思路一

面向动静态物体极目标的视觉SALM

设定好物体的属性：

车 人 自行车之类的为动态物体

路标，广告牌，路灯之类的为静态物体

正常做法都是将图像进行语义分割后，只取静态物体上的特征点进行定位和建图

但是如果是在地下停车场，或者路边停放的有静止的车的情况下，先验信息告诉我们不能用车上边的特征点，这样的话就有点浪费，会增加跟踪失败的可能性

如此，增加运动一致性检测就非常有必要

因此，在语义分割之后，进行一个运动一致性检验，目的是利用更多的静态特征点完成定位

（可以动态物体设置为 高动态 中动态 低动态） 如果判定该物体为动态物体了，通过运动一致性检验看是否为低动态物体，若是低 可以用里边的特征点

（可以将特征点提取加速融合进去，）

思路二

结合yolo 跟踪线程基于关键目标 使用光流法追踪代替特征点匹配 理论上是可以加速的，前提是动态物体的剔除部分一定要做好，否则很容易产生漂移

（对关键目标做一个特征点均匀化）

思路三

结合yolo先用静态特征点完成完成位姿初估计，然后利用运动一致性和几何约束，将动态框里的静态点拿出来，与之前的静态点一起对位姿进行优化 如图模型所示：

图示

描述已自动生成

用yolo检测两种属性的框 静态框，动态框，设两个个阈值

比如静态框里需要判定百分之80以上的特征点为动态点，才能认定 为动态物体

而动态属性的框 只需要有百分之40以上就可以认为是动态

判断静态也是如此

最终思路

突然有了一个感觉还不错的思路，有一篇论文是基于静态目标做的(皮家豪)，但是我有一个疑问，如果初始化关键目标达不到他的数量该如何是好？？？

应该是有把八对匹配点，最好有八个静态目标，每个目标里拿出一个匹配点，

这种情况可能只有在复杂路口，或闹市才能满足，平时的话或许精度也就一般了，因此我想可以在一般的路况下，就用简单的动态目标剔除，剩余点匹配求解位姿，一旦找到了满足要求的静态目标(比如设为8个，这可能比较难满足)，使用静态目标里的匹配点求解位姿，这样确实能规避一些动态物体的影响，但是静态目标要是绝对静的，不可能移动的物体！！！！

3.10.2023--(动态物体的后检验用局部光流(多视图几何)，如果满足静态物体的阈值)

下边我将开始对自己所看的博客，文献，以及任何资料都做一个记录，以免有时候忘记想要查看的内容

2023

3-23 用的检测后的txt文件都是别人弄好的，然后我自己也弄了一次，主要是

写一个脚本将yolo5检测好的txt文件按照改好的读取格式的转换

脚本我放进了记录文件夹 yolo-to-format.py

使用该脚本已经没有问题了，我也测试过用我标注好的txt去运行，没有问题

./Examples/RGB-D/rgbd\_tum Vocabulary/ORBvoc.txt Examples/RGB-D/TUM3.yaml /media/wangxudong/xudong/TUM-dynaslam-data/rgbd\_dataset\_freiburg3\_walking\_xyz /media/wangxudong/xudong/TUM-dynaslam-data/rgbd\_dataset\_freiburg3\_walking\_xyz/associate.txt ../output/ 换成自己txt文件路径

但是如果yolo没有检测到某张图片的任何信息，暂时先给他按顺序填补一下，让他能运行下去，然后在来想办法

用一个脚本给他补全 add\_null\_txt.py 放进记录文件夹保存了

格式什么的都没问题，但是我发现他把car都识别成打上person标签了

因为上边转换时是按造coco里边的定义来的，所以干脆写个脚本把person换成car就好了，暂时先不写了

---------------------------------------------------------------------------------------

addflow-stereo分支

今天把光流验证部分加进去了，发现确实有点效果

不过需要注意的是切换分支后一定要编译一下，不然还是用的别的分支

说一下问题：我发现加入光流后，确实会有一部分静止的车被检测出来，让我们能用他们的特征点，但是只要一走进，静止的车又变成了红框，并且里边的点也变成了红点，不知道为啥？

猜测可能是光流检测近处的东西不好使？？？

具体原因明天在继续探究

---------------------------------------------------------------------------------

3-25

addDisparityBaseFlow 分支

接着昨天的问题，仅用光流是不可行的

(更何况我的光流还只用了前后两帧的数据，误判的可能性非常大，如果把阈值设置小点，静止的物体也会判定为运动的，如果阈值设置大点，运动的物体有可能达不到标准而被判定为静止的)

因为我用的是双目的，所以加入了视差检测方法

这样的话，我就有了两个阈值 光流阈值和视差阈值

同时满足我的两个阈值才会判定为动态的，于是我感觉应该会好一点了，但其实好像并没有。

--------------------------------------------------------------------------------------------

addSW-Base-DisparityAndFlow 分支

所以我决定增加更多的数据关联，把关键帧数据拉进来可能比较麻烦，注意的地方太多了，于是乎我决定用一个滑动窗口，关联5个连续普通帧，不知道效果如何。

我已经成功的添加了一个滑动窗口来进行数据关联，但是我发现用的还是连续帧的数据，窗口并没有啥用，而且很慢

--------------------------------------------------------------------------------------------

addSW-Base-DisparityAndFlowPLUS 分支

于是我再开一个分支 plus版 尝试把数据关联搞定

在PLUS分支 数据关联部分搞定了，但是非常非常的慢，看代码就知道了，但是将前边的数据利用起来了，会计算一个平均视差，平均光流，保持窗口里始终有5帧，进来一帧新的就把旧的踢掉

我的分支太多了，所以画个箭头避免忘记各个分支都干了什么

stereo-semantic (加了语义信息,读取检测框)

addflow-stereo (加了光流验证)

addDisparityBaseFlow (加了光流验证和视差法)

addSW-Base-DisparityAndFlow (关联5个普通帧,不过并没有实际关联上,虽说有窗口,但实际上还是前后两帧的数据)------速度很慢

addSW-Base-DisparityAndFlowPLUS (真正关联上5帧数据,用平均光流和平均视差判定物体属性,按理说应该会准一点,但看起来好像也一般般)------速度很慢

每次都要5帧数据算一下,太可怕了!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

引言:随着计算机视觉和机器人技术的快速发展，视觉定位与地图构建（SLAM）在移动机器人、无人驾驶汽车和增强现实等领域得到了广泛应用[1]。SLAM技术的关键挑战之一是在动态环境中实现鲁棒性。动态物体的运动可能导致误报的特征点和错误的相机位姿估计，从而降低SLAM算法的精度[2]。

特征点法SLAM，如ORB-SLAM系列算法[3,4]，已经取得了显著的性能提升。然而，它们在处理动态场景时仍然面临挑战。为了解决这一问题，许多研究者已经尝试将物体检测与SLAM算法相结合[5,6]。通过区分动态物体和静态物体，可以提高SLAM算法在动态场景下的鲁棒性。

本文提出了一种结合ORB-SLAM3和yolov5的方法，通过光流检验和视差法来区分动态物体和静态物体。我们的方法首先利用yolov5[7]进行实时物体检测。接着，对检测到的物体应用光流检验[8]和视差法[9]来判断它们是静态的还是动态的。最后，我们利用静态物体进行位姿估计以提高ORB-SLAM3在动态场景下的鲁棒性。

本文的组织结构如下：第二部分回顾了与我们工作相关的文献，包括SLAM算法、物体检测和光流检验。第三部分详细描述了我们的方法，包括ORB-SLAM3与yolov5的结合，以及光流检验和视差法的实现。第四部分介绍了我们在不同数据集上进行的实验以及实验结果。最后，第五部分总结了我们的工作，并提出了未来研究的方向。

[1] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., ... & Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. IEEE Transactions on Robotics, 32(6), 1309-1332.

[2] Barsan, I. A., Liu, P., Pollefeys, M., & Geiger, A. (2018). Learning to detect motion boundaries. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp.4050-4059).

[3] Mur-Artal, R., Montiel, J. M. M., & Tardos, J. D. (2015). ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Transactions on Robotics, 31(5), 1147-1163.

[4] Mur-Artal, R., & Tardos, J. D. (2017). ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Transactions on Robotics, 33(5), 1255-1262.

[5] Yang, S., Song, Y., Kaess, M., & Xiao, J. (2018). Dense object SLAM. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 6312-6319). IEEE.

[6] Zhou, T., Brown, M., Snavely, N., & Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1851-1858).

[7] Jocher, G., Stoken, A., Borovec, J., & others. (2020). YOLOv5. GitHub. [https://github.com/ultralytics/yolov5](https://github.com/ultralytics/yolov5" \t "/home/wangxudong/文档\\x/_new)

[8] Farnebäck, G. (2003). Two-frame motion estimation based on polynomial expansion. In Scandinavian conference on Image analysis (pp. 363-370). Springer, Berlin, Heidelberg.

[9] Scharstein, D., & Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision, 47(1-3), 7-42.

在第二部分，我们回顾了与我们工作相关的文献，包括SLAM算法、物体检测和光流检验。我们讨论了这些技术如何相互作用以改善SLAM算法在动态场景下的性能。在第三部分，我们详细描述了我们的方法，包括ORB-SLAM3与yolov5的结合，以及光流检验和视差法的实现。我们解释了如何将这些方法整合在一起，以提高SLAM算法的鲁棒性。第四部分介绍了我们在不同数据集上进行的实验以及实验结果。我们展示了我们的方法在处理动态场景方面的优势，并对比了与其他现有方法的性能。最后，第五部分总结了我们的工作，并提出了未来研究的方向。我们讨论了潜在的改进和扩展，以进一步提高SLAM算法在动态场景下的性能。

贡献：

1. 我们提出了一种将ORB-SLAM3与yolov5相结合的方法，利用物体检测信息来区分动态和静态环境特征。
2. 我们引入了光流检验和视差法来进一步验证物体的动态属性，从而提高我们方法的准确性。
3. 我们在动态场景下对位姿估计进行了优化，仅使用静态点进行计算，提高了鲁棒性。
4. 通过在多个数据集上进行实验，我们证明了我们的方法在动态场景下优于现有的SLAM方法。

未来研究方向：

1. 探索更加有效的动态物体检测和跟踪方法，进一步提高SLAM系统在复杂动态场景下的性能。
2. 研究其他类型的传感器，如激光雷达，以获得更精确的物体位置和速度信息，从而提高位姿估计的准确性。
3. 考虑使用深度学习方法来预测动态场景中物体的运动，进一步改善SLAM系统的鲁棒性。
4. 实现实时性能优化，使我们的方法更适合于实际应用，例如自动驾驶和机器人导航。

通过强调我们方法的贡献和优势，以及未来的研究方向，我们为论文的其余部分奠定了基础。这将帮助读者更好地理解我们的工作以及它在现有研究中所处的位置。

3-27

找不到11-21序列的kitti数据集的GT，不知道为什么，所以准用kitti01序列和04序列来验证一下改进后是否能有所提升

需要做一下两个数据集的txt文件

kitti04 对比实验 记录

evo\_ape kitti orb3-kitti04.txt 04-GT.txt -p --plot\_mode=xyz -a -s -v

evo\_ape kitti orb3\_addDisparityBaseFlow-kitti04.txt 04-GT.txt -p --plot\_mode=xyz -a -s -v

orb3-kitti04 rpe

max 0.048526

mean 0.018367

median 0.016668

min 0.002398

rmse 0.020340

sse 0.111704

std 0.008740

orb3\_addDisparityBaseFlow-kitti04 rpe

max 0.048829

mean 0.017785

median 0.016981

min 0.002938

rmse 0.019449

sse 0.102134

std 0.007873

orb3-kitti04 ape

max 0.449099

mean 0.215769

median 0.196441

min 0.074938

rmse 0.235491

sse 15.028543

std 0.094338

orb3\_addDisparityBaseFlow-kitti04 ape

max 0.431474

mean 0.196339

median 0.171750

min 0.042974

rmse 0.219618

sse 13.070939

std 0.098404

kitti04好像是提升了一点点的样子，不过也可以忽略了，才5%左右

kitti01一点效果没有，不知道为什么，先不写数据了

然后我发现运行速度太慢了，0.24秒一帧，而原版的只需要0.02秒，我这也太慢了，开个分支试试能否对速度优化一下

此时我是处于addDisparityBaseFlow

创建分支Faster-addDisparityBaseFlow就是对视差和光流优化一下，只计算感兴趣的区域里的，不计算整张图像了,看看能否提升点速度

效果可以，改进后直接变成了 0.02秒一帧，很好，还是老问题，改进后的精度并未提升？？？

我现在要改一个mono模式下加上光流法，去验证是否有效果，去除动态物体

创建分支mono-addflow

但是我突然想到能否看一下单目运行kitti01数据集与真值评估，结果发现单目模式下得到的KeyFrameTrajectory.txt是TUM格式的，无法与真值进行评估，这好像有点麻烦，那我怎么知道我的精度呢，所以需要先解决这个问题

下载了apploauto数据集，动态环境是满足我的要求的，但是文件读取格式还得修改，这就有点烦烦了，估计要改的地方还不少呢，而且轨迹评估还得用它提供的，感觉很麻烦的样子

3-28

突然想起来不是滑动窗口做过一个改进吗，忘记评估一下定位精度了

数据如下： addSW-Base-DisparityAndFlowPLUS 分支 运行04数据集

因为这个数据集比较小

median tracking time: 0.824668

mean tracking time: 1.34108

ape：

max 0.370668

mean 0.164458

median 0.141951

min 0.057500

rmse 0.181978

sse 8.974459

std 0.077909

不知道为什么相对位姿没有什么优化，不过没关系，有的论文也是只写绝对位姿

rpe:

max 0.053189

mean 0.020507

median 0.019153

min 0.003098

rmse 0.021928

sse 0.129824

std 0.007765

Faster-addSW-Base-DisparityAndFlowPLUS 分支

现在使用gpu进行了优化优化过后的

median tracking time: 0.605327 快了一丢丢

mean tracking time: 0.623804

不过还是不行，需要想办法继续优化

于是创建分支moreFaster-addSW-Base-DisparityAndFlowPLUS

在这里我只想计算感兴趣区域内的视差和光流

median tracking time: 0.0402428

mean tracking time: 0.0485587 速度很奈斯，下边看精度

运行kitti04数据集

ape：

max 0.374275

mean 0.171392

median 0.167447

min 0.030519

rmse 0.190500 提升19% 挺好

sse 9.834682

std 0.083156

ORB\_SLAM3\_addsemantic 里边分支太多了

于是有关单目的部分我就防到 ORB-SLAM3 文件里开分支

mono-addsemantic-tum 分支 想用单目跑动态数据集评估一下，总是出现问题

Optimizer::GlobalBundleAdjustemnt(mpAtlas->GetCurrentMap(),20);

这个地方应该有内存问题，明天来了要解决一下