(/)

深蓝学院 shenlanxueyuan.com

搜索

课程中心 (/courselist) 公开课 (/open/course/explo

学习时长
3个月左右
每周至少10个小时

答疑服务
专属微信答疑群
讲师助教均参与

作业批改
每章节设计作业
助教及时批改评优

返回讨论区 (/course/201/threads/show) / 话题

第一二次作业讲解PPT

(/user/32470)

By buxiaoyi (/user/32470) • 3天前 • 18次浏览

第一次作业-第1题

深蓝学院 shenlanxueyuan.com

- 作业要求：通过2D匹配求解F矩阵，并恢复出R,t
- 解题思路
  - 思路1：通过 Fundamental 矩阵求解
    1. 利用对极约束 $p_2^T F p_1 = 0$, 求解得到 $\boldsymbol{F}$ 矩阵
    2. 利用 $E = K^T F K$, 求解得到 $\boldsymbol{E}$ 矩阵
    3. SVD 分解E矩阵，恢复 $\boldsymbol{R}, \boldsymbol{t}$
  - 思路2：通过 Essential 矩阵求解
    1. 利用对极约束 $x_2^T E x_1 = 0$, 求解得到 $\boldsymbol{E}$ 矩阵
    2. SVD 分解E矩阵，恢复 $\boldsymbol{R}, \boldsymbol{t}$

p为像素点
x为归一化平面点

第一次作业-第1题

深蓝学院 shenlanxueyuan.com

- 代码示例
  - 思路1

```
240     {
241         cv::Mat inlier_mask;
242         cv::Mat cv_F = cv::findFundamentalMat(point_last, point_curr, cv::FM_RANSAC, 3, 0.99, inlier_mask);
243         int num_inlier_F = cv::countNonZero(inlier_mask);
244         printf("%d - F inlier: %d(%d)\n", i, num_inlier_F, landmarks.size());
245         Eigen::Matrix3d F;
246         cv::cv2eigen(cv_F, F);
247         Eigen::Matrix3d E = K.transpose() * F * K;
248         Eigen::Matrix3d R;
249         Eigen::Vector3d t;
250 
251         cv::Mat cv_E = cv_K.t() * cv_F * cv_K;
252         cv::Mat cv_R;
253         cv::Mat cv_t;
254 
255         cv::recoverPose(cv_E, point_last, point_curr, cv_K, cv_R, cv_t, inlier_mask);
256         int num_inlier_Pose = cv::countNonZero(inlier_mask);
257         printf("%d - RT inlier: %d(%d)\n", i, num_inlier_Pose, num_inlier_F);
258         cv::cv2eigen(cv_R, R);
259         cv::cv2eigen(cv_t, t);
260 
261         /* comput Twc_cur = Twc_last * T_last_curr */
262         Eigen::Matrix4d T_curr_last = Eigen::Matrix4d::Identity();
263         T_curr_last.block(0, 0, 3, 3) = R;
264         T_curr_last.block(0, 3, 3, 1) = t * t_scale;
265         Twc_curr = Twc_last * T_curr_last.inverse();
266     }
```

第一次作业-第1题

深蓝学院 shenlanxueyuan.com

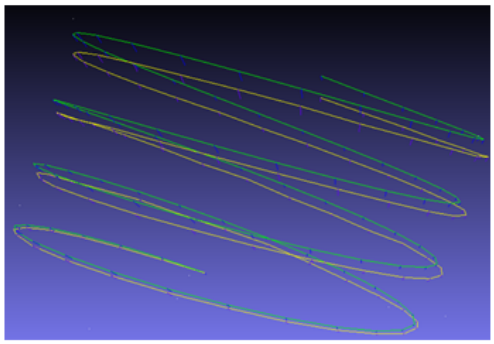
- 代码示例
  - 思路2

```
255         cv::Mat inlier_mask;
256         cv::Mat cv_E = findEssentialMat(point_last, point_curr, cv_K, cv::RANSAC, 0.999, 1.0, inlier_mask);
257         int num_inlier_E = cv::countNonZero(inlier_mask);
258
259         cv::recoverPose(cv_E, point_last, point_curr, cv_K, cv_R, cv_t, inlier_mask);
260         int num_inlier_Pose = cv::countNonZero(inlier_mask);
261         printf("%d - RT inlier: %d(%d)\n", i, num_inlier_Pose, num_inlier_F);
262         cv::cv2eigen(cv_R, R);
263         cv::cv2eigen(cv_t, t);
264
265         /* comput Twc_cur = Twc_last * T_last_curr */
266         Eigen::Matrix4d T_curr_last = Eigen::Matrix4d::Identity();
267         T_curr_last.block(0, 0, 3, 3) = R;
268         T_curr_last.block(0, 3, 3, 1) = t * t_scale;
269         Twc_curr = Twc_last * T_curr_last.inverse();
```

第一次作业-第1题

深蓝学院 shenlanxueyuan.com

- 易错点
  - recoverPose()恢复出R,t是last->cur的变换
  - recoverPose()恢复出的t缺少scale
- 优秀作业展示



```
RPE w.r.t. translation part (m)
for delta = 1 (frames) using consecutive pairs
(with SE(3) Umeyama alignment)

       max      0.044161
      mean      0.013281
    median      0.011246
       min      0.001123
      rmse      0.016462
       sse      0.025745
       std      0.009727
```

第一次作业-第2题

深蓝学院 shenlanxueyuan.com

- 作业要求：修改landmark分布和匹配点像素误差，评估精度
- 解题思路
  - 修改landmark分布

```
38      std::mt19937 gen{12345};
39      std::normal_distribution<double> d_x{0.0, 4.0};
40      std::normal_distribution<double> d_y{0.0, 10.0};
41      std::normal_distribution<double> d_z{0.0, 10.0};
42      for (int i = 0; i < 200; i ++)
43      {
44          Eigen::Vector3d pt;
45          pt[0] = std::round(d_x(gen));
46          pt[1] = std::round(d_y(gen));
47          pt[2] = std::round(d_z(gen));
48          points.push_back(pt);
49      }
```

  - 修改匹配点像素误差

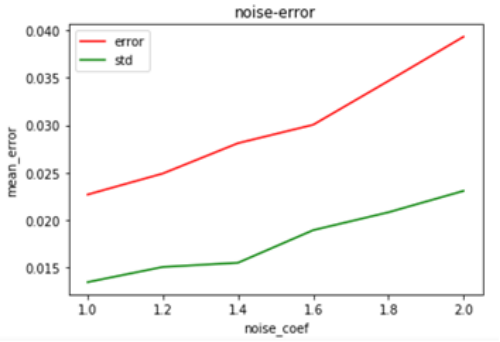
```
93      std::mt19937 gen{12345};
94      const float pixel_sigma = 1.0;
95      std::normal_distribution<> d{0.0, pixel_sigma};
96
110         float noise_u = add_noise ? std::round(d(gen)) : 0.0f;
111         float noise_v = add_noise ? std::round(d(gen)) : 0.0f;
112
113         Eigen::Vector3d ft = K * cP;
114         int u = ft[0]/ft[2] + 0.5 + noise_u;
115         int v = ft[1]/ft[2] + 0.5 + noise_v;
116         Eigen::Vector2i obs(u, v);
117         features.push_back(obs);
```

第一次作业-第2题

深蓝学院 shenlanxueyuan.com

- 易错点
  - 实验的数据量过少，一两组数据就得出结论
  - 在求解F矩阵之前，没有检查两组特征点的数目是否相同
- 优秀作业展示

分布范围	max	mean	median	min	rmse	sse	std
dx{0,4} dy{0,10} dz{0,10}	0.182712	0.050850	0.040726	0.006184	0.061398	0.358118	0.034409
dx{0,4} dy{0,4} dz{0,10}	0.225129	0.057744	0.044603	0.004413	0.072587	0.500541	0.043983
dx{0,4} dy{0,15} dz{0,10}	0.207232	0.046480	0.038294	0.004948	0.057749	0.316816	0.034271



第一次作业-第3题

深蓝学院 shenlanxueyuan.com

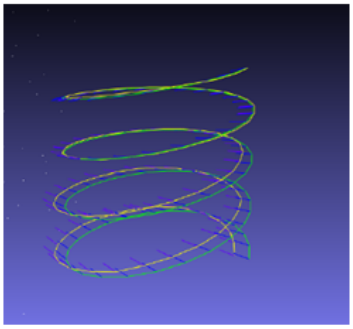
- 作业要求：用OBR_SLAM2求解F矩阵及恢复R,t的代码替换OpenCV函数
- 解题思路

```
322 void FindFundamental(const vector<cv::Point2f>& vPts1, const vector<cv::Point2f>& vPts2,
323                       const vector<cv::DMatch> &vMatches12, vector<bool> &vbMatchesInliers,
324                       cv::Mat &F21, float &score, float sigma, int maxIterations)
325 {
326     // Number of putative matches
327     const int N = vbMatchesInliers.size();
328     {
348
349     // Perform all RANSAC iterations and save the solution with highest score
350     std::mt19937 gen{12345};
351     std::uniform_int_distribution<int> d(0, vMatches12.size()-1);
352     for(int it=0; it<maxIterations; it++)
353     {
354         // Select a minimum set
355         for(int j=0; j<8; j++)
356         {
357             int idx = d(gen);
358
359             vPn1i[j] = vPn1[vMatches12[idx].queryIdx];
360             vPn2i[j] = vPn2[vMatches12[idx].trainIdx];
361         }
362
363         cv::Mat Fn = ComputeF21(vPn1i,vPn2i);
```

第一次作业-第3题

深蓝学院 shenlanxueyuan.com

- 易错点
  - 两种方法最好给出耗时方面的对比
- 优秀作业展示



```
RPE w.r.t. translation part (m)
for delta = 1 (frames) using consecutive pairs
(with SE(3) Umeyama alignment)

       max      0.182443
      mean      0.045239
    median      0.037978
       min      0.002942
      rmse      0.055795
       sse      0.295747
       std      0.032658
```

第二次作业-第1题

深蓝学院 shenlanxueyuan.com

- 作业要求：用OBR_SLAM2跑通EuRoC的MH03数据集，并估计误差
- 易错点
  - 双目运行的时候，只送了左目的图像路径
  - 评估单目时没有进行尺度对齐
  - EuRoC的ground truth数据格式没有转换为TUM格式
  - 单目和双目的评价方式不一样，应该统一评估RPE误差

第二次作业-第2题

深蓝学院 shenlanxueyuan.com

- 作业要求：将OBR_SLAM2中筛点函数由四叉树方法替换为网格筛选法
- 解题思路

直接替换提点函数即可

```
1043 void ORBextractor::operator()( InputArray _image, InputArray _mask,
1044                                 vector<KeyPoint>& _keypoints, OutputArray _descriptors)
1045 {
1046     if(_image.empty())
1047         return;
1048
1049     Mat image = _image.getMat();
1050     assert(image.type() == CV_8UC1 );
1051
1052     // Pre-compute the scale pyramid
1053     ComputePyramid(image);
1054
1055     vector < vector<KeyPoint> > allKeypoints;
1056     //ComputeKeyPointsOctTree(allKeypoints);
1057     ComputeKeyPointsOld(allKeypoints);
1058
1059     Mat descriptors;
```

第二次作业-第3题

深蓝学院 shenlanxueyuan.com

- 作业要求：将OBR_SLAM2中提点函数替换为OpenCV的ORB特征提取函数
- 解题思路

在operator()函数中，替换原有提点函数

```
1043 void ORBextractor::operator()( InputArray _image, InputArray _mask, vector<KeyPoint>& _keypoints,
1044                                 OutputArray _descriptors)
1045 {
1046     if(_image.empty())
1047         return;
1048
1049     Mat image = _image.getMat();
1050     assert(image.type() == CV_8UC1 );
1051
1052     // Pre-compute the scale pyramid
1053     ComputePyramid(image);
1054
1055     Ptr<ORB> orb_detector = ORB::create(nfeatures,scaleFactor,nlevels,31,0,2, ORB::HARRIS_SCORE,31,iniThFAST);
1056     orb_detector->detectAndCompute(image, Mat(), _keypoints, _descriptors);
1057
1058     return;
1059
```

第二次作业-第3题

深蓝学院 shenlanxueyuan.com

- 易错点
  - OpenCV函数提取特征点时数目过少，初始化失败
  - 没有计算金字塔图像

第二次作业-第3题

深蓝学院 shenlanxueyuan.com

- 优秀作业

统计量	单目			双目		
	原始	网格筛点	opencv 提取	原始	网格筛点	opencv 提取
最大值	0.073501	0.077009	0.104836	0.106445	0.090523	0.133159
均值	0.029408	0.030415	0.038943	0.031218	0.028366	0.025727
中值	0.023154	0.024427	0.034077	0.023386	0.020251	0.023487
最小值	0.002979	0.004178	0.009756	0.001384	0.002128	0.003532
均方根误差	0.033785	0.035140	0.044533	0.038436	0.034295	0.044894
残差平方和	0.197469	0.191400	0.291534	4.431972	3.528522	6.046501
标准差	0.016631	0.017599	0.021602	0.022423	0.019275	0.027168

第二次作业-第3题

深蓝学院 shenlanxueyuan.com

- 优秀作业

代码	平均误差	方差	平均跟踪时间(s)	特征点分布
单目原始	0.853	1.00	0.03049	均匀
双目原始	0.0353	0.0183	0.07063	均匀
网格筛选单目	0.906	1.13	0.02498	比较均匀
网格筛选双目	0.0357	0.0164	0.06988	比较均匀
opencv单目	0.940	1.02	0.03119	比较集中
opencv双目	0.0733	0.0459	0.07589	比较集中

0 回复

还没有回复，赶快添加一个吧！

添加回复

源码

添加回复