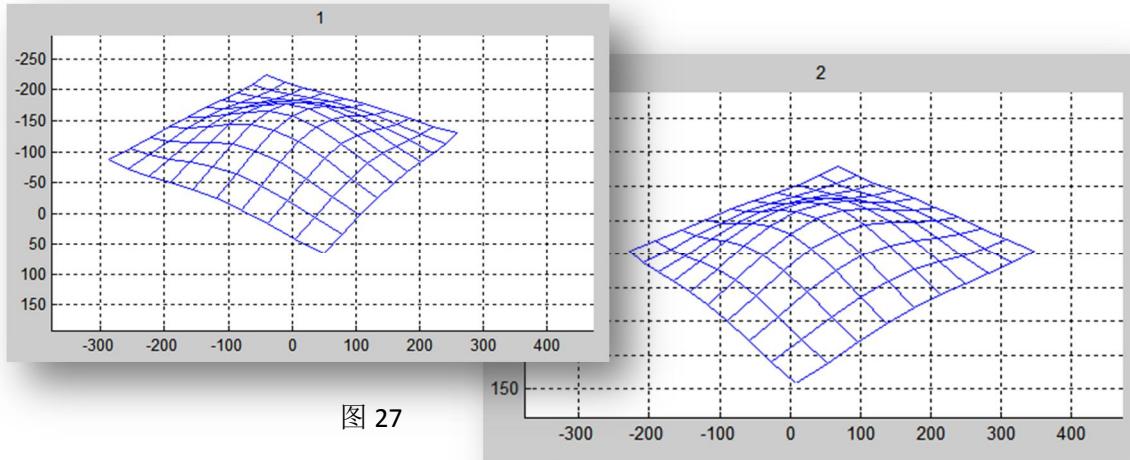


图 27

提取模拟图像中的特征点数据 `mn`，用 `mn2vcam` 函数转换成矢量，用 `deturn` 函数使相机朝向一致，运行 `deturn_process` 进行后期运算，还原出特征点的空间坐标。
运行结果可视化：

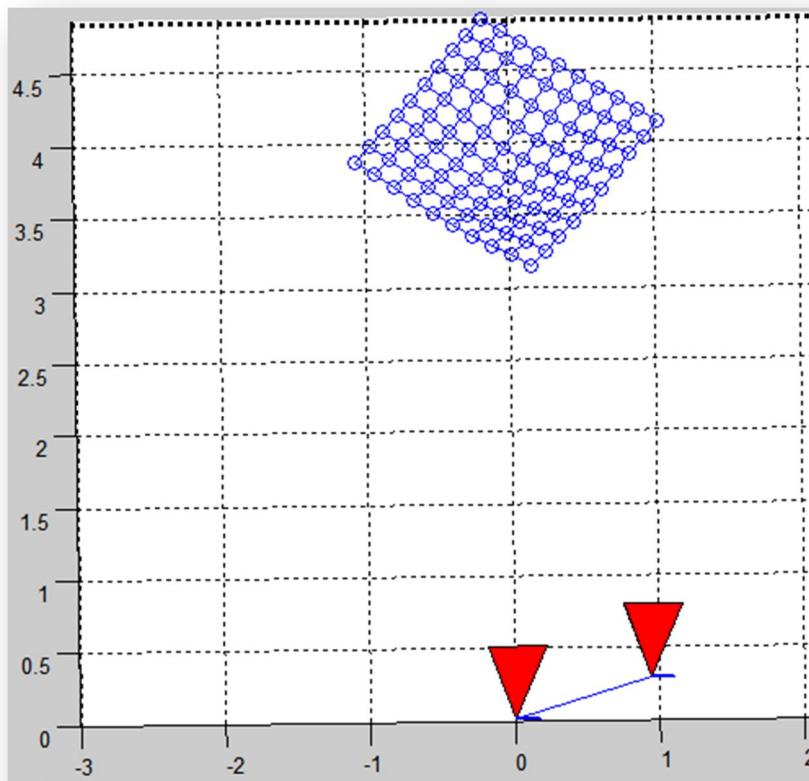


图 28

渲染后的效果：

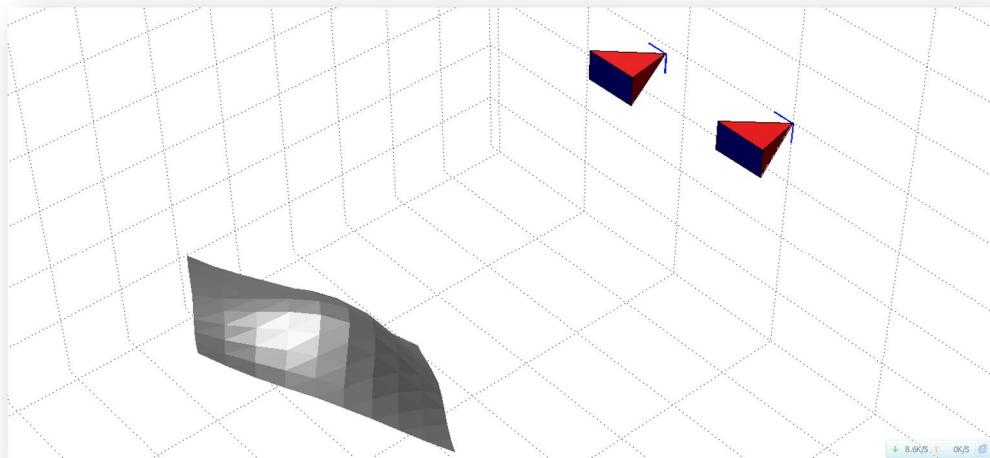


图 24

附录

参考文献

- [1] 张志涌, 等. MATLAB 教程 2010a [M]. 北京:北京航空航天大学出版社, 2010
- [2] MathWorks. MATLAB R2010a. [EB/OL]. http://www.mathworks.cn/cn/help/pdf_doc/allpdf.html 2010, 2012-11-29