http://bhxb.buaa.edu.cn jbuaa@buaa.edu.cn
DOI: 10.13700/j.bh.1001-5965.2023.0552

基于自适应阈值和速度优化的轻量化语义VSLAM方法

齐浩[1]，付悦欣[1]，胡祝华[1,*]，吴佳琪[1]，赵瑶池[2]

(1. 海南大学 信息与通信工程学院，海口 570228；2. 海南大学 网络空间安全学院，海口 570228)

摘 要：视觉同步定位与地图构建（VSLAM）是一种利用视觉等传感器来获取未知环境信息的技术，广泛应用于无人驾驶、机器人、增强现实等领域。然而，室内场景下的VSLAM对动态对象进行像素级的语义分割存在较高的计算开销，并且光照变化使得动态物体的外观也发生变化，导致其与静态环境产生遮挡或混淆。针对以上问题，提出了一种基于自适应阈值和速度优化的轻量化语义VSLAM模型。采用了轻量化的一阶段目标检测网络YOLOv7-tiny，结合光流算法，有效地检测了图像的动态区域，并对不稳定特征点进行了剔除。同时，特征点提取算法基于输入图像的对比度信息，自适应地调整阈值。结合二进制词袋与局部建图线程精简的优化方法，加快了加载和匹配速度，提高了系统在室内动态场景下的运行速度。实验结果表明：所提算法在室内高动态场景下能够有效地剔除动态特征点，提高了相机的定位精度。在运行速率方面平均处理速度达到了19.8 FPS，在实际场景下可以满足实时性的需求。

关 键 词：VSLAM；动态场景；YOLOv7-tiny；自适应阈值；特征点

中图分类号：V221+.3; TB553

文献标志码：A **文章编号**：1001-5965(2025)07-2562-11

视觉同步定位与地图构建(visual simultaneous localization and mapping, VSLAM)由于其低成本传感器,易于融合其他传感器以及更丰富的环境信息的优势而发展迅速。其不仅能提供自我定位、位姿估计和环境地图构建的方法,而且能在未知且复杂的环境下提供丰富的场景信息[1]。目前的VSLAM方法存在诸多不足,具体如下:①现有的VSLAM系统通常假设环境是静态的,而实际生活与应用场景都是非静止的,VSLAM算法的精度不够理想[2];②多数VSLAM算法没有充分利用场景信息进行语义分析,并且部分算法时间复杂度过高,难以满足实时性的需求;③目前动态场景下特征点的误匹配概率较高,整体系统的稳定性较差;④在环境光照变化明显的场景中,现有VSLAM系统不能随着环境改变而调整阈值参数,导致系统的鲁棒性不高[3]。

现有的VSLAM系统主要分为依靠直接法的VSLAM和特征点法的VSLAM。关于直接法,Newcombe等[4]提出了稠密跟踪与地图构建(Dense Tracking and Mapping, DTAM),其利用不同帧深度图直接匹配来估计位姿,但在光照变化场景下精度较差。Forster等[5]提出了半直接单目视觉里程计(semi-direct monocular visual odometry, SVO),其通过角点提取和利用周围像素信息进行位姿估计,实现了高速度和较高准确性。然而,缺乏后端优化和闭环检测线程使SVO容易累积误差,对长时间位姿估计

收稿日期：2023-08-28；**录用日期**：2024-03-22；**网络出版时间**：2024-05-08 10:50

网络出版地址：link.cnki.net/urlid/11.2625.V.20240507.1513.002

基金项目：国家重点研发项目 (2022YFD2400504)；国家自然科学基金 (62161010)；海南省重点研发计划项目 (ZDYF2022SHFZ039, ZDYF2022GXJS348)

* **通信作者**. E-mail：eagler_hu@hainanu.edu.cn

引用格式：齐浩，付悦欣，胡祝华，等. 基于自适应阈值和速度优化的轻量化语义VSLAM方法[J]. 北京航空航天大学学报，2025，51(7)：2562-2572. QI H，FU Y X，HU Z H，et al. A lightweight semantic VSLAM approach based on adaptive thresholding and speed optimization[J]. Journal of Beijing University of Aeronautics and Astronautics，2025，51(7)：2562-2572 (in Chinese).

的精度构成挑战。关于特征点法，Davision 等[6] 提出了单目即时定位与地图构建（monocular simultaneous localization and mapping, Mono SLAM），其采用特征点法和扩展卡尔曼滤波算法，能够在小范围环境中完成自身位姿估计，但是在大环境中会出现大范围漂移且精度较低的问题。Klein 等[7] 的并行跟踪与建图（parallel tracking and mapping, PTAM）系统引入了基于非线性优化的方法，通过并行化跟踪线程和建图线程，采用关键帧联合优化的思路，摒弃了逐帧处理的方式。在定向 FAST 与旋转 BRIEF SLAM2（oriented FAST and rotated BRIEF SLAM2, ORB-SLAM2）[8] 的基础上，Campos 等[9] 进一步提出了 ORB-SLAM3，通过引入惯性测量单元（inertial measurement unit, IMU）数据进行辅助，实现了多传感器融合的视觉惯性 SLAM 系统。这一创新为多传感器融合 VSLAM 系统提供了新的思路和解决方案。为高效利用场景图像信息，Shen 等[10] 基于 ORB-SLAM3 算法，将图像特征与离线的词袋模型相结合实现一种实时在线更新词袋的闭环检测模型。

上述 VSLAM 算法都假设机器人所处的环境为静止状态，因此无法对复杂的动态场景进行识别。动态场景中的 VSLAM 算法主要可以分为以下 2 类。

第 1 类主要是借助传统的方法利用静态特征点进行位姿估计和地图构建。基于传统的几何方法如帧差法和背景扣除法[11-12]，都是通过对输入图像的像素进行操作来检测环境中存在的动态对象。除此以外，还可以利用光流法来判别场景中的运动物体。Klappstein 等[13] 利用光流法计算动态对象与场景中光流运动的偏离程度，从而实现动态物体的分割。Derome 等[14] 利用不同图像之间的残差计算光流，之后再通过残差场中的点来判定动态对象。光流法不仅能够准确表现运动物体的信息，还可以提供丰富的场景数据用于三维重建。但是光流法的前提是亮度恒定，系统极易受光照变化的影响。

第 2 类借助图像中的语义信息，对输入图像中的动态对象进行检测并将图像中包含的动态区域剔除。动态语义 SLAM（dynamic semantic SLAM，DS-SLAM）[15] 在 VSLAM 的三大线程之外建立了并行的语义分割线程，利用 SegNet[16] 网络对动态对象进) 行检测分割。这种多线程处理虽然可以提高定位效果，但也增加了系统的复杂性，导致系统更加冗余庞大。Bescos 等[17] 提出了 DynaSLAM 系统，通过采用多视图几何和像素级的语义分割网络来处理动态对象，并借助静态图对动态对象遮挡的背景进行修复。然而像素级的语义分割网络通常需要进行大量的计算，无法满足实时性要求。Fu 等[18] 和 Cai 等[19] 将注意力模块融入掩码区域卷积神经网络（mask region-based convolutional neural network, Mask R-CNN）[20] 的特征提取网络，以提升动态目标的特征提取和分割精度。但受限于 Mask R-CNN 神经网络的特性，该系统仍然难以满足实时性的要求。Qi 等[21] 和 Chang 等[22] 均采用 YOLO 网络结合光流法对动态对象进行剔除，但是没有考虑到光流法的提取效果易受不同环境下光照变化的影响。

通过对以上研究现状的分析可知，尽管 VSLAM 系统在动态环境中表现出了一定的适用性，但仍然存在系统冗余和复杂性较高的问题，以及未充分考虑环境光照变化对动态 VSLAM 的影响。针对目前 VSLAM 在动态场景下存在的挑战，本文进行了以下工作。

1）为了提高对动态目标的检测速度，本文在对 VSLAM 算法改进的基础上，构建了基于 YOLOv7-tiny 的轻量化语义 SLAM 算法模型。通过将 YOLOv7-tiny 模型与卢卡斯-卡纳德（Lucas-Kanade, LK）光流算法相结合，本文算法不仅实现了对图像动态区域的精准检测，还高效剔除了动态特征点，从而充分满足了 VSLAM 系统实时运行的需求。

2）本文提出了基于自适应阈值的特征点提取算法，并对 VSLAM 系统进行速度优化。在传统的特征点提取算法中，采用固定的阈值常常使得系统难以适应各种环境变化，从而导致其鲁棒性相对较弱。为此，本文基于输入图像的对比度信息实时对提取阈值的大小进行调整，从而提高 SLAM 系统的稳定性和特征点提取效率。此外，通过二进制词袋与局部建图线程简化相结合的优化方法，进一步提升了 SLAM 系统的速度。

1　本文方法

本文将性能优异的 YOLOv7-tiny 网络结合光流法作为动态物体的剔除模块，融合到基于自适应阈值和速度优化的 ORB-SLAM2 的系统中，改进后的视觉系统整体框架如图 1 所示。在初始化阶段，系统根据时间戳加载输入图像和二进制词袋模型，读取相机标定等参数，完成 VSLAM 系统的初始化。之后系统用 YOLOv7-tiny 网络提取结果，并用光流法进行动态物体判断和剔除。与此同时，系统根据每张图像的对比度自适应的确定特征点提取过程中的阈值，跟踪线程依据语义信息提取静态物体上的特征点，并利用特征点进行局部地图跟踪。

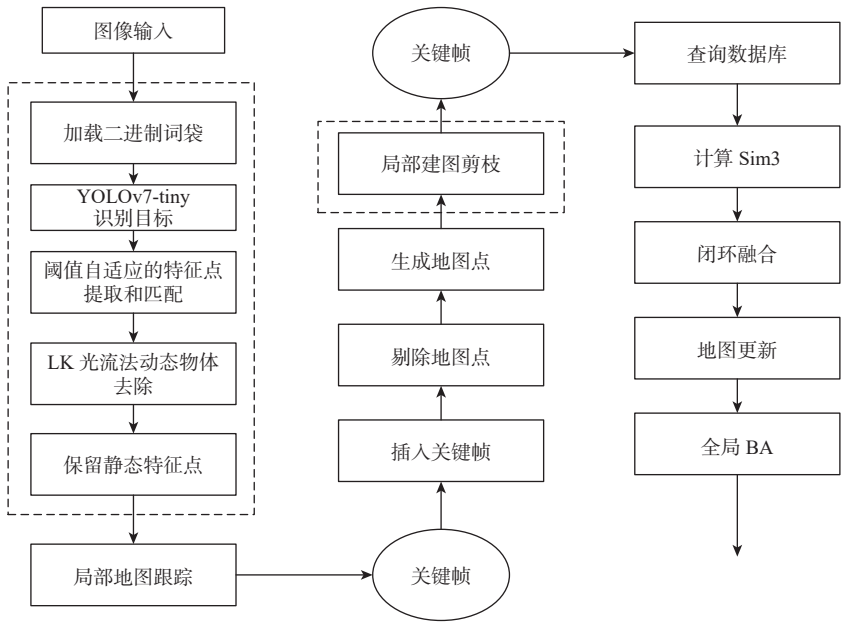


图1 系统整体框

Fig. 1 System structure

在局部地图构建线程中，系统主要完成了关键帧插入、地图点的剔除和生成。

1.1 动态特征点检测

YOLOv7[23]是近年来提出的实时目标检测模型之一，其在5～160帧/s的目标检测模型中都具有较高的速度和准确度。在GPU V100上，当实现30帧/s的实时检测速度时，其能够达到56.8%AP的准确率。YOLOv7-tiny是专门针对边缘GPU设备而设计的模型，在具有较高准确度的同时，也能够满足语义VSLAM系统对于实时运行的速率要求。相较于YOLOv7模型，YOLOv7-tiny模型在规模上进行了缩减，实现了高效且紧凑的设计。同时，YOLOv7-tiny也仍然延续了YOLOv7网络的模型重参数化和标签分配策略，对新型的高效层聚合网络(efficient layer aggregation networks, ELAN)结构进行精简，并对级联型的网络进行模型的缩放以此来保持网络模型初始设计时的最佳结构和性能。

在使用YOLOv7-tiny模型进行物体检测之前，首先根据物体的动态特性将室内物体划分为2类：一级动态物体和二级动态物体。一级动态物体包括人和动物，以及被人移动的物体，如书本和杯子。二级动态物体基本不会移动，如冰箱、洗衣机和电视。根据YOLOv7-tiny模型的检测结果，若为一级动态物体，则利用LK光流算法进一步进行判定。若为二级动态物体，则直接视为静态物体。相比于将所有特征点进行动态检测的方法，该策略在高动态数据集下能够提供更好的定位精度，同时也减少了系统的实时性影响。

光流法通过比较2幅连续图像中对应像素的亮度值变化，能够精确地推断出这些像素在图像中的运动方向和大小。假设将图像的灰度视为时间的函数，在时刻t，位于图像坐标(x,y)处的ORB特征点的灰度表示为

$$I(x,y,t)=I(x+\mathrm{d}x,y+\mathrm{d}y,t+\mathrm{d}t) \tag{1}$$

式中：t和$t+\mathrm{d}t$分别为相邻图像帧对应的时间；$I(x,y,t)$和$I(x+\mathrm{d}x,y+\mathrm{d}y,t+\mathrm{d}t)$则为对应的灰度。基于光流法的灰度不变假设，推导出了相关结果[22]。

$$\left[\boldsymbol{I}_x\boldsymbol{I}_y\right]_k\begin{bmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{bmatrix}=-\boldsymbol{I}_k \qquad k=1,\cdots,k^2 \tag{2}$$

式中：$\boldsymbol{u}$为特征点在X轴上的运动速度；$\boldsymbol{v}$为在y轴上的速度；$\boldsymbol{I}_x$为图像在该点处x轴方向的梯度；I_y为图像在该点处y轴方向的梯度；$\boldsymbol{I}_k$为特征点灰度对时间的变化量。在LK光流中，以特征点为中心假设6×6的窗口，即$k=6$，内部36个像素具有相同的运动速度和方向。式(2)为一个超定方程，常用最小二乘法求解，如果一次迭代效果不佳，可以多迭代几次即可求解。

当相在拍摄过程中发生移动时，静态部分由于相机的运动也会产生光流矢量。

$$\begin{bmatrix}\boldsymbol{U}\\ \boldsymbol{V}\end{bmatrix}=\frac{1}{N}\sum_{i=1}^{N}\begin{bmatrix}\boldsymbol{u}_i\\ \boldsymbol{v}_i\end{bmatrix} \tag{3}$$

式中：$\boldsymbol{U}$为静态部分的特征点在x轴上的运动速度；$\boldsymbol{V}$为静态部分的特征点在y轴上的运动速度。接下来，利用式(2)计算得到的动态光流速度$\boldsymbol{u}$和$\boldsymbol{v}$，与静态部分的速度$\boldsymbol{U}$和$\boldsymbol{V}$进行计算，以判断特征点是否为动态特征点。

$$\sqrt{(\boldsymbol{U}-\boldsymbol{u})^2+(\boldsymbol{V}-\boldsymbol{v})^2}>\varepsilon_{\mathrm{th}} \tag{4}$$

ε_{th}由实验结果所得，为静态平均光流矢量的 2 倍。最后根据求解结果，如果特征点的光流大于阈值ε_{th}，则该点被认为是动态点。

1.2　FAST 图像特征点提取

基于加速分段测试的特征提取（features from accelerated segment test, FAST）是一种用于快速目标检测和识别的角点检测器。在光照不足或者光线过强的情况下，图像细节的信息缺失较多，相应图像的对比度也大幅下降。但是在 VSLAM 系统运行过程中，FAST 特征点提取过程中的阈值参数是固定不变的，不能根据环境的变化动态调整或者做出改变。在这样的情况下，如果仍然沿用和正常光照条件下同样的阈值参数，就会导致 VSLAM 系统不容易提取到图像中的特征点，甚至因初始特征点较少而难以完成 VSLAM 系统的初始化，进而影响后续特征匹配、关键帧的生成等过程。

针对以上问题，本文采用了一种基于自适应阈值的改进 FAST 特征点提取方法，根据图像的对比度信息自适应调整原本系统中设置的固定阈值。一方面，能够提高系统在不同光照条件下的鲁棒性和稳定性。另一方面，在光照正常且图像纹理结构特征丰富、特征点信息充足的环境中，能够在保持系统性能和准确性的同时，适度减少提取的特征点数目，进一步降低了系统整体的算力消耗。

图像对比度是指像素点和周围临近像素之间黑与白灰度层级的反差，视为图像中黑色与白色的比值[24]。一般情况下图像的对比度越高，图像表现的层次越丰富，包含的信息与细节也就越丰富。图像对比的计算公式如式（6）所示，C代表图像对比度，i和j分别表示当前像素和其邻域像素的灰度值，$\mu(i,j)$表示当前像素与邻域像素之间的灰度差，$P_\mu(i,j)$表示相邻像素间灰度差为μ的概率。在图像中像素邻域通常有四邻域、对角邻域和八邻域 3 种，其中四邻域和对角邻域叠加可得八邻域，在本文中使用的是四邻域。

$$\mu(i,j)=|i-j| \tag{5}$$

$$C=\sum_{\mu}\mu(i,j)^2P_\mu(i,j) \tag{6}$$

FAST 的阈值根据文献 [8] 的推荐设置为 20 和 7，该推荐基于其在 ORB-SLAM2 上对 EuRoc[25] 数据集的多次实验。本文根据求出的图像对比度C，设计自适应阈值λ：

$$\lambda=\alpha\ln C+\beta \tag{7}$$

式中：α和β为自适应参数，α范围为 0.5～1，β范围为 10～15，取值根据实验数据确定。自适应阈值λ能够使系统更好地适应不同的图像和场景。对数变换可以使得对比度C在较小的范围内取值时，阈值λ有一个较大的变化；而在C取值较大时，λ的变化相对较小。当图像对比度较低时，算法能够更加敏感地调整阈值；而当对比度较高时，阈值的调整相对较为平缓，从而更好地适应不同对比度的图像。

接下来根据本文设置的自适应阈值λ，进行特征点的提取。如图 2 所示，FAST 特征点的提取以检测点P为圆心，以半径为 3 的像素点构成一个检测圆，在检测圆上选取 16 个连续像素点，比较圆上像素点的灰度值与检测点P的灰度值的大小。

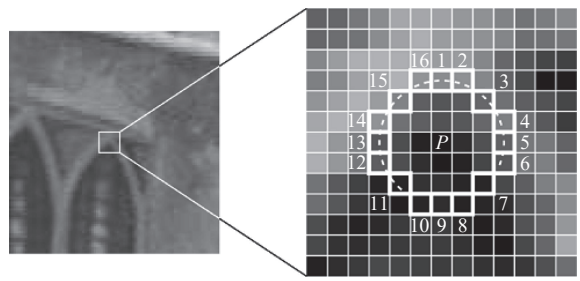


图 2　FAST 特征点提取示意[26]

Fig. 2　Schematic diagram of FAST feature point extraction[26]

特征点提取分为以下 4 个步骤：

步骤 1　初步筛选：首先，计算圆周上的像素点P_1、P_9与中心像素点的像素差。如果这 2 个像素差的绝对值都小于设定的阈值，则该点不可能是一个特征点。否则，将其加入候选特征点的列表中。

步骤 2　进一步的筛选：对于候选点P，计算P_1、P_5、P_9和P_{13}与P点的像素差。如果这些像素差的绝对值中，小于阈值的个数不超过 1 个，则P点是一个候选特征点。否则，P 点不是一个特征点。

步骤 3　确定特征点：对于候选特征点P，如果其像素圆上有连续 12 个像素与中心像素的像素差绝对值大于设定的阈值，则P点是一个特征点。否则，P 点不是一个特征点。

步骤 4　遍历并检查所有像素：最后，遍历选择区域内的每一个像素点，对每个像素执行上述 3 个步骤。基于自适应阈值的特征提取算法流程如图 3 所示。算法 1 为自适应阈值算法流程。

算法　自适应阈值算法。

输入： 图像 images。

输出： 特征点集合 keypoints。

1.计算自适应阈值：$\lambda=\alpha\ln C+\beta$

2.以点 P 为圆心选择 16 个点

for P in image do

if ($|P_1-P|<\lambda$) or ($|P_9-P|<\lambda$) then

P 不是特征点

break

end if

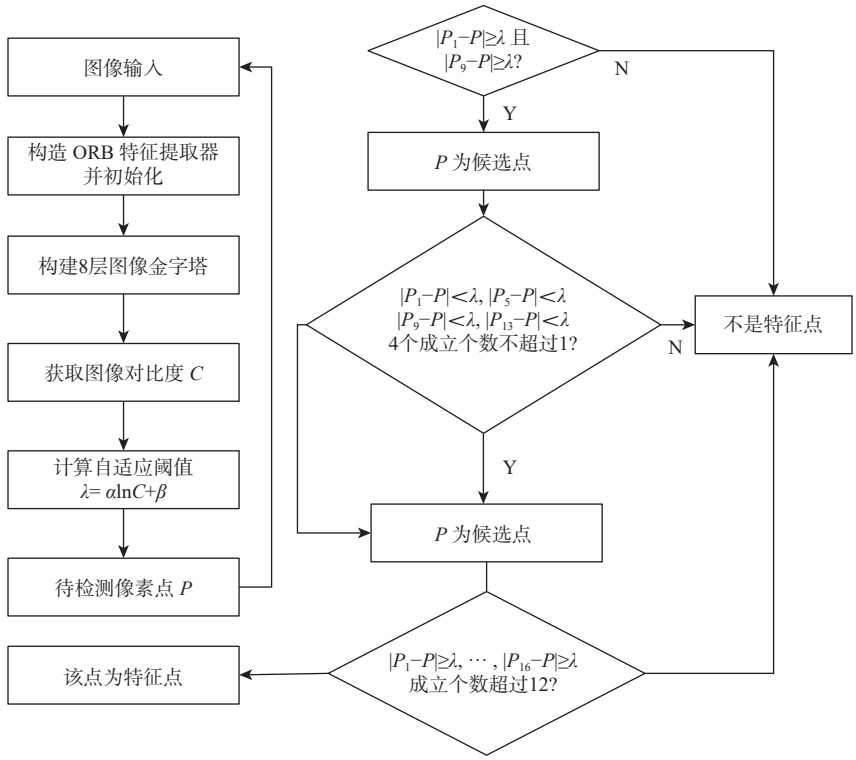


图 3 基于自适应阈值的特征点提取算法流程

Fig. 3 Based on adaptive threshold feature point extraction algorithm flow chat

if 满足$|P-P_i|\geqslant\lambda, i=1,5,9,13$的个数大于 1 then

P 不是特征点

break

end if

if 有 12 个连续点满足$|P-P_j|\geqslant\lambda, j=1,\cdots,16$ then

P 是特征点

else

break

end if

end for

1.3 VSLAM 系统的速度优化

本文系统中的字典和词袋模型都是基于词袋库 DBoW2[27]。在原算法中预先放置的词袋格式为文本，文件体积大，加载读取时间过长，对 SLAM 系统的运行效率具有较大的影响。因此，本文将文本格式的词袋转化为二进制词袋文件，并修改相应代码重新编译。相较于文本格式，二进制词袋文件不仅能确保内容不易被修改或损坏，而且文件体积更小且存在进一步压缩的可能。更为重要的是，二进制词袋文件更契合计算机的读取方式，从而减少了原有文本格式转化和数据处理的时间。同时，二进制词袋不仅能够在 VSLAM 系统的初始化阶段加快地图和词袋模型加载时间，而且也能够加速回环检测时匹配检测速度。视觉词袋模型流程如图 4 所示。

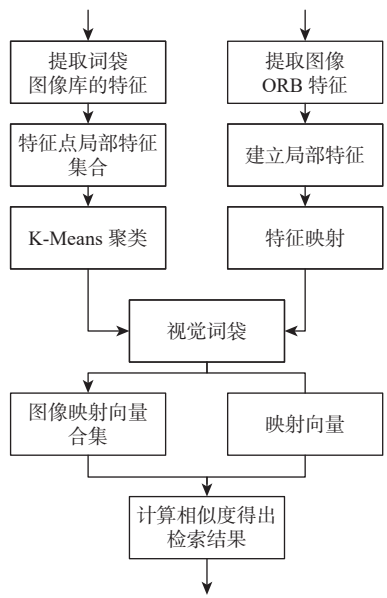


图 4 视觉词袋模型流程

Fig. 4 Visual bags of word structure diagram

局部建图线程主要是对关键帧进行处理，完成地图点的剔除和创建，与此同时对地图进行局部最优处理，最后将关键帧加入闭环检测队列。在这个过程中耗时最长的为局部光束法平差（bundle adjustment, BA）优化，其主要是对共视图中的关键帧进行局部求解最优。在回环检测线程中，系统对整体的关键帧和地图点进行全局 BA 优化。因此，删减局部 BA 部分的优化策略能够减少时间消耗，并且对算法精度的影响较小。优化后的局部地图构建线程如图 5 所示，其中虚线部分表示已被删减的内容。

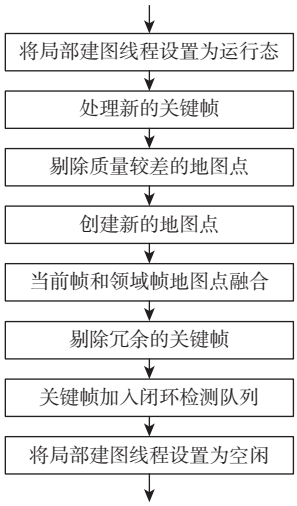


图 5　优化后的局部地图构建线程流程

Fig. 5　Optimized local mapping flow chart

2　实　验

2.1　数据集与实验环境

本文实验中使用的是 TUM 公开数据集[28]。TUM RGB-D 数据集是由 Microsoft Kinect 摄像头与深度传感器所记录的一系列多样化场景序列组成，包含动态场景、3D 物体重建、机器人 SLAM、结构与低纹理等众多应用和研究场景。该数据集除了提供包含深度信息的彩色图片序列外，还包括图片序列对应的时间戳和外部设备捕捉到的相机位姿，能够为评估算法的准确度提供很好的依据。本文实验采用 TUM 数据集中的 8 组序列，包括 4 个低动态数据序列 fr3_sitting 和 4 个高动态数据序列 fr3_walking。

在对实验结果进行定量评估时，本文使用 SLAM 领域通用的评价标准，即相对位姿误差（relative pose error, RPE）和绝对轨迹误差（absolute trajectory error, ATE）[28]。在实验中，分别计算了动态场景下每个数据集 ATE 和 RPE 的均方根误差（root mean squared error, RMSE）。文中相对提升率 η 的计算公式为

$$\eta = \frac{E_0 - E_M}{E_0} \times 100\% \tag{8}$$

实验中 VSLAM 部分均采用 C++开发，深度学习部分为 python 语言，其他实验环境参数如表 1 所示。

2.2　速度优化实验

二进制词袋和原有词袋占用空间和时间效率的对比如表 2 所示。从表中可以得出，在对原有的词袋模型进行优化和格式转换后，无论是储存空间的占用还是模型的加载时间都优于原有词袋模型。这主要因为二进制词袋在被计算设备读取或者查找时，不需要再进行格式转换和数据处理，从而提升了时间效率。

表 1　实验环境

Table 1　Experimental environment

配置名称	配置情况
处理器	Intel(R) Core(TM) i9-10900x CPU@3.70 GHz
显卡	NVIDIA GeForce RTX 3080
操作系统	Ubuntu 18.04
深度学习框架	Keras 2.0.9、TensorFlow-gpu 1.14.0

表 2　不同词袋性能对比

Table 2　Performance comparison of different BOW

性能	词袋空间大小/MB	平均加载时间/ms
原有词袋	145.3	8 289.01
二进制词袋	42.3	267.92

注：二进制词袋比原有词袋在词袋空间和平均加载时间分别提升了 70.89%、96.77%。

在对系统进行优化前，首先利用 C++标准库的时钟函数对跟踪线程和局部地图构建线程的时间消耗进行统计，最终统计结果如表 3 所示。从表中可以看出，局部地图构建线程平均耗时为 609.71 ms，其中局部 BA 优化部分耗时最长，达到 458.52 ms。不难发现，对局部建图线程进行优化是较为高效的方式。通过对局部 BA 部分的删减，能够将局部地图构建部分的时间缩短为原来的四分之一。

表 3　不同线程各部分耗时对比

Table 3　Time-consuming comparison of various parts of different threads

线程	模块	用时/ms	标准差/ms
跟踪线程	特征提取	17.78	3.85
	位姿估计	2.87	1.31
	整体耗时	33.22	13.99
局部地图构建线程	关键帧插入	17.96	8.11
	地图点创建	91.17	30.59
	局部BA优化	458.52	319.41
	总体耗时	609.71	368.44

为了进一步研究实验的效果，本文统计了各个算法在跟踪线程中处理单帧图片的用时，实验结果如表 4 所示。将 ORB-SLAM2 算法与 YOLOv3[29] 和 YOLOv4[30] 算法结合，形成了 YOLOv3-SLAM 和 YOLOv4-SLAM 系统。从表中可以看出，基于 YOLO 目标检测网络的语义 SLAM 系统在运行效率上要明显高于基于像素级实例分割 SLAM 系统，避免了复杂度高和计算量大的图像分割。本文算法与 YOLOv3-SLAM 相比，处理单帧图片时间平均减少了 33.44%，与 YOLOv4-SLAM 相比平均减少了 36.35%。这是因为本文算法采用了高效的 YOLOv7-tiny 网

表4 跟踪线程处理单帧图片用时对比

Table 4 Time-consuming comparison of tracking thread processing single frame

数据集名称[28]	不同模型处理单帧图片耗时/ms			
	YOLOv3-SLAM[29]	YOLOv4-SLAM[30]	DynaSLAM[17]	本文算法
fr3_s_static	55.96	55.54	1 444.07	25.57
fr3_s_xyz	61.36	61.90	1 616.43	42.36
fr3_s_half	70.43	66.75	1 534.07	55.78
fr3_s_rpy	63.69	58.57	1 491.33	41.89
fr3_w_static	83.42	82.97	1 676.27	62.32
fr3_w_xyz	89.47	87.60	1 718.56	69.37
fr3_w_half	95.11	90.74	1 683.11	50.36
fr3_w_rpy	82.57	72.72	1 519.81	54.78

络作为动态特征检测网络，以及对VSLAM系统进行了优化。在局部建图线程中优化处理关键帧的策略以提升计算效率，并采用二进制词袋模型对特征描述子进行压缩编码，显著加速特征点匹配与地图数据的加载过程。同时，基于自适应阈值的特征点提取算法能在对比度高、纹理特征丰富的场景下少提取一定数量的特征点进而减少计算量提高效率。由此可以看出本文算法具有更好的运行效率，更能满足动态场景下VSLAM系统对于实时性的需求。

2.3 自适应阈值实验

基于自适应阈值的特征点提取算法在曝光不足和过曝情况下的提取效果如图6所示。从图中能够看出，在不同的环境光照下基于阈值自适应的提取算法均能较多地提取图像中的特征点。

为了深入研究自适应阈值的效果，进一步将实验分为2组：一组增加了自适应阈值算法的模块，另一组则没有增加该模块。2组实验在其余模块均进行了相应的优化和改进，以确保唯一的变量为自适应阈值算法的引入与否。同时，对2个动态数据集进行了对比度和亮度的调整，并使用绝对轨迹误差的RMSE值作为评价指标。从表5和表6中可以看出，在不同的环境下，加入自适应阈值部分的算法有着更低的绝对轨迹误差。

(a) 原算法特征点提取效果 (b) 原算法特征点提取效果

(c) 改进后特征点提取效果 (d) 改进后特征点提取效果

图6 改进前后特征点提取效果对比图

Fig. 6 Comparison of feature point extraction effect before and after improvement

表5 对比度降低和亮度减少的定位效果

Table 5 Positioning effect of contrast reduction and brightness reduction m

数据集名称	未加入阈值法	加入阈值
fr3_walking_static	0.184 179	0.179 079
fr3_walking_xyz	0.023 520	0.021 872

表6 对比度降低和亮度增加的定位效果

Table 6 Positioning effect of contrast reduction and brightness increase m

数据集名称	未加入阈值	加入阈值
fr3_walking_static	0.209 476	0.188 325
fr3_walking_xyz	0.028 880	0.023 741

2.4 动态特征点剔除的精度实验

在8个包含高动态和低动态场景的数据集序列上进行了综合比较实验，包含ORB-SLAM2、YOLOv3-SLAM、YOLOv4-SLAM、DynaSLAM以及本文算法。评估结果主要基于绝对轨迹误差和相对位姿误差，并通过计算RMSE值进行量化分析。实验结果如表7和表8所示。从实验结果来看，在低动态环境下，由于物体运动速度相对较慢，传感器噪声较低，ORB-SLAM2算法已经能够较好地跟踪并定位相机位姿。而在高动态环境中，本文算法的绝对轨迹误差和相对位姿误差比ORB-SLAM2算法有显著的降低。

与其他动态SLAM算法相比，本文算法比以YOLOv3和YOLOv4为主的轻量化模型的定位误差更低。与DynaSLAM相比，在不同的数据集下均有最小的定位误差。在实际应用中，系统一般需要考虑准确性和实时性两方面。由于本文算法使用了轻量级的目标检测网络YOLOv7-tiny，在需要快速响应或资源有限的场景中，既可以保证准确性的同时，又可以保证实时性。综上，本文算法性能更优。

为了更进一步直观地展示本文算法的效果，本文展示了在不同动态环境中轨迹图的表现。图7展示了在高动态场景下，ORB-SLAM2算法、Dyna-SLAM与本文算法的估计轨迹与误差分布。其中，真实轨迹以黑色线条表示，估计轨迹用蓝色线条表示，红色线条则代表真实轨迹与估计轨迹之间的差异。仔细观察可见，在高动态环境中，ORB-SLAM2算法的估计轨迹和真实轨迹存在较大的差距，而本

表 7　绝对轨迹误差的 RMSE 对比

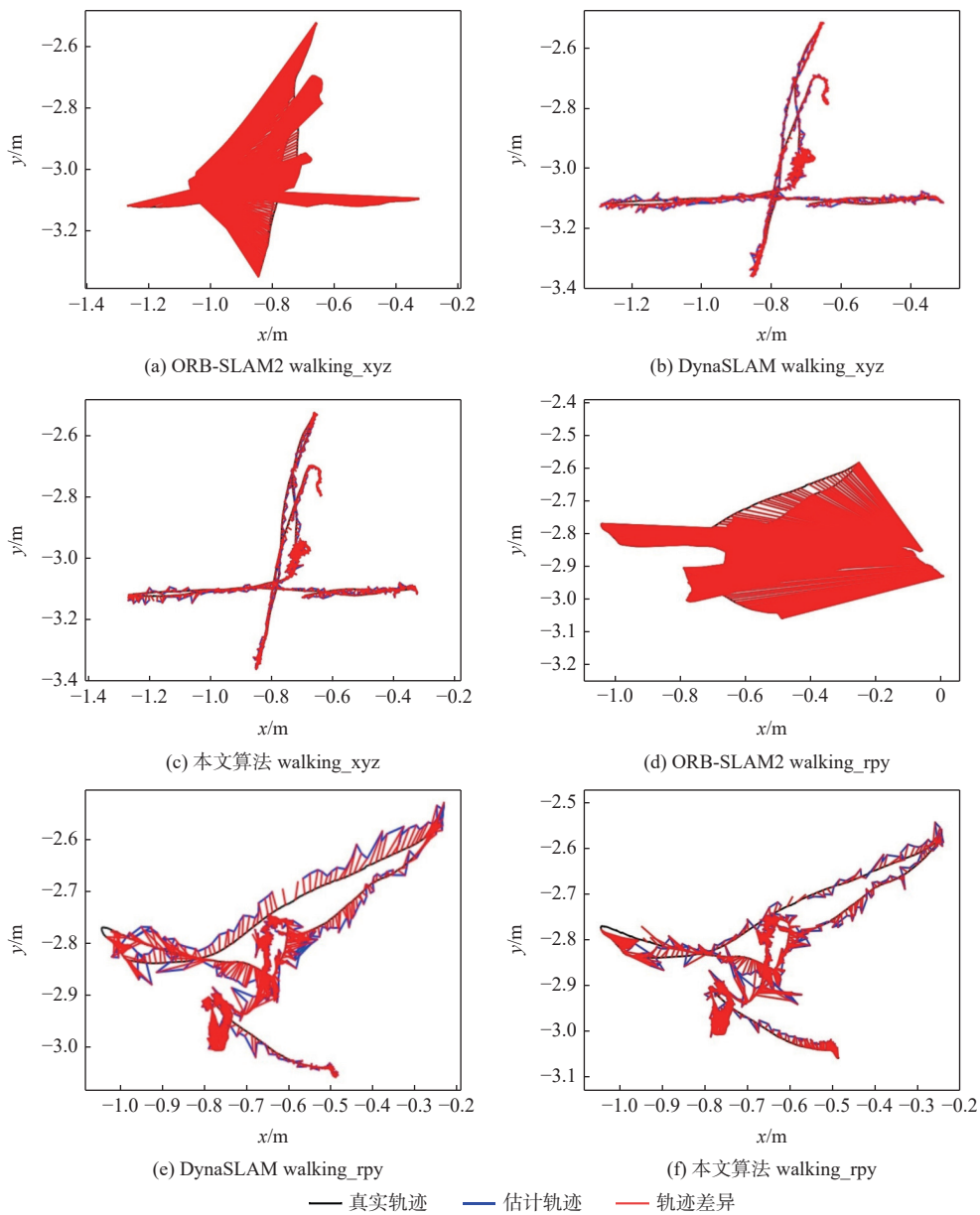
Table 7　RMSE comparison of absolute trajectory error

数据集名称	ORB-SLAM2[8]	YOLOv3-SLAM[29]	YOLOv4-SLAM[30]	DynaSLAM[17]	本文算法
fr3_sitting_static	0.149 578	0.357 965	0.343 385	0.078 521	0.210 825
fr3_sitting_xyz	0.017 513	0.020 901	0.023 578	0.022 394	0.017 827
fr3_sitting_halfsphere	0.032 686	0.029 224	0.025 969	0.025 076	0.025 955
fr3_sitting_rpy	0.144 598	0.276 151	0.299 480	0.253 892	0.328 201
fr3_walking_static	2.757 299	0.261 943	0.220 086	0.124 637	0.136 923
fr3_walking_xyz	1.440 868	0.017 448	0.019 748	0.020 895	0.017 363
fr3_walking_halfsphere	0.977 844	0.041 687	0.052 667	0.030 216	0.033 667
fr3_walking_rpy	2.283 019	0.101 932	0.205 203	0.079 428	0.076 706

表 8　相对位姿误差的 RMSE 的对比

Table 8　RMSE comparison of relative pose error

数据集名称	ORB-SLAM2[8]	YOLOv3-SLAM[29]	YOLOv4-SLAM[30]	DynaSLAM[17]	本文算法
fr3_sitting_static	0.005 814	0.006 678	0.006 538	0.006 379	0.006 153
fr3_sitting_xyz	0.011 067	0.012 211	0.012 060	0.012 741	0.011 950
fr3_sitting_halfsphere	0.011 109	0.028 343	0.030 124	0.018 423	0.025 955
fr3_sitting_rpy	0.016 804	0.016 104	0.016 180	0.020 151	0.016 260
fr3_walking_static	0.025 154	0.009 937	0.009 911	0.008 539	0.009 858
fr3_walking_xyz	0.032 623	0.015 785	0.015 911	0.014 958	0.015 203
fr3_walking_halfsphere	0.068 258	0.016 998	0.017 734	0.015 907	0.016 233
fr3_walking_rpy	0.029 868	0.022 164	0.022 131	0.024 993	0.022 125



(a) ORB-SLAM2 walking_xyz　(b) DynaSLAM walking_xyz

(c) 本文算法 walking_xyz　(d) ORB-SLAM2 walking_rpy

(e) DynaSLAM walking_rpy　(f) 本文算法 walking_rpy

图 7　高动态场景下估计轨迹和真实轨迹对比

Fig. 7　Comparison of real and estimated trajectories in high dynamic scenes

文算法和 DynaSLAM 算法都能够较为精确地估计和跟踪物体的运动轨迹。在图 8 中，可以观察到在低动态场景下，3 种算法的相机位姿估计轨迹与真实轨迹大致相符。综上，改进算法针对高动态场景下的优化更为显著，而在低动态场景中，其提升效果并不明显。

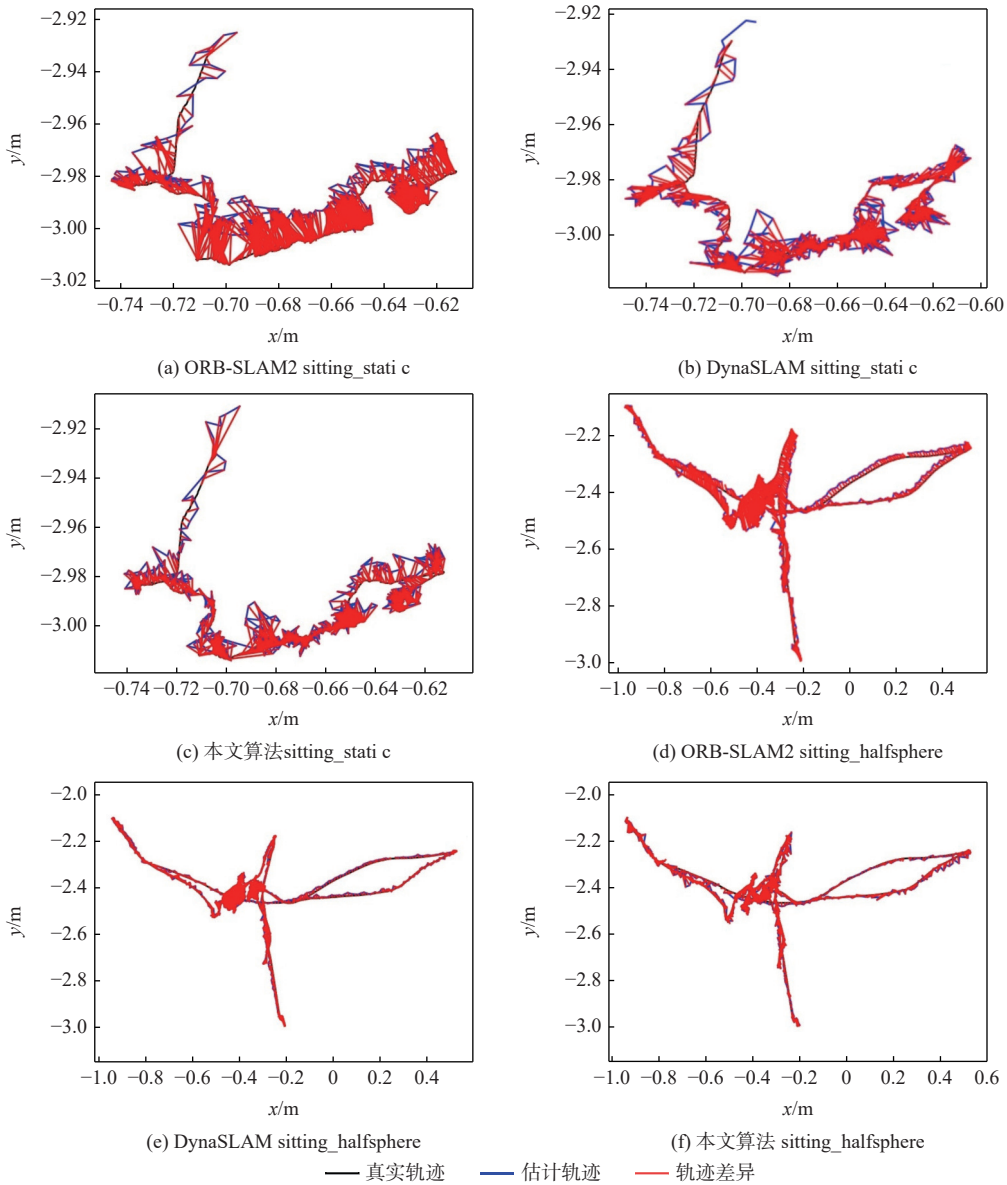


图 8 低动态场景下估计轨迹和真实轨迹对比

Fig. 8 Comparison of real and estimated trajectories in low dynamic scene

2.5 讨 论

在室内动态场景中，基于特征的 ORB-SLAM2 算法依赖于连续帧之间的特征匹配，而动态物体会影响相机姿态估计和地图生成的可靠性。同时，传统的固定阈值方法在光照变化较大时容易产生误检测或漏检测。针对以上问题，本文算法根据图像对比度自适应地调整阈值，在曝光不足和过曝的室内环境下提取效果显著提升，提高了动态物体检测的鲁棒性。相较于 YOLOv3-SLAM 和 YOLOv4-SLAM 算法，本文算法处理单帧图片的时间平均减少了 33.44% 和 36.35%。本文更高的运行效率可以满足动态场景下系统对于实时性的需求。但是，在室内极端光照条件下或者低动态场景中仍然存在很大的进步空间，未来可以结合图像增强算法或使用多传感器信息进行光照估计，以提供更准确的光照补偿，进而改善动态物体检测的性能。

3 结 论

本文研究动态场景下的 VSLAM 算法，旨在充分利用图像中的高层次语义信息，提高在动态环境中的 SLAM 算法的精度和鲁棒性。

1）在处理单帧图片耗时上，与 YOLOv3-SLAM 和 YOLOv4-SLAM 相比，本文算法单帧处理时间分别平均降低了 33.4% 和 36.4%，显著提升了系统运

行效率。

2）定位精度方面，在高动态环境下，本文算法在动态特征点剔除精度上优于 ORB-SLAM2，其绝对轨迹误差和相对位姿误差均显著降低。相较于基于 YOLOv3/YOLOv4 轻量化模型的动 SLAM 方法以及 DynaSLAM，本文算法在不同数据集下展现出更低的定位误差。得益于采用的轻量级目标检测网络 YOLOv7-tiny，本文算法在保证动态场景下定位准确性的同时，有效满足了系统实时性要求，综合性能更优。

未来本课题将会从以下方面开展研究：一方面，在之后的研究中可以通过融合激光雷达、IMU 传感器、轮速计等多种传感器信号，采用多模态传感器融合的方法来增强算法的鲁棒性和适用性。另一方面，轻量化语义 SLAM 模型还需要进一步优化，以保证在算力受限制的嵌入式平台上保持实时性。

致谢　感谢深圳市云视机器人有限公司对本工作在资金和实验设备上的大力支持 (HD-KYH-2 021 307)。

参考文献（References）

[1] CHEN W F, SHANG G T, JI A H, et al. An overview on visual SLAM: from tradition to semantic[J]. Remote Sensing, 2022, 14(13): 3010.

[2] AI Y B, RUI T, LU M, et al. DDL-SLAM: a robust RGB-D SLAM in dynamic environments combined with deep learning[J]. IEEE Access, 2020, 8: 162335-162342.

[3] YU L J, YANG E F, YANG B Y. AFE-ORB-SLAM: robust monocular VSLAM based on adaptive FAST threshold and image enhancement for complex lighting environments[J]. Journal of Intelligent & Robotic Systems, 2022, 105(2): 26.

[4] NEWCOMBE R A, LOVEGROVE S J, DAVISON A J. DTAM: dense tracking and mapping in real-time[C]//Proceedings of the 2011 International Conference on Computer Vision. Piscataway: IEEE Press, 2011: 2320-2327.

[5] FORSTER C, PIZZOLI M, SCARAMUZZA D. SVO: fast semi-direct monocular visual odometry[C]//Proceedings of the 2014 IEEE International Conference on Robotics and Automation. Piscataway: IEEE Press, 2014: 15-22.

[6] DAVISON A J, REID I D, MOLTON N D, et al. MonoSLAM: real-time single camera SLAM[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(6): 1052-1067.

[7] KLEIN G, MURRAY D. Parallel tracking and mapping for small AR workspaces[C]//Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. Piscataway: IEEE Press, 2007: 225-234.

[8] MUR-ARTAL R, TARDÓS J D. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras[J]. IEEE Transactions on Robotics, 2017, 33(5): 1255-1262.

[9] CAMPOS C, ELVIRA R, RODRÍGUEZ J J G, et al. ORB-SLAM3: an accurate open-source library for visual, visual–inertial, and multimap SLAM[J]. IEEE Transactions on Robotics, 2021, 37(6): 1874-1890.

[10] SHEN X Q, CHEN L H, HU Z H, et al. A closed-loop detection algorithm for online updating of bag-of-words model[C]// Proceedings of the 2023 9th International Conference on Computing and Data Engineering. New York: ACM, 2023: 34-40.

[11] CHENG Y H, WANG J. A motion image detection method based on the inter-frame difference method[J]. Applied Mechanics and Materials, 2014, 490-491: 1283-1286.

[12] CUTLER R, DAVIS L S. Robust real-time periodic motion detection, analysis, and applications[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 22(8): 781-796.

[13] KLAPPSTEIN J, VAUDREY T, RABE C, et al. Moving object segmentation using optical flow and depth information[C]//Proceedings of the Advances in Image and Video Technology. Berlin: Springer, 2009: 611-623.

[14] DEROME M, PLYER A, SANFOURCHE M, et al. Moving object detection in real-time using stereo from a mobile platform[J]. Unmanned Systems, 2015, 3(4): 253-266.

[15] YU C, LIU Z X, LIU X J, et al. DS-SLAM: a semantic visual SLAM towards dynamic environments[C]//Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway: IEEE Press, 2018: 1168-1174.

[16] BADRINARAYANAN V, KENDALL A, CIPOLLA R. SegNet: a deep convolutional encoder-decoder architecture for image segmentation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(12): 2481-2495.

[17] BESCOS B, FÁCIL J M, CIVERA J, et al. DynaSLAM: tracking, mapping, and inpainting in dynamic scenes[J]. IEEE Robotics and Automation Letters, 2018, 3(4): 4076-4083.

[18] FU Y X, HAN B, HU Z H, et al. CBAM-SLAM: a semantic SLAM based on attention module in dynamic environment[C]//Proceedings of the 2022 6th Asian Conference on Artificial Intelligence Technology. Piscataway: IEEE Press, 2022: 1-6.

[19] CAI D P, HU Z H, LI R Q, et al. AGAM-SLAM: an adaptive dynamic scene semantic SLAM method based on GAM[C]//Proceedings of the Advanced Intelligent Computing Technology and Applications. Berlin: Springer, 2023: 27-39.

[20] HE K M, GKIOXARI G, DOLLÁR P, et al. Mask R-CNN[C]//Proceedings of the 2017 IEEE International Conference on Computer Vision. Piscataway: IEEE Press, 2017: 2980-2988.

[21] QI H, HU Z H, XIANG Y F, et al. ATY-SLAM: a visual semantic SLAM for dynamic indoor environments[C]//Proceedings of the Advanced Intelligent Computing Technology and Applications. Berlin: Springer, 2023: 3-14.

[22] CHANG Z Y, WU H L, SUN Y L, et al. RGB-D visual SLAM based on Yolov4-tiny in indoor dynamic environment[J]. Micromachines, 2022, 13(2): 230.

[23] WANG C Y, BOCHKOVSKIY A, LIAO H M. YOLOv7: trainable bag-of-freebies sets new state-of-the-art for real-time object detectors[C]//Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press,

2023: 7464-7475.

[24] GONZALES R C, WINTZ P. Digital image processing[M]. Reading: Addison-Wesley Longman Publishing Co., Inc., 1987.

[25] BURRI M, NIKOLIC J, GOHL P, et al. The EuRoC micro aerial vehicle datasets[J]. The International Journal of Robotics Research, 2016, 35(10): 1157-1163.

[26] VISWANATHAN D G. Features from accelerated segment test (fast)[C]// Proceedings of the 10th Workshop on Image Analysis for Multimedia Interactive Services. Piscataway: IEEE Press, 2009: 6-8.

[27] QADER W A, AMEEN M M, AHMED B I. An overview of bag of Words;Importance, implementation, applications, and challenges[C]// Proceedings of the 2019 International Engineering Conference. Piscataway: IEEE Press, 2019: 200-204.

[28] STURM J, ENGELHARD N, ENDRES F, et al. A benchmark for the evaluation of RGB-D SLAM systems[C]//Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway: IEEE Press, 2012: 573-580.

[29] REDMON J, FARHADI A. YOLOv3: an incremental improvement [EB/OL]. (2018-04-08)[2023-07-25]. https://doi.org/10.48550/arXiv.1804.02767.

[30] BOCHKOVSKIY A, WANG C Y, LIAO H M. YOLOv4: optimal speed and accuracy of object detection[EB/OL]. (2020-04-23)[2023-07-25]. https://doi.org/10.48550/arXiv.2004.10934.

A lightweight semantic VSLAM approach based on adaptive thresholding and speed optimization

QI Hao[1], FU Yuexin[1], HU Zhuhua[1,*], WU Jiaqi[1], ZHAO Yaochi[2]

(1. School of Information and Communication Engineering, Hainan University, Haikou 570228, China;
2. School of Cyberspace Security (School of Cryptology), Hainan University, Haikou 570228, China)

Abstract: Visual simultaneous localization and mapping (VSLAM) is a technology that utilizes visual and other sensory sensors to acquire information about unknown environments. It is widely applied in fields such as autonomous driving, robotics, augmented reality, and more. However, pixel-level semantic segmentation of dynamic objects entails high computing costs for indoor visual SLAM, and variations in lighting make dynamic items appear more difficult to see, potentially causing occlusions or confusion with the static surroundings. To address these challenges, a lightweight semantic VSLAM model is proposed, which is based on adaptive thresholding and velocity optimization. Initially, a lightweight one-stage object detection network, YOLOv7-tiny, is utilized in conjunction with the optical flow algorithm to effectively detect dynamic regions within images and filter out unstable feature points. Additionally, the feature point extraction algorithm dynamically adjusts the threshold based on the contrast information of the input images. Moreover, the combination of a binary bag-of-words method with a simplified optimization technique for local mapping threads improves the system's loading and matching speed in indoor dynamic scenarios. Experimental results show that the proposed algorithm can effectively eliminate dynamic feature points in indoor high-dynamic scenes, improving the positioning accuracy of the camera. The average processing speed reaches 19.8 frames per second (FPS), meeting real-time requirements in practical scenarios.

Keywords: visual simultaneous localization and mapping; dynamic scenes; YOLOv7-tiny; adaptive threshold; feature points

Received: 2023-08-28; **Accepted**: 2024-03-22; **Published Online**: 2024-05-08 10:50

URL: link.cnki.net/urlid/11.2625.V.20240507.1513.002

Foundation items: National Key Research and Development Program of China (2022YFD2400504); National Natural Science Foundation of China (62161010); Key Research and Development Project of Hainan Province (ZDYF2022SHFZ039,ZDYF2022GXJS348)

* **Corresponding author**. E-mail: eagler_hu@hainanu.edu.cn