分类号：TP273　　单位代码：10363

密　级：公开　　学　　号：2210330147

题　目 基于深度学习的室内动态场景 SLAM 算法研究

英文并列

题　目 RESEARCH ON SLAM ALGORITHM FOR INDOOR DYNAMIC SCENES BASED ON DEEP LEARNING

学生姓名：张开平

指导教师：黄宜庆（教授）

专　　业：电子信息

研究方向：人工智能与系统集成

论文答辩日期：

安徽工程大学学位论文原创性声明

本人郑重声明：我恪守学术道德，崇尚严谨学风。所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已明确注明和引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品及成果的内容。论文为本人亲自撰写，我对所写的内容负责，并完全意识到本声明的法律后果由本人承担。

学位论文作者签名：

日期:　　年　　月　　日

安徽工程大学学位论文版权使用授权书

学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅或借阅。本人授权安徽工程大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保密 □，在 ____ 年解密后适用本版权书。

本学位论文属于

不保密 □。

学位论文作者签名：  　　指导教师签名： 

日期：　　日期：

基于深度学习的室内动态场景 SLAM 算法研究

摘 要

实时定位与建图（Simultaneous Localization and Mapping, SLAM）一直是移动机器人领域的核心技术。基于视觉传感器的 SLAM 算法通常被称为视觉 SLAM，视觉传感器可以获得周围环境丰富的特征信息，例如颜色、纹理等信息。传统的视觉 SLAM 都是以静态环境为理想条件而进行研究的，但是在实际应用场景中，动态对象的出现是无法避免的，这些动态物体会造成错误的数据关联，最终会致全局估计轨迹的错误。本文研究了视觉 SLAM 在动态场景下定位精度下降的问题，主要研究内容如下：

首先，对于 SLAM 在实际应用中的实时性需求，本文首先分析了目前主流的神经网络模型轻量化方法的优缺点，对于结合深度学习的动态环境 SLAM 算法无法满足 SLAM 系统的实时性需求问题，本文在 YOLOv5s 目标检测网络的基础上参考 GSCONV 模块对其进行轻量化改进并嵌入 SLAM 系统以获得场景语义信息，同时也可以满足 SLAM 系统的实时性要求。

其次，针对目标检测网络获取的场景语义信息冗余的问题，本文设计一种深度信息随机采样一致性(Depth-RANSAC)算法，将语义先验信息与场景深度信息相关联，对待检测区域的动态特征信息进行过滤，避免了特征信息误剔除，保留了静态的背景信息，系统定

位结果更加精确。对于场景中少量潜在动态物体，本文研究了多视图几何约束原理，在此基础上对剩余特征信息进行二次筛选，去除潜在动态物体对于系统定位精度的影响。

最后，对于上述算法改进，本文在公开数据集 TUM 上进行有效性验证。首先对于目标检测网络的轻量化改进进行验证，与原算法在检测速度与精度方面进行对比试验；其次对于深度信息采样一致性算法结合多视图几何约束的方法进行动态特征信息滤除试验；随后在 TUM 中的动态序列上对本文算法进行定位精度评估，经过分析证明本文算法相比于原算法，在动态场景下的定位精度得到提升；最后将本文算法与 ORB-SLAM2 在动态数据集上进行稠密建图结果对比，表明本文算法可以构建静态的环境点云地图。

关键词：实时定位与建图，目标检测，深度学习，多视图几何约束，三维稠密建图

RESEARCH ON SLAM ALGORITHM FOR INDOOR DYNAMIC SCENES BASED ON DEEP LEARNING

ABSTRACT

Simultaneous Localisation and Mapping (SLAM) has been a core technology in the field of mobile robotics. SLAM algorithms based on visual sensors are often referred to as visual SLAM, and visual sensors can obtain rich feature information of the surrounding environment, such as colour, texture and other information. Traditionally, vision SLAM has been studied with static environment as the ideal condition, but in real application scenarios, the appearance of dynamic objects is unavoidable, and these dynamic objects can cause erroneous data correlation, which ultimately leads to the error of global estimation trajectory. In this paper, we study the problem of degradation of localisation accuracy of visual SLAM in dynamic scenes, and the main research contents are as follows:

First, for the real-time demand of SLAM in practical applications, this paper first analyses the advantages and disadvantages of the current mainstream neural network model lightweight methods, for the dynamic environment SLAM algorithms combined with deep learning can not meet

the real-time demand of the SLAM system, this paper on the basis of YOLOv5s target detection network with reference to the GSCONV module to lightweight its improvement and embedded in the SLAM system to obtain the scene semantic information, and at the same time can also meet the real-time requirements of the SLAM system.

Secondly, for the problem of redundancy of scene semantic information obtained by the target detection network, this paper designs a Depth-RANSAC algorithm, which correlates the semantic a priori information with the scene depth information, filters the dynamic feature information in the area to be detected, avoids false rejection of the feature information, and retains the static background information, so that the system localisation results are more accurate. For a small number of potential dynamic objects in the scene, this paper investigates the principle of multi-view geometric constraints, on the basis of which the remaining feature information is filtered twice to remove the influence of potential dynamic objects on the positioning accuracy of the system.

Finally, for the above algorithm improvement, this paper verifies the effectiveness on the public dataset TUM. Firstly, the lightweight improvement of the target detection network is verified and compared with the original algorithm in terms of detection speed and accuracy; secondly, the depth information sampling consistency algorithm combined with multi-view geometric constraints is used to perform dynamic feature

information filtering tests; subsequently, the algorithm of this paper is evaluated for its localisation accuracy on dynamic sequences in TUM, and it is proved that the algorithm of this paper has improved its localisation accuracy in dynamic scenes as compared with the original algorithm. After analysis, it is proved that the positioning accuracy of this algorithm is improved in the dynamic scene compared with the original algorithm; finally, the results of comparing this algorithm with ORB-SLAM2 on the dynamic dataset for densely constructing maps show that the algorithm in this paper can construct the static point cloud map of the environment.

KEY WORDS: Simultaneous Localization and Mapping, Target Detection, Deep Learning, Multi-view Geometry Constraints，3D Dense Mapping

目录

插图清单

插表清单

第1章 绪论

1.1 研究的背景和意义

近年来随着人工智能、机算机视觉、物联网等技术的发展，移动机器人技术[1]迎来了高速发展，在人类生活中扮演着越来越重要的角色。移动机器人技术是集计算机科学、电子信息、机械工程于一体的智能设备系统，它体现了一个国家的工业与科学技术发展的水平。移动机器人可以代替人类在复杂环境下进行高风险活动，如图1-1所示。因此，如何使移动机器人在未知的复杂环境下自主获取当前的环境信息成为了近年来学术界的热点研究话题之一。要解决这个问题首先需要面对三个问题：首先是机器人得知道自己目前处在什么位置，其次是要获取周边的环境信息，最后是建立起周边环境的地图模型[2]。

(a)月球探索机器人　　(b)水下机器人

图 1-1 移动机器人

同时定位与建图(Simultaneous Localization and Mapping，SLAM)技术是指在没有先验信息的条件下，移动机器人通过自身所携带的传感设备，通过自身所在位置的移动来获取环境信息，建立周围环境的地图模型[3]。根据携带传感器种类的区别，SLAM 可分为激光 SLAM 和视觉 SLAM(visual SLAM,VSLAM)。在早期研究中，研究者们主要通过激光雷达来获取环境信息。激光雷达可以发射脉冲信号到物体表面，通过分析脉冲信号返回的时间来获取物体的距离信息，从而创建周围环境的 3D 点云地图，最终得到机器人在环境中的定位信息[4]。这

种方法成熟稳定，精确度高，但是由于激光雷达传感器的价格比较昂贵，故很难做到普遍的商业化生产。由于相机的使用成本低，并且随着计算机视觉技术的发展，相机可以获取到丰富的环境纹理信息，VSLAM 成为了 SLAM 领域中的研究热点[5]。根据视觉传感器的种类不同，相机可分为单目相机、双目相机、RGB-D 相机以及事件相机等不同类型。

VSLAM 虽然在近年来得到了极大的发展，但是仍然在一些场景下存在限制。许多优秀的开源 VSLAM 算法例如 ORB-SLAM2，它的理想工作场景是在低动态环境下，当场景中出现少量动态信息时，会通过随机采样一致性算法[6]将这些信息作为外点剔除。但是当动态物体占据了场景中大部分的时候，随机采样一致性算法就会将这些特征信息作为内点保存下来，导致特征信息误关联等问题，从而导致系统估计的相机运动漂移，定位精度差，跟踪失败等情况。在实际的应用场景下，动态物体的出现是无法避免的，所以必须提高 SLAM 算法在复杂环境下的定位精度。

另一方面，传统的 VSLAM 算法是利用多视图几何的方法来构建出周围的环境地图模型，无法获取环境的高层次信息，例如环境的语义信息，此类信息可以加深机器人对于环境的理解和感知能力。随着人工智能与机器视觉技术的发展，将深度学习加入 VSLAM 算法中，使 SLAM 获取到场景中的语义信息，建立三维语义地图[7]，是目前 SLAM 领域的研究热点。

1.2 国内外研究现状

基于视觉传感器来实现 SLAM 的方法通常被称为视觉 SLAM，与激光 SLAM 相比，使用视觉传感器可以获得更充分的环境特征信息，因此视觉 SLAM 成为近年来热门的研究领域。

1.2.1 传统视觉 SLAM 研究现状

Davison 等[8]在 2007 年实现了首个使用单目相机实时恢复在未知场景中快速运动的 3D 轨迹的算法，MonoSLAM。在此之前的 SLAM 算法研究都是基于传感器运动时采集的图像信息，在离线状态下来恢复移动设备的运动轨迹，无法在实时状态下进行系统的运行。MonoSLAM 在后端使用了扩展卡尔曼滤波器(EKF)[9]来优化传感器采集到的稀疏的特征信息，将传感器当前的状态和路标点

为状态量，更新均值和协方差，实现了相机运动轨迹和环境地图结构的实时概率估计。但是由于 MonoSLAM 获取的是稀疏的特征信息，因此在系统运行中会出现跟踪丢失的问题。此外该算法只有单一的线程，其前端特征信息采集、相机运动轨迹估计、环境地图构造等工作只能顺序执行，故存在计算量大，计算资源浪费等问题。

为了解决上述这些问题，在同一年 Klein 等人提出了并行跟踪与建图(Parallel Tracking and Mapping，PTAM) [10]算法。此算法首次提出并实现了跟踪线程与建图线程的并行化运行，跟踪线程需要对图像信息进行及时响应，而建图线程则不需要同步构建地图，后端优化可以在系统后台挂起，在地图信息更新时再进行线程的同步。这是视觉 SLAM 研究中首次出现前端与后端的概念，往后许多优秀的视觉 SLAM 算法都是受到此种思想的影响开发的。此外 PTAM 还是首个使用光束法平差 (Bundle Adjustment, BA)[11]这类非线性优化作为后端的算法，解决了 EKF 等状态估计稀疏性的问题。此算法还引入了关键帧的概念，因为采集每一帧图像的间隔时间不长，故连续的图像帧可能存在大量共视区域，故把每一帧图像都进行优化会造成大量计算资源的浪费，PTAM 选择其中具有代表性特征信息的图像帧来进行相机运动轨迹的计算与地图的构建。PTAM 也存在应用场景小，跟踪线程易丢失等问题。

西班牙 Zaragoza 大学的 Mur-Arta 等人在 2005 年提出了 ORB-SLAM 系统[12]，这是一个代码构造优秀的视觉 SLAM 系统，非常适合实际工程的移植。它在 PTAM 多线程理念的基础上，首次提出了跟踪、地图构建、回环检测三个线程并行运行的理念，并利用 ORB 特征点[13]来作为提取的环境特征信息。ORB 特征点是基于 FAST 特征点[14]改进而来，利用灰度质心法和图像金字塔对其进行了优化，因此具有方向和尺度不变性。在回环检测中通过词袋模型来进行重定位的判断，最终构建全局同步的信息地图，此系统在没有图像处理器的情况下也可以实现良好的实时运行效果。2017 年，原作者在 ORB-SLAM 的基础上，使系统兼容了双目相机与 RGB-D 相机，提出了 ORB-SLAM2[15]，此算法是特征点法视觉 SLAM 系统的代表之作。ORB-SLAM2 的图优化算法采用的是基于因子图的非线性优化方法，将相机的位姿与地图点位置的误差最小化，提高了地图构建的精确性。Campos 等人在 2021 年提出了 ORB-SLAM3[16]，此系统加入了

对惯性测量单元(Inertial Measurement Unit, IMU)和鱼眼相机的工作模式,实现了单目和双目的视觉惯导系统，并且是首个基于特征的紧耦合 VIO 系统，其精度相对于 ORB-SLAM2 提升了 2 到 5 倍。可以说 ORB-SLAM3 是特征点视觉 SLAM 系统的巅峰之作。

1.2.2 动态环境下视觉 SLAM 算法研究现状

传统的视觉 SLAM 算法的研究都是为静态或低动态场景设计的，在实际应用场景中，动态对象无法避免，这些动态物体会造成错误的数据关联，破坏每一帧图像数据之间的约束关系，最终会致全局估计的错误[17]。动态场景下 SLAM 系统定位精度下降的问题成为近年来研究的热门，目前主要有两种解决方案。

第一类是基于多视图几何的方法。Xu 等人提出了一种新的前端视觉里程计算法[18]，通过结合图像的空间几何信息去除运动物体，并依赖剩余的特征信息来估计相机的位置。Sun 通过相邻两帧图像间匹配特征点的颜色信息得到基本矩阵，根据得到的基本矩阵估计动态特征点，然后通过粒子滤波的方法跟踪动态特征点，实现动态和静态特征信息的分割[19]。Wang 等人利用图像的深度信息进行聚类，通过基本矩阵约束相机运动，剔除不匹配的动态特征点[20]。Li 等人通过移动概率传播模型定位图像中的动态特征点[21]。Fang Y 通过 EFK 结合光流法[22]来检测动态物体，并剔除其动态特征信息。上述基于多视图几何的方法由于约束条件苛刻或数学模型的精度达不到要求，所以并不能准确区分场景中的静态特征与动态特征信息。

第二类则是基于语义信息的方法。由于近年来深度学习技术的发展，此类方法成为热门的研究方向。Zhong 等人使用 SSD(Single Shot MultiBox Detector)来检测动态物体[23]。Zhang 等人将 YOLO(You Only Look Once)[24]网络模型集成到视觉 SLAM 框架中，用于检测场景中的行人、车辆等动态物体，并将这些动态物体框内的特征信息全部去除[25]。由于目标检测网络无法准确获取动态物体的轮廓，将框内的特征信息全部去除会造成有用的特征信息丢失，导致 SLAM 系统精度的下降。Bescos 等人提出了 Dyna-SLAM[26]，将语义分割网络 Mask R-CNN[27]加入到视觉 SLAM 框架中，结合多视图几何确定动态和静态特征信息，并使用关键帧删除动态特征后修复图像背景。Yu 等人提出 DS-SLAM[28]，此方

法使用语义分割网络 SegNet[29]获取场景中的语义信息，并可以使用关键帧构建语义八叉树地图。Cheng 等人提出了 DM-SLAM[30]，此方法结合了 Mask R-CNN、光流和极线约束来判断场景中获取的离群特征点，并将这些离群点作为动态信息剔除。Yuan 在 ORB-SLAM 的基础上利用相邻帧之间匹配的特征点中的几何约束来判断场景中的动态特征，再对场景中进行离线的语义分割来获取语义信息[31]。Fan 提出了 Blitz-SLAM[32]，此系统利用 BlitzNet[33]网络对场景进行语义分割获得场景中物体的 Mask，再通过场景深度信息结合多视图几何约束来对静态与动态信息进行识别。然而，此类方法依赖于语义分割网络的质量，过于庞大的网络提取动态信息的过程耗时太长，不符合 SLAM 系统的实时性要求;而太简单的网络则无法从场景中准确识别动态物体。

1.3 本文研究内容及章节安排

如今比较成熟的开源 SLAM 算法大多是以静态的工作环境为出发点而开发的，而在面对复杂的动态场景时，尤其是动态物体占据场景大部分比例时，这些 SLAM 算法的表现就会大打折扣。目前较为热门的研究方向是结合深度学习以及多视图几何的方法来检测这些动态信息从而剔除，实现动态场景下定位与建图的准确性。而基于深度学习算法的语义分割网络带来的巨大计算量会降低 SLAM 系统工作的实时性，多视图几何方法由于苛刻的条件以及数学模型的精度要求太高而无法准确识别场景中的动态特征，故目前此类方法还不足以达到运用在实际场景下的条件。

针对以上这些问题，本文以开源 SLAM 算法 ORB-SLAM2 为基础，针对室内动态场景下的实时定位与建图精度下降的问题，提出结合深度学习算法以及多视图几何的解决方案。首先，本文在 YOLOV5s 目标检测网络的基础上，对其主体架构进行轻量化改进，在可接受的精度范围内，减少其网络的计算量，提高网络运行的速度及效率，将轻量化处理后的网络嵌入到 ORB-SLAM2 中，以获得场景中的动态物体框以及先验语义信息；其次，本文提出一种结合目标检测网络获取的场景语义信息和RGB-D相机获取的场景深度信息的D-RANSAC算法，这种方法可以准确剔除动态框内落在动态物体上的特征点，而保留框内的静态特征点；最后针对场景中的潜在动态物体，本文使用多视图几何约束的

方法来进行动态特征二次剔除。

本文章节安排如下：

第 1 章为绪论。主要分析了本文的研究背景及动态场景下 SLAM 算法研究意义，随后研究了传统的视觉 SLAM 算法以及动态场景 SLAM 算法的国内外研究现状，最后概述本文的研究工作内容并介绍了章节架构安排。

第 2 章为视觉 SLAM 技术理论及方法。本章分析了视觉 SLAM 的基本数学模型并研究了其每个模块的工作原理，随后研究了 ORB-SLAM2 算法的基本框架和工作流程。

第 3 章为基于 YOLOV5s 目标检测网络的轻量化改进。本章首先对机器学习和卷积神经网络的各类结构进行分析；其次对 YOLOV5s 架构进行阐述，最后针对实际应用需求对其进行轻量化改进,并对轻量化处理的 YOLOV5s 目标检测网络进行动态物体检测实验，与其他目标检测模型进行对比。

第 4 章为室内动态场景下 RGB-D SLAM 算法研究。阐述本文算法整体架构，将第三章所设计的轻量化目标检测网络作为单独线程嵌入 ORB-SLAM2。目标检测网络可以获取到场景中的动态物体框信息，但是将框内所有特征信息全部剔除，会导致当动态物体占据场景大部分比例时，会出现特征数据误匹配的问题。对此本文提出一种结合语义先验信息和场景深度信息的 D-RANSAC 算法，此算法可以准确提出动态框内落在动态物体上的特征信息，有保留了静态背景信息。对于场景中潜在的动态物体，本文使用多视图几何约束方法来进行动态特征二次滤除。最后通过实验证明动态特征信息剔除的有效性。

第 5 章为算法实验与分析。本章首先说明了实验场景以及实验平台；随后对本文构建的针对室内动态场景下的视觉 SLAM 算法进行实验，并与当前主流的动态 SLAM 算法进行对比定位评估实验；最后通过稠密建图的结果来证明本文算法改进工作的有效性。

第2章 视觉 SLAM 基础理论分析

移动机器人要感知周围环境，就必须知道自己当前的位置和方位。因此，人们提出了 SLAM 的概念，它指的是一个不断运动的物体在未知环境下，通过自身携带传感器采集的环境信息，逐渐建立起周围环境模型，并估计自身位置和运动情况的问题解决方案。视觉 SLAM 就是利用各类相机所采集的图像信息来实现上述功能。本章将对视觉 SLAM 的基本架构、各类组成部分以及 ORB-SLAM2 算法的基本架构进行阐述。

2.1 视觉 SLAM 基本框架

目前传统的视觉 SLAM 架构主要由视觉传感器数据采集、前端视觉里程计、后端优化、回环检测和地图构建这五个部分组成。视觉传感器数据采集主要作用是图像数据的采集。前端视觉里程计的作用是粗略计算相邻两帧图像的位姿变换关系，估算出相机当前位姿。后端优化主要用于对不同时刻前端视觉里程计估算的位姿变换关系并消除误差。在回环检测中，会判断机器人是否回到了相同的场景下，并消除之前的模块中累计的噪声。地图构建基于上面四个模块，建立全局一致的三维地图，以满足各类工作任务需求。下面本文将详细阐述上面各个模块的工作原理以及相关数学模型。

2.1.1 视觉传感器数据采集

移动机器人通过自身携带的传感器来获取一些物理量，从而实现定位与地图构建，这就是 SLAM 的实现方法。视觉 SLAM 就是通过相机这类视觉传感器来实现定位与建图的功能。目前应用在 SLAM 领域主流的视觉传感器根据种类的不同，分为单目(Monocular)相机、双目(Stereo)相机和深度相机(RGB-D)三种类型。

单目 SLAM(Monocular SLAM)是指通过单目相机作为传感器采集的环境数据来实现 SLAM 的方法。单目相机因为其原理简单、价格便宜等原因，被广泛运用在视觉 SLAM。但是单目相机的本质就是将三维的场景投影在了二维的相机成像平面上，所以在单目相机中，通过单张图片是无法获取相机与场景中物

体的距离的，并且如果想恢复场景原本的三维结构，必须通过相机的运动才能够构建场景的三维结构以及物体的距离。在一帧图像中，由于近大远小的的原理，场景物体的真实尺度无法确定。由于单目相机的硬件缺陷，单目 SLAM 所估计的运动轨迹与相机的真实运动轨迹之间具有尺度不一致性的问题，所以单目相机存在尺度不确定性[34]。

双目相机本质上就是由两个距离固定的单目相机组成，这个距离被称为基线(Baseline)。通过两个单目相机在不同角度拍摄的场景图片可以模拟单目相机在不同位姿下获取的图片，经过计算可以获取像素点在三维空间中的深度距离。由于不受其他传感器的限制，双目相机的应用场景较为广泛，即可用于室内也可用于室外。双目相机可计算深度的范围受限于基线的长度以及相机的分辨率，并且每个像素点深度的计算需要大量的计算资源，这也导致双目相机若对场景深度进行实时计算，就需要图像处理单元设备的支持。

RGB-D 相机与双目相机获取场景深度信息的方式不同，其向物体主动发送光信号，通过接收到返回信号的返回时间，通过计算获取相机与场景物体之间的深度信息。由于深度信息是通过物理方法获取，所以相比于单目相机和双目相机，RGB-D 相机利用物理计算方法来实现 SLAM 方法所需计算资源大大下降。不过 RGB-D 由于是通过光信号来进行深度的测量，所以在光线复杂环境中工作时，非常容易受到外部光线噪声的影响。并且当测量光线接触表面光滑物体时会形成漫反射，会使深度测量的精度大大下降，故 RGB-D 相机只适用于室内场景下。

相机利用小孔成像原理将 3D 场景中的物体投影到 2D 成像平面，如图 2-2 所示。设相机坐标系为$O-x-y-z$，指向相机正前方的为Z 轴，向右为X 轴，向下为y 轴，模型中相机的光心为点O。三维场景中的P 点经过O 的投射后，其成像点就是相机成像平面$O'-x'-y'-z'$中的点P'。设点P 在三维场景中的坐标为$[X,Y,Z]^T$，点P'为$[X',Y',Z']^T$，相机成像平面到相机光心O 的距离为 f，如图 2-1 所示。

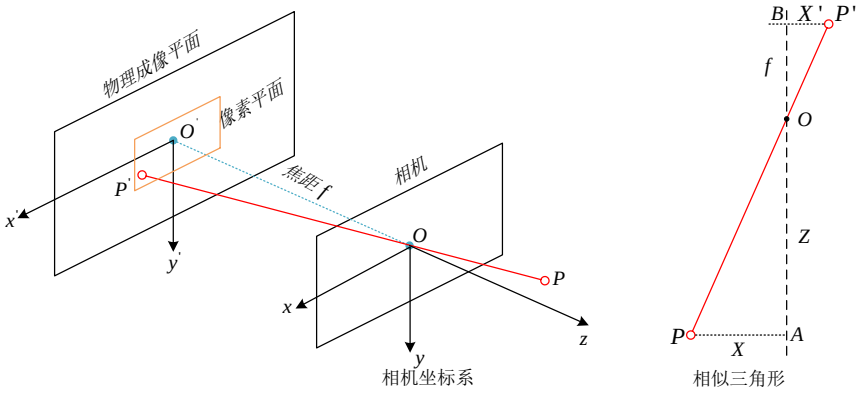


图 2-1 针孔相机模型

根据相似三角形原理，有

$$\frac{Z}{f}=-\frac{X}{X'}=-\frac{Y}{Y'} \tag{2-1}$$

因为在小孔成像模型中，在成像平面成的像是倒立的，如图 2-2 所示。

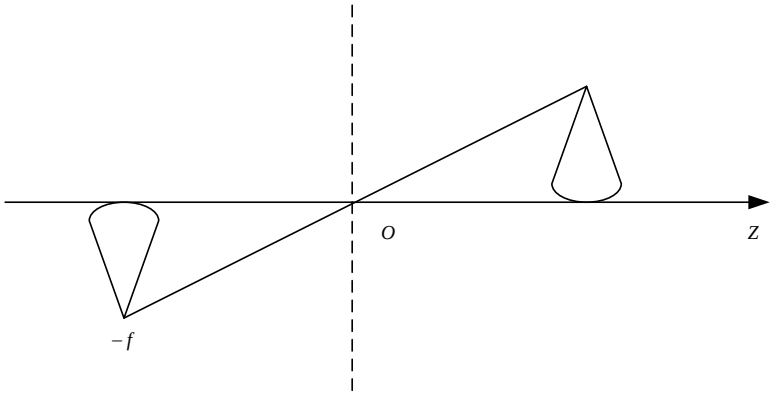


图 2-2 真实成像平面

故式(2-1)中含有负号，目前绝大多数相机会通过软件算法来翻转所成的像，这样就可以将式(2-1)中的符号去除，如式(2-2)所示：

$$\frac{Z}{f}=\frac{X}{X'}=\frac{Y}{Y'} \tag{2-2}$$

整理可得

$$\begin{aligned} X' &= f\frac{X}{Z} \\ Y' &= f\frac{Y}{Z} \end{aligned} \tag{2-3}$$

式(2-3)表示三维场景中的点 P 与成像平面的点 P' 之间的坐标变换关系。但

是相机最终获取的每一帧图像其实是一个个像素点，是由数字组成的矩阵，所以我们需要在相机成像平面对所成的像进行坐标变换和量化处理。设在成像平面中有一个固定的像素平面$o-u-v$，o'为像素坐标系的原点，位于每帧图像的左上角。那么P'在像素坐标系中的坐标我们可以表示为$[u,v]^T$。相机成像平面经过尺度变换和原点平移就可以转换成像素坐标系，所以P'在成像平面的坐标与在像素平面的坐标的转换关系就可以表示为

$$\begin{cases} u = \alpha X' + c_x \\ v = \beta Y' + c_y \end{cases} \tag{2-4}$$

在式(2-4)中，α和β分别表示像素平面相对于成像平面的缩放比例，c_x和c_y表示两个坐标系之间原点的平移距离。将式(2-3)带入式(2-4)，将αf和βf记为f_x和f_y，有

$$\begin{cases} u = f_x \dfrac{X}{Z} + c_x \\ v = f_y \dfrac{Y}{Z} + c_y \end{cases} \tag{2-5}$$

将式(2-5)表示成矩阵的形式：

$$Z\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = KP \tag{2-6}$$

式(2-6)中间的矩阵用K表示，被称为相机内参(Camera Intrinsics)矩阵。一般相机的内参是由其硬件结构所决定的，所以在使用过程中可以看作是固定的参数。式(2-6)中，点P的坐标是在相机坐标系下获得的，由于相机的运动，所以点P的相机坐标与其世界坐标P_w经过了位姿变换，所以用旋转矩阵R和平移向量t来描述相机的位姿，则像素坐标与世界坐标的关系就可以表示为

$$ZP_{uv} = Z\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K(RP_w + t) \tag{2-7}$$

其中R，t称为相机的外参数，它们随着相机的运动发生改变，代表着相

机的真实运动轨迹。准确计算相机的外参数，尽可能还原相机的真实运动轨迹，是 SLAM 的主要任务之一[35]。

在实际应用中，相机为了获取更高质量的图像，需要在镜头前方加上透镜，但是透镜会对光线的传播产生影响，使实际获得的图像产生畸变。此类畸变根据产生的原因分为径向畸变和切向畸变。径向畸变是由于透镜自身的形状引起的，光线入射角度会被透镜所改变。径向畸变根据现象的不同，有桶形畸变和枕形畸变两种类型。桶形畸变的现象是离光轴越远的图像位置其放大率越高，枕形畸变则相反，如图2-3所示。而切向畸变的产生是成像平面偏离了垂直平面所导致的，如图 2-4 所示。

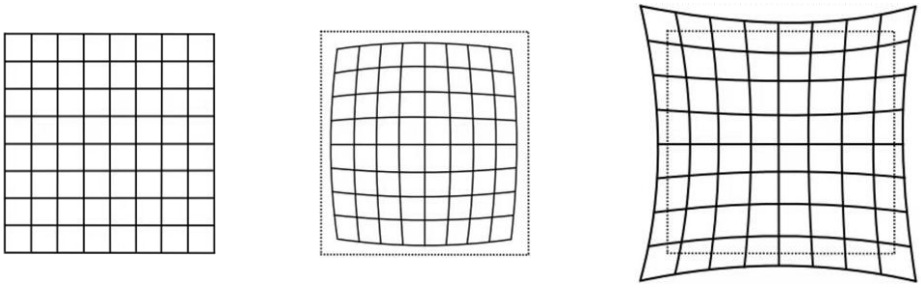


图 2-3 径向畸变

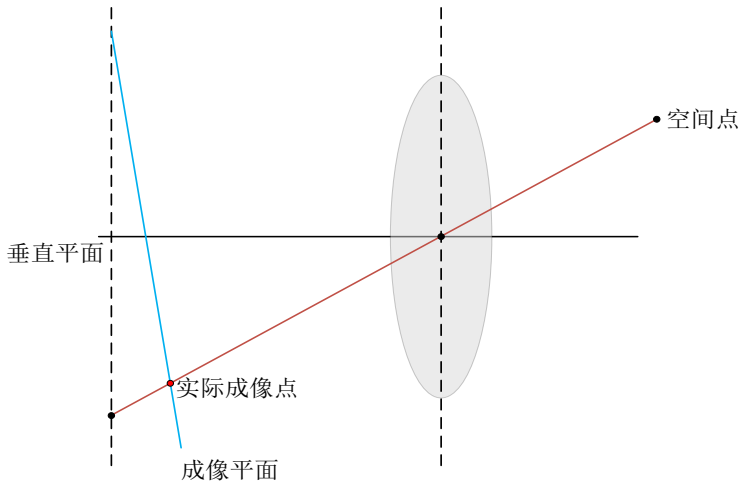


图 2-4 切向畸变

解决畸变的问题，需要为畸变问题构建数学模型。径向畸变可以看作是坐标点距离原点的长度发生了变化，将径向畸变用多项式来表示，如式(2-8)：

$$\begin{cases} x_{distorted} = x(1+k_1r^2+k_2r^4+k_3r^6) \\ y_{distorted} = y(1+k_1r^2+k_2r^4+k_3r^6) \end{cases} \tag{2-8}$$

式(2-8)中，$[x,y]^T$，$[x_{distorted},y_{distorted}]^T$ 分别是畸变前和畸变后的成像点在归一化平面的坐标。r 代表成像点与坐标系原点之间的距离。对于切向畸变，可以引入 p_1 和 p_2，用另外的多项式来表示：

$$\begin{cases} x_{distorted} = x+2p_1xy+p_2(r^2+2x^2) \\ y_{distorted} = y+p_1(r^2+2y^2)+2p_2xy \end{cases} \tag{2-9}$$

将上述两式结合可得：

$$\begin{cases} x_{distorted} = x(1+k_1r^2+k_2r^4+k_3r^6)+2p_1xy+p_2(r^2+2x^2) \\ y_{distorted} = y(1+k_1r^2+k_2r^4+k_3r^6)+p_1(r^2+2y^2)+2p_2xy \end{cases} \tag{2-10}$$

可以将发生畸变后的点 $[x_{distorted},y_{distorted}]^T$ 经过内参数矩阵并结合式(2-5)就可以得到像素平面上的点 $[u,v]^T$，如式(2-11)所示

$$\begin{cases} u = f_x x_{distorted}+c_x \\ v = f_y y_{distorted}+c_y \end{cases} \tag{2-11}$$

2.1.2 前端视觉里程计

视觉里程计(Visual Odometry，VO)主要关注的是连续两帧场景间的位姿变化和相机运动，通过估计相机在连续帧之间的位姿变化，进而计算相机在三维空间中的运动状态。视觉里程计目前有直接法与特征点法两种实现方式，这两种方式分别使用了不同的计算方法来计算相机的运动位姿。

直接法是直接根据像素的亮度值来计算出相机的运动变化，不考虑场景中特征信息的提取，所以其计算时间小于特征点法，并且在纹理稀疏，缺少特征信息的场景下的表现更加鲁棒。但是直接法的一些缺点也无法忽视。首先由于直接法会利用场景图像中的所有信息来计算相机位姿变化，这使得后端的优化问题的计算量远远大于特征点法。

如何通过图像来计算相机的位姿，是视觉里程计的核心问题。一帧图像本质上就是一个由灰度值组成的数字矩阵，如果直接从矩阵的角度来计算位姿，计算量将会巨大。所以，研究者只从一帧图像中选取一些具有代表性的信息来进行处理。由于角点相较于边缘和区块，在相机发生少量的位姿变化后更具有

稳定性，所以绝大部分特征提取方法特指是角点特征信息。在这些点的基础上来计算相机的位姿的方法，被称为特征点法。目前成熟的角点提取方法有Harris 角点[36]、FAST 角点[37]等。但是在实际应用中，单纯的角点检测并不足以精确计算出相机的运动，因为角点不具有尺度变换不变性。例如在不同距离观察同一场景的情况下，有可能检测出的角点并不同。此外当镜头出现旋转时，角点的形状可能会发生变化。为了解决角点不稳定的问题，研究者们设计了更加鲁棒的图像特征，如 SIFT[38]、SURF[39]、ORB[13]等，这些图像特征相比于单纯的角点，拥有以下优点：

1.可重复性：相同特征在不同的场景中可以用同一特征点表示

2.可区别性：不同特征点对应不同的描述信息

3.高效性：特征点的数量远远小于总的像素点数量。

4.本地性：特征信息仅与邻近窗口区域有关

由于本文算法是基于 ORB 特征点来进行 SLAM 研究，下面将详细阐述该特征点的原理。

ORB(Oriented FAST and Rotated BRIEF)特征点是由 FAST 关键点和 BRIEF 描述子[40]两部分组成，这里关键点指的是图像中特征点本身。在此方法中，关键点指的是图像中那些具有显著性和可重复性的像素点，这些点对于图像的尺度、旋转和光照变化都具有一定的鲁棒性。为了准确描述关键点，研究者们人为设计了一种向量被称为描述子，用于量化关键点的局部图像信息。这两种量组合起来形成完整的 ORB 特征点。FAST 主要关注的是图像像素灰度值变化明显的部分，其优点是特征点计算时间少，如图2-5所示。它的主要思想是若图像中某一像素点与邻域圆上的 16 个像素点中的连续 12 个点(FAST12)的像素灰度值差距超过一定阈值，则这个点就可以被认为是 FAST 特征点。为了使特征点提取更加高效，改进的 FAST 特征提取方法直接选取像素点邻域圆上的第 1，5，9，13 个像素点与之比较，如果其中有超过或等于 3 个像素点的灰度值差大于阈值，则可以直接判定为 FAST 特征点，这种操作大大加快了 FAST 特征点的检测速度。第一遍检测完成的特征点容易出现密集分布的现象，所以在提取完成后还要进行非极大值抑制的方法来使 FAST 特征点在一帧图像中均匀分布。

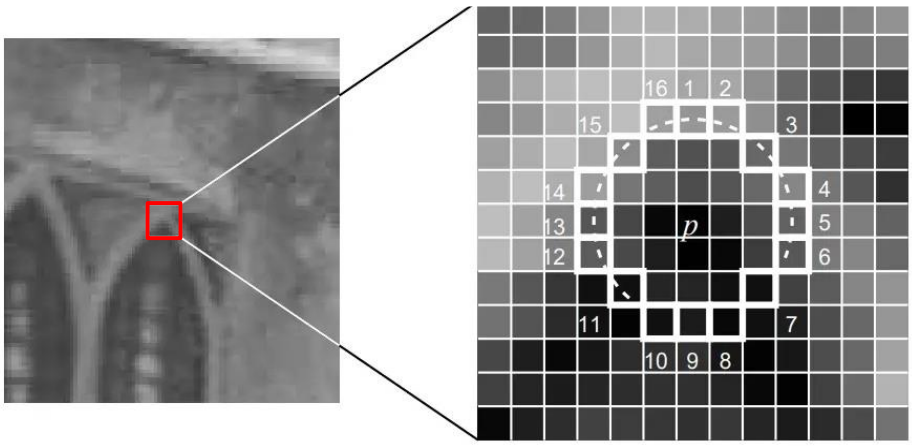


图 2-5 FAST 关键点

ORB 针对 FAST 关键点缺少尺度和指向性的缺点，加入了尺度与旋转的概念。尺度不变性是由图像金字塔来实现的，如图2-6所示。图像金字塔最下面一层是原始大小的图像，其中每一层都是对上一层图像的固定倍率缩放，这样图像金字塔的每一层都模拟了不同距离下的场景图像。其中越往上层就可以模拟相机与场景距离越远的情况。当匹配特征时，将模拟不同分辨率下的图像进行匹配，这就实现了在不同尺度下的特征点匹配。

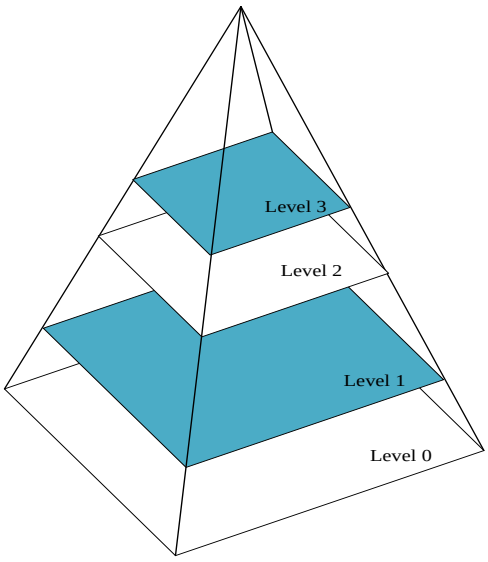


图 2-6 图像金字塔

关于旋转的概念，ORB 特征点通过计算点附近窗口的灰度质心来实现，其含义是以图像块灰度值为权重的中心点。其计算过程将特征点附近图像块的灰度质心与几何中心进行连接，其方向向量就为该特征点的方向。其中灰度质心

的计算首先要计算特征点附近半径为R的图像块 U 的矩：

$$\begin{aligned} m_{00} &= \sum_{x=-R}^{R}\sum_{y=-R}^{R} I_{(x,y)} \\ m_{10} &= \sum_{x=-R}^{R}\sum_{y=-R}^{R} xI_{(x,y)} \\ m_{01=} &\sum_{x=-R}^{R}\sum_{y=-R}^{R} yI_{(x,y)} \end{aligned} \tag{2-12}$$

那么灰度质心为

$$\begin{aligned} C_x &= \frac{m_{10}}{m_{00}} \\ C_y &= \frac{m_{01}}{m_{00}} \end{aligned} \tag{2-13}$$

图像块几何中心到灰度质心的方向就可以表示为

$$\theta = \arctan 2(c_y, c_x) \tag{2-14}$$

这种具有方向性和旋转不变性的 FAST 关键点被称之为 Oriented FAST。

BRIEF 描述子是 2010 年 ECCV 会议上提出的一种以二进制码串作为特征点的描述向量，其基本原理是在特征点附近选取 N 对像素点(p_n, q_n)比较灰度值的大小关系,并将 p_n 与 q_n 连接得到一组向量。其中当 p_n 的灰度值大于 q_n 时，向量的值为 1，反之则为 0，最后得到 N 维的由 0,1 组成的向量。

特征点完成提取后，需要进行相邻两帧图像之间的特征点匹配，如图 2-7 所示。传统的做法为暴力匹配，将每一个特征点与下一帧图像中的所有特征点都逐个计算其描述子的汉明距离，两个描述子之间的汉明距离代表了两个特征点之间的相似度。当匹配完成后，汉明距离最短的就是匹配的特征点对。虽然这种方法简单，但是当图像中的特征点数量很多时，暴力匹配方法会消耗大量计算资源，导致效率的降低。故目前采用最多为快速最近邻(FLANN)算法[41],其主要做法只在待匹配的特征点附近进行匹配，大大加快了特征点匹配的速度，这种算法的依据是假设相邻两帧图像之间的运动变换不会很大。

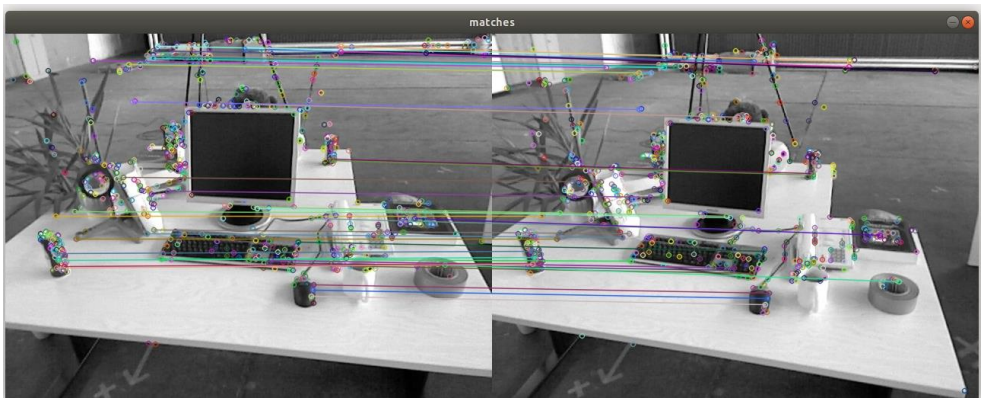


图 2-7 ORB 特征点匹配

特征点匹配结束后，需要计算两帧图像之间的位姿变换关系。单目相机所获取的图像缺少深度信息，需要利用对极几何[42]的原理来估计相机的运动，由获取的运动关系并通过三角测量原理来计算出像素点在空间中的位置；双目相机以及 RGB-D 相机可以直接获取像素点在场景中的深度信息，所以像素点在三维空间中的坐标是已知的，可以采用 PnP[43]、ICP[44]算法来计算相机的位姿变换。ICP 算法是直接通过相邻帧匹配的相机坐标来计算位姿，相机坐标的深度值是直接通过传感器获取，因此 RGB-D 相机的自身精度对其影响较大。PnP 算法通过一些 3D-2D 点对信息，即已知的世界坐标下的 3D 坐标，以及这些 3D 坐标在相机图像坐标系上的 2D 投影点，利用这些信息来计算相机的位姿变换。

2.1.3 后端优化

视觉里程计只考虑相邻帧之间的相机位姿变换，若仅通过视觉里程计来计算相机的位姿，会不可避免的受到一些噪声的干扰，例如视觉传感器的噪声、环境噪声等，这些噪声会随着相机运动的过程逐渐累积，会导致相机运动轨迹的计算出现漂移，无法构建全局一致的地图。后端优化则是为了解决位姿估计过程的噪声问题，将前端视觉里程计的估计值与观测值进行全局的优化，从而矫正相机运动的整体轨迹。

传统 SLAM 中的后端优化使用的滤波器在一定程度上假设了马尔可夫性，即假设机器人当前k时刻的运动状态只与$k-1$时刻的状态与观测有关，而与$k-1$时刻之前无关，这就导致对于相机位姿的优化只考虑相邻帧之间的关系。而实际上相机当前位姿不仅与前一时刻有关，同时也会受到历史观测数据噪声的影响，这就需要使用非线性的优化方法来进行全局的校正。拓展卡尔曼滤波

(Extended Kaiman Filter, EKF)方法是标准卡尔曼滤波在非线性状态下的一种优化方法，但是 EKF 的有效工作范围有限，只能将小范围内的非线性多项式进行线性化处理，若运动方程的参数过多，超出了 EKF 的计算资源，则会导致较大的非线性误差，因此 EKF 往往不适用于大场景下的后端优化。集束调整(Bundle Adjustment, BA) 优化是目前 SLAM 中主流的非线性优化方法，主要做法是首先根据相机模型和匹配两帧图像的特征点，将其中一帧中特征点对应的空间坐标，并将其投影到另一帧成像平面上，其投影点与对应匹配点不会重合。集束优化就是将每对匹配点的投影误差组成超定方程，利用非线性优化方法求解最佳结果。

2.1.4 回环检测

前端视觉里程计提供特征信息的提取以及相机位姿的初值，后端对这些数据进行非线性优化。但是视觉里程计只考虑相邻帧之间相机位姿的变化，但两帧之间的位姿变换存在一定误差，这些误差会随着系统的运行而进行累积。后端优化会将对这些累积的误差进行非线性优化，但是只经过后端优化处理不足以使估计的相机运动轨迹与真实轨迹完全对应，导致在构建全局地图和轨迹时出现不一致性的问题。

回环检测的含义就是检测机器人是否到达过相同的场景，若检测到出现回环，就可以将当前帧的场景与之前的历史图像帧进行关联，并将整个运动过程中累积的历史误差进行修正，得到全局一致的地图和运动轨迹。回环检测最原始的方法就是暴力匹配，将当前帧与之前观测的每一帧数据进行匹配，若两帧之间的特征信息相似度达到一定阈值，则判定当前两帧出现回环。但是此种做法的计算量巨大，随着相机持续的运动，每一次匹配所做的计算就越复杂。所以目前视觉 SLAM 主流的回环检测方法是建立词袋模型(Bag-of-Words,BoW)[45]来衡量两帧场景中的相似度。词袋模型的概念就是将场景中的特征信息视为一个单词，那么需要训练一个包含所有类型特征的集合，并在此集合的基础上为图像中的每个特征创建一个对应的词集，这也被称为词袋。这样可以通过比较词集来评估图像的相似性，从而大大加快了回环检测过程。

2.1.5 建图线程

建图是 SLAM 系统的两大任务之一。在视觉 SLAM 中，地图代表了所有被观测的路标点集合，因此在视觉里程计或是后端优化中，都是在对路标点的位姿进行建模，也就是建图。在实际的应用中，由于任务的复杂度不一样，所以对于地图的需求也不相同。一般对于视觉 SLAM 地图的需求，主要有以下几种：

1.定位。定位是地图的基本作用。在视觉里程计以及后端优化中，花费了大量的资源来计算相机的位姿，也就是实现定位功能。若将当前场景的全局地图保存，当机器人关机后重新工作时，会在已保存的地图上继续进行 SLAM 工作，而不是重新进行位姿初始化等工作，此类工作只需要稀疏的点云地图就足以完成。

2.导航与避障。导航与避障的意思是机器人可以在地图信息中规划路径，在任意两个路标点之间规划出最佳的路线，并且避开其中的障碍点，这就需要机器人知道在地图当中那些位置是可以通过，哪些是不可以通过，此类工作则需要稠密地图的数据信息才可以完成。

3.重建与交互。有一些应用需要 SLAM 技术来实现场景的重建，这类需求一般用来展示或者通信行业，其中人机交互也是基于这一类需求所实现，通常用在增强现实(Augmented Reality，AR)。此类任务不仅需要稠密的地图信息，同时要求地图中要包含丰富的纹理信息。

2.2 ORB-SLAM2 基本框架

ORB-SLAM2 是目前视觉 SLAM 的经典算法之一，也是首个同时支持单目、双目、RGB-D 相机传感器的开源 SLAM 方案。ORB-SLAM2 有三个并行的线程，分别是跟踪线程、局部建图线程以及回环检测线程。本文在室内场景下使用 RGB-D 相机来实现动态 SLAM 方法，因此在 ORB-SLAM2 的基础框架上进行相关的研究工作。

跟踪线程的第一个工作是提取场景中的特征信息，这里为 ORB 特征点。提取特征信息完成后进行相机位姿的初始化，主要有三种方法：恒速运动模型、参考关键帧模型、重定位模型。恒速运动模型是假设相邻帧之间的运动速度无差异，将当前帧根据前一帧的运动速度来进行位姿计算，再将匹配的特征信息

进行重投影匹配。若恒速运动模型匹配失败，则会进行参考关键帧匹配。若计算每一帧的特征匹配，会占用计算资源过大，使系统无法进行实时的位姿计算，故 ORB-SLAM2 在跟踪线程中引入了关键帧的概念。由于相机使连续运动，所以采集到的连续的图像信息中有太多冗余信息，ORB-SLAM2 中在跟踪线程中会计算每一帧之间的相似度，当相似度低于一定阈值或两帧相隔一定时间时，会将此帧设为关键帧，在后面的位姿计算以及地图构建时，只计算关键帧的信息，这大大降低了系统的计算量。在参考关键帧模型中，会将当前帧与上一关键帧进行词袋模型匹配，若匹配失败，则会进行重定位模型来估计位姿，主要使用 BoW 模型进行关键帧相似度匹配或者进行位姿的重新估计再进行 BA 优化。

跟踪线程所筛选的关键帧会在局部建图线程中得到处理。跟踪线程所传入的关键帧会保存在局部建图线程中的关键帧队列中，首先会利用已经生成的地图点对关键帧进行共视关系匹配，再剔除错误的地图点并生成新的地图点。局部建图线程会删除冗余的关键帧，对共视关键帧和生成的局部地图点进行优化，并将重投影误差降至最低，最后会将当前关键帧插入到回环检测的关键帧队列中。

回环检测检测线程主要任务使计算当前相机所在位置是否在之前到达过，并在发生回环时进行矫正。首先会将当前关键帧与历史关键帧进行 BoW 模型匹配，检测是否发生闭环场景。如果发生回环，则使用 Sim3 相似性变换生成地图点，使用本质图匹配优化位姿，最后进行 BA 全局优化，以实现全局消除累积误差，提高系统的鲁棒性。

2.3 本章小结

本章从视觉传感器、前端视觉里程计、后端优化、回环检测、地图构建这五个方面对视觉 SLAM 进行了详细分析。将相机成像模型进行数学建模进行研究，同时对 ORB 特征点的提取以及匹配方法进行分析，系统地分析了经典视觉 SLAM 的技术与方法。随后对 ORB-SLAM2 的基本框架以及各线程进行分析，本文在室内针对动态场景下的 RGB-D SLAM 算法研究是在 ORB-SLAM2 的基础框架上进行设计开发。

第3章 改进的轻量化 YOLOv5s 目标检测网络研究

传统 SLAM 工作可以使机器人在未知环境下实现定位、建图等任务，然而在面对复杂场景，例如环境中存在大量动态物体时，SLAM 中的跟踪过程会受到这些动态信息的影响，造成数据关联错误等问题，会导致位姿估计失败和运动轨迹计算结果的漂移。有一些任务会需要机器人不仅要建立全局一致的地图，而且还需要获取场景中物体的类别信息，这类任务就需要得到场景中的语义信息。故本文在 SLAM 系统的跟踪线程中加入基于深度学习的目标检测网络线程，以此来获取当前场景下物体的类别信息以及识别动态物体并将这些动态的特征信息在后续的工作中剔除，实现来保证 SLAM 算法在动态场景中对位姿估计和相机运动轨迹计算的准确性。

3.1 卷积神经网络基本结构

卷积神经网络(Convolutional Neural Networks, CNN)被广泛的应用在图像和视频处理领域，相较于传统的神经网络，CNN 具有局部感受野、参数共享与降采样等特点，可以自动学习图像中的特征信息，因此对于图像和序列数据有着更好的处理能力[46]。典型的 CNN 由输入层、卷积层、池化层、全连接层与输出层组成，如图 3-1 所示，下面将研究各个部分的工作原理。

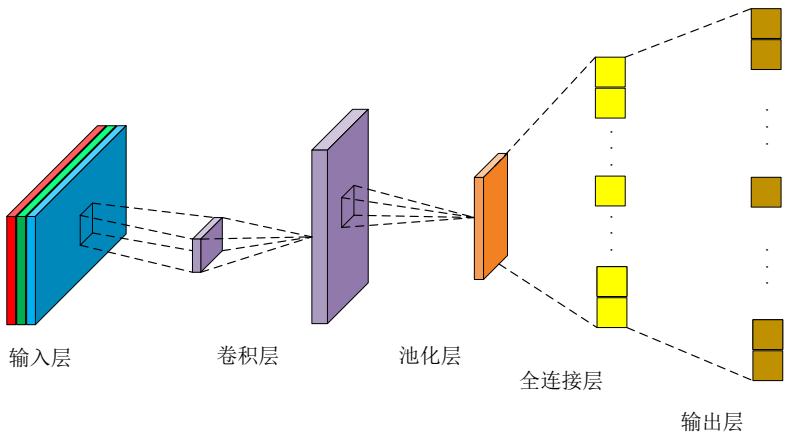


图 3-1 卷积神经网络结构

输入层的任务是将图像数据作为输入传入网络，CNN 的主要任务就是进行对图像相关数据的处理。图像信息是通过二维矩阵的形式保存在计算机中，图

像中每一个像素的灰度值大小组成了这个二维矩阵，如图3-2所示。像素灰度值的大小范围为 0~255，0 表示纯黑色，255 表示纯白色。日常生活中的彩色图像由三个通道组成，分别是红（R）、绿（G）、蓝（B），也被称为 RGB 图像，RGB 图像在作为神经网络的输入时，会以三个二维矩阵的形式保存并进行处理。

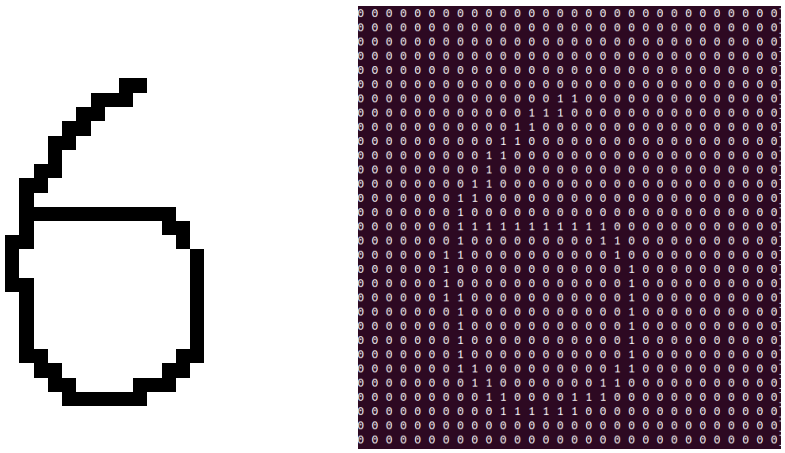


图 3-2 图像与二值矩阵

卷积层是 CNN 的核心层，其本质就是对图像矩阵与滤波矩阵做内积，以此来提取图像中期望获取的特征信息。在卷积神经网络中，卷积层的设计融合了局部感受野和参数共享的概念，这两者对于网络性能的提升至关重要。局部感受野的核心理念在于，图像中的像素间存在着局部的空间连接性。由于这种局部性，卷积层中的每个神经元无需对整幅图像进行全面处理，而是专注于感知其所在局部区域的信息。这种局部处理的策略不仅减少了计算复杂度，还使得网络能够更高效地提取图像的局部特征。与此同时，参数共享机制进一步提升了卷积层的效率。在卷积层中，不同位置的神经元使用相同的权重和偏置参数进行卷积操作，从而实现了参数的共享。通过结合局部感受野和参数共享，卷积层能够在保持高效计算的同时，有效提取图像的局部特征，为后续的图像识别和处理任务提供了有力的支持。因此，在卷积神经网络中，局部感受野与参数共享的思想被广泛应用，成为了构建高效且性能优越的卷积层的关键所在。这一设计使得卷积神经网络在计算机视觉领域取得了显著的成果，为图像分类、目标检测等任务提供了强大的工具。卷积计算过程如图 3-3 所示。

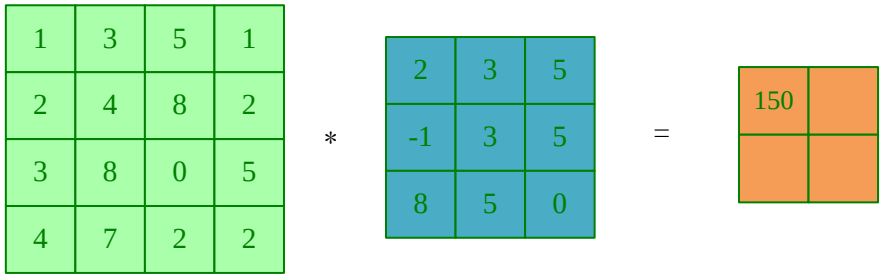


图 3-3 卷积计算

图 3-3 中的输入图像为 5*5 的二维矩阵，卷积核的大小为 3*3，其输出特征图的第一个元素的计算如式(3-1)所示。

$$O_{0,0}=(1*2)+(3*3)+(5*5)+(-1*2)+(4*3)+(8*5)+(3*8)+(5*8)=150 \quad (3\text{-}1)$$

整个卷积过程就是将输入数据降维的过程，通过卷积核不停的移动并计算，可以从图像中提取可用的特征信息，其中卷积核每次移动的距离，被称为步长(stride)，图 3-3 中的步长为 1。可以看到图像的边缘部分在卷积运算中只被计算了一次，当特征包含在边缘信息中，则会影响对特征的提取，所以在对输入图像矩阵进行卷积之前，可以在二维矩阵的周围进行拓展，以此来达到每个位置都被公平计算的目的，这样就不会丢失包含在边缘信息中的特征，此类做法又被叫做 Padding，如图 3-4 所示。

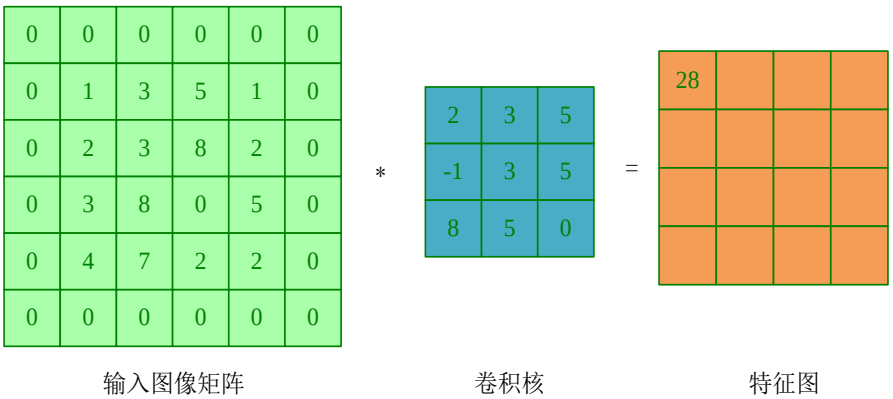


图 3-4 Padding

在卷积计算中，每次卷积核只能接收自身大小的局部输入，这就是局部感受野的思想，卷积核会通过移动计算整个图像矩阵，这体现了参数共享的概念。

在实际使用卷积神经网络对图像信息进行处理的过程中，每个参数不同的卷积核都对应不同的特征，所以通常会使用多个参数不同的卷积核来对同一图

像信息进行特征提取，这样就可以获得图像中不同的特征信息。

由于使用了多个卷积核进行卷积，得到的输出特征图此时是一个三维矩阵。通常一个完整的卷积神经包含有多个卷积层，那么上一层卷积运算得出的特征图输出会作为下一卷积层的输入数据，设第 L 层卷积层得到的输出特征图尺寸为 $H \times W \times D$，卷积核尺寸为 $H^l \times W^l \times D$，个数为 D^l，Stride 值为 S，Padding 值为 P，那么得到的特征图输出的尺寸为：

$$(\frac{H-H^l+2P}{S}+1)\times(\frac{W-W^l+2P}{S}+1)\times D^l \tag{3-2}$$

随着 CNN 中层数的增加，其局部感受野的大小也在变化，当感受野的值越大，其特征图包含的特征信息更为全局；当值越小时，特征图所包含的特征信息就越趋向局部和细节。感受野的变化过程如图 3-5 所示：

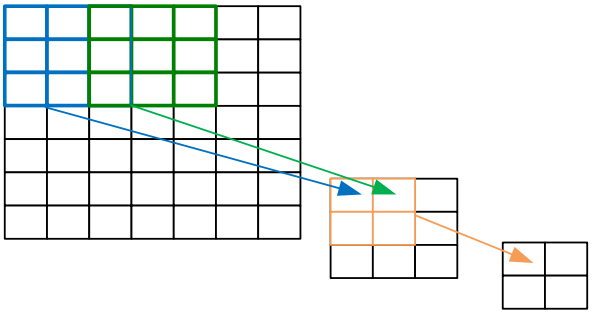


图 3-5 卷积神经网络感受野

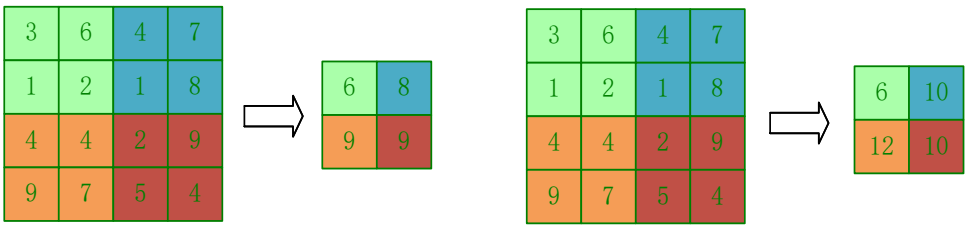
图 3-5 中，输入的图像矩阵大小为 7×7，CONV1 中 kernel_size=3，stride=2，padding=0，CONV2 中 kernel_size=2，stride=1，padding=0，通过式(3-11)计算可得最终输出特征图的尺寸为 2×2。可以看出，输入图像矩阵中每个元素感受野为 1，CONV1 中每个元素的感受野为 3，而 CONV2 中每个元素的感受野为 5，其计算公式如下：

$$RF_n=\begin{cases} K_1 & n=1 \\ RF_{n-1}+(K_{n-1})\prod_{i=1}^{n-1}S_i & n>1 \end{cases} \tag{3-3}$$

式(3-3)中，RF_n 为第 n 层的感受野大小，K_n 表示第 n 层卷积核的尺寸，S_i 表示第 i 层的 Stride，因此可以看出，经过数层小卷积核计算后得出的特征图，

其感受野的大小与只使用一层大尺寸卷积核进行卷积的感受野大小相当，这样做的好处是在减少参数数量的同时可以提高网络容量。

根据式(3-2)可得，特征图的个数 D^l 与当前卷积层的卷积核个数相等，卷积核的个数也对应着可以提取的特征数量。在实际对图像的卷积操作中，卷积核的数量会很庞大，这也导致特征图的数量会非常多，这样会出现很多冗余的特征信息，导致过拟合和参数量过大的问题。池化层的出现就是为了解决这个问题。池化层又称之为降采样，其本质就是在卷积操作后，将输出的特征图再进行一次特征提取，将最显著的特征提取出来，就可以实现减小过拟合和减低参数量的问题。目前主流的降采样有最大池化和平均池化两种，最大池化就是取局部信息的最大值，目的是获取当前窗口最显著的特征信息；平均池化就是取局部信息的平均值，目的是将局部所有信息与此处特征相关联。其过程如图 3-6 所示。



最大池化　　平均池化

图 3-6 池化处理

原始的图像矩阵经过卷积层和池化层的处理后，会输出尺寸为 $H \times W \times D$ 的特征图，在分类任务中，卷积神经网络最终需要输出原始图像相对于某一期望特征的概率，所以在输出层之前，需要将特征图展开成尺寸为 $1 \times H \times W \times D$ 的一维向量，这个过程也称为全连接，而这一层叫做全连接层(Fully Connected Layers，FC)。输出层将全连接层输出的一维向量进行计算处理，就会得到相应标识的概率[47]，在深度学习中，需要识别的结果通常是多类的，因此每个位置都有一个概率值，表示被识别为当前值的可能性，概率值最高的就是最终的识别结果。在训练过程中，可以不断调整参数值，最大限度地提高模型的准确性，使识别结果更加准确。

3.2 目标检测网络轻量化改进

3.2.1 YOLO 算法

随着近年来卷积神经网络的快速发展，目标检测成为了计算机视觉领域的热门研究问题。目前目标检测主要可分为两类方法，第一类先利用区域提议网络(Region Proposal Network，RPN)[48]生成待定区域，再对待定区域进行分类和边界框回归，这类方法从输入到输出由两个阶段组成，所以也称为两阶段目标检测算法(two-stage)，典型的算法如 RCNN[49]，Faster-RCNN[50]；第二类可以直接在网络中生成预测结果，不需要提取候选区域的单阶段目标检测方法(one-stage)，其典型方法就是 YOLO 系列算法。

YOLOv1[51]在 2016 年发布，其核心思想是将目标检测问题转化为回归问题进行处理。具体做法就是首先将输入图片分成$S\times S$的网格，若一个物体的中心点落在某个网格中，那么此网格就负责预测该物体的信息。其中每个网格会预测两个包围框，每个包围框含有位置信息，置信度得分信息以及标签概率信息，其原理如图 3-7 所示。

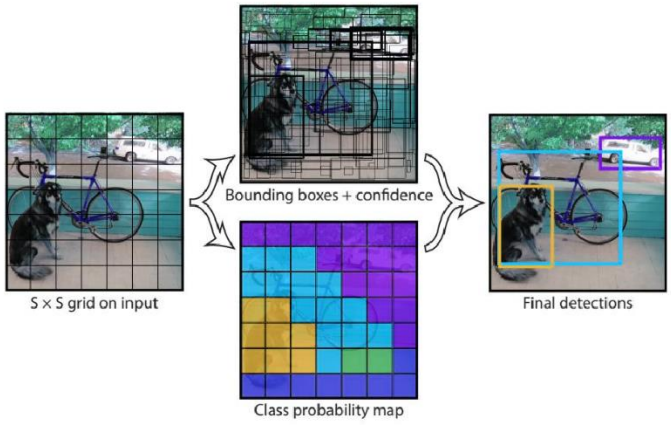


图 3-7 YOLOv1 目标检测方法

YOLOv1 针对 PASCAL VOC 数据集，将图片分成了 7×7 的网格，每个单元格需要回归 2 个边界框，每个边界框包含了四个坐标信息，分别是中心点(x, y)，边界框的高h和宽w以及置信度得分C，每个单元格还需要预测 20 中类别信息，故 YOLOv1 网络的输出是一个 7×7×30 的张量，其整体网络架构如

图 3-8 所示。

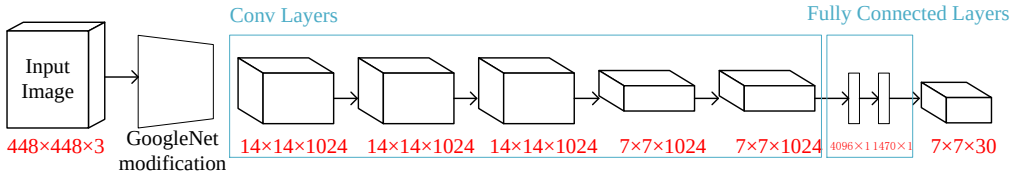


图 3-8 YOLOv1 网络架构

从图 3-8 可以看出，YOLOv1 主要是由输入层、卷积层、全连接层和输出层构成，其输入是 448×448×3 的彩色图像，经过 24 个卷积层处理，在通过两个全连接层进行线性回归，最终输出预测结果。YOLOv1 的损失函数是由三部分组成，分别为坐标损失，置信度损失和网络预测类别损失

（1） 坐标损失

$$L_{loc}=\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}O_{ij}^{obj}[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2]+ \\ \lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}O_{ij}^{obj}[(\sqrt{w_i}-\sqrt{\hat{w}_i})^2+(\sqrt{h_i}-\sqrt{\hat{h}_i})^2] \tag{3-4}$$

如式(3-4)所示，坐标损失由预测框中心坐标误差和宽高数值误差组成，O_{ij}^{obj} 表示第 i 个网格中的第 j 个预测框中是否负责预测某个物体，若负责预测则为 1，相反则为 0，所以只有当预测框对网格中标定的真实框(ground truth)负责时，才会对坐标损失进行计算。λ_{coord} 为坐标损失的权重系数，宽高使用平方根来求误差会使得小框对误差更敏感。

（2） 置信度损失

$$L_{con}=\sum_{i=0}^{S^2}\sum_{j=0}^{B}O_{ij}^{obj}(C_i-\hat{C}_i)^2+\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}O_{ij}^{noobj}(C_i-\hat{C}_i)^2 \tag{3-5}$$

如式(3-5)所示，置信度损失由两部分组成，第一部分是包含目标时的置信度损失，第二部分是不包含目标时的置信度损失。O_{ij}^{obj} 和 O_{ij}^{noobj} 体现了预测框中是否包含目标的情况，$\hat{C}_i$ 为置信度预测值，C_i 为置信度得分，在 YOLO 中，置信度的取值范围是 0~1，其含义是模型对目标的检测确信程度，其计算公式如下：

$$C = Pr(object) * IOU_{pred}^{truth} \tag{3-6}$$

其中 $Pr(object)$ 表示边界框内是否存在对象，若存在为 1，反之则为 0。IOU_{pred}^{truth} 表示预测框与真实框之间的交并比，如图 3-9 所示。

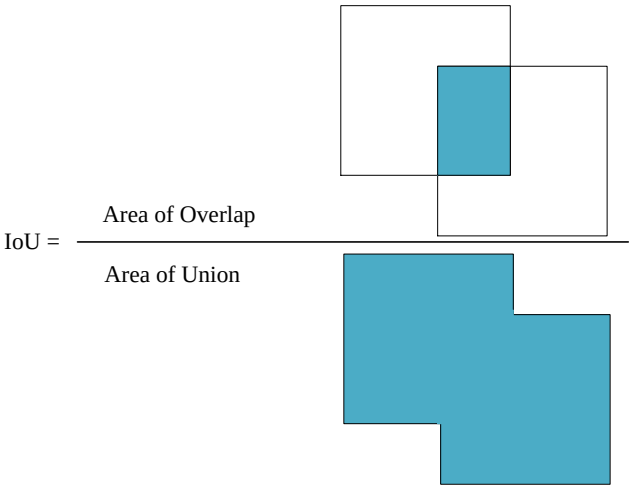


图 3-9 IoU 示意图

（3） 网络预测类别损失

$$L_{cla} = \sum_{i=0}^{S^2} O_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{3-7}$$

故 YOLOv1 网络的总损失函数为式(3-8)

$$L = L_{loc} + L_{con} + L_{cla} \tag{3-8}$$

由式(3-5)~(3-7)可知，当预测框中不含有目标时，其损失函数只计算其置信度损失。YOLOv1 的运行速度快，可以实现实时运行，其网络结构简单，在检测准确度与速度之间取得了较好平衡，加快了目标检测网络的发展。但是由于输出层为全连接层，所以 YOLOv1 只支持固定尺寸的输入图像，且由于每个网格只包含两个预测框，最终只能输出 IOU 最高的预测框作为物体检测，所以当网格中包含多个物体时，只能检测出其中的单个物体 。

YOLOv2[52]在 v1 的基础上，将主干网络设计为 Darknet-19 的网络结构，参数量得到大幅下降。v2 舍弃了全连接层，对每一层的输入数据都进行批量归一化(batch normalization)[53]操作，网络的收敛速度得到了加快。在 YOLOv2 中参考了 Mask R-CNN，引入了先验框(anchor boxes)的概念，在每个网格中预设了一组大小和宽高比不同的边框，使得每个网格可预测的类别数量得到提升。同时

YOLOv2 使用了细粒度特征融合(Fine-Grained Features)方法，将浅层特征与深层特征进行连接后进行输出，提高了对小目标物体的检测能力。

相对于 YOLOv2 的主干网络，YOLOv3[23]将 Darknet-19 替换成了 Darknet-53，其主要由两种尺寸的卷积层组成，分别为 1×1 和 3×3，每个卷积层后还有一个批量归一化层和 Leaky Relu 激活函数，Darknet-53 中的基本卷积单元 DBL 就是由卷积层，批量归一化层以及 Leaky Relu 组成的。YOLOv3 中加入了残差网络[55]，解决了网络在训练时出现的梯度消失和梯度爆炸问题。YOLOv3 在对于小目标检测方面，采取了 3 种尺度的先验框，且每种先验框还具有三种尺度，故共有 9 种不同尺度的先验框，其目的是预测不同尺度的目标物体。在 YOLOv1 和 YOLOv2 中，由于输出层采用的 softmax 函数，其输出值只能为 0 或 1，所以每个目标只属于一种类别。但是实际应用场景中，一个目标可能包含多个标签种类，所以 YOLOv3 采用了 sigmoid 函数，将输出的范围控制在 0~1 内，若输出的值大于某个预设阈值时，就代表该目标包含有某个标签类别。之后的 YOLOv4[54]则在 v3 的基础上加入了更多的卷积处理和残差网络，将当时各类先进算法都加入了网络架构中，提升了其性能表现。

3.2.2 YOLOv5 目标检测网络架构及流程

YOLOv5[56]相较于 v4 具有更轻量化的网络结构，其参数量得到了大幅降低。目前 YOLOv5 共衍生出了 YOLOv5s、YOLOv5m、YOLOv5l、YOLOv5x 四个版本，其中 YOLOv5s 为网络深度和宽度最小的模型，其网络结构框架如图 3-10 所示。

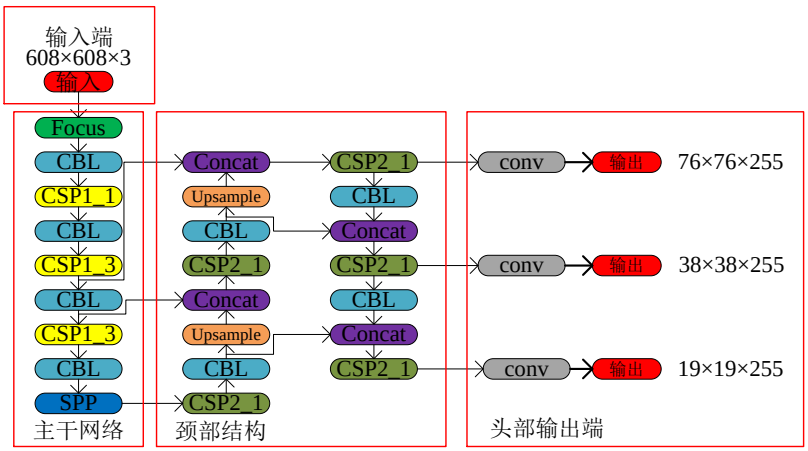


图 3-10 YOLOv5s 网络结构图

YOLOv5 作为一个先进的实时目标检测模型，其高效的网络结构图是其性能卓越的关键。该网络结构主要由几个核心部分组成：输入端、主干网络、颈部结构和头部输出端。在输入端，YOLOv5 引入了一种名为 Mosaic 的数据增强技术。这一技术通过精心设计的策略，将四张图片进行随机缩放、裁剪，并以特定的方式拼接在一起。这种数据增强方法不仅增加了训练数据的多样性，而且有效地提高了网络对于小目标的检测能力，因为通过拼接，小目标在不同尺度、不同位置上都得到了充分的展示。YOLOv5 将计算初始锚框的过程直接整合到了代码中。这意味着模型能够在训练过程中自动适应不同数据集，并计算出最佳的锚框值。这一创新大大提高了模型的灵活性和泛化能力。此外，YOLOv5 在图片缩放填充方面也进行了优化。传统的缩放填充方法往往会在图像周围添加大量的黑边，这不仅降低了图像的有效利用率，还可能导致模型推理速度下降。而 YOLOv5 通过改进算法，确保在缩放填充时添加最少的黑边，从而加快了网络的推理速度，进一步提升了模型的实时性能。

在主干网络中，Focus 结构会先对输入图片进行切片操作，类似于邻近下采样，将输入图片在没有信息丢失的情况下将输入通道扩大 4 倍，将切片后的图片进行拼接，这样原始的 3 通道 RGB 图像就变成了 12 通道，如图 3-11 所示。

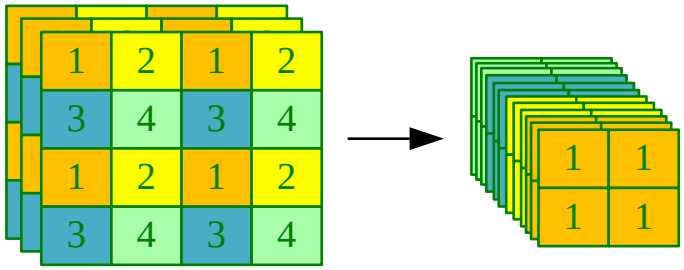
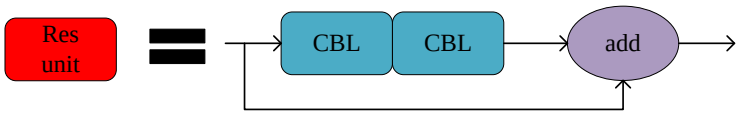


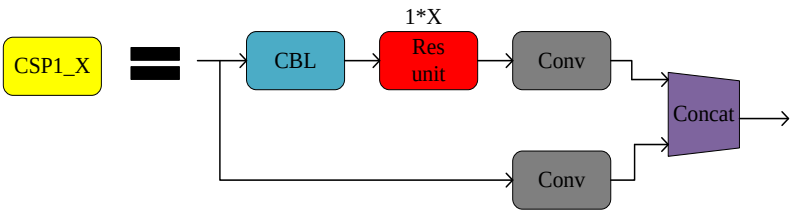
图 3-11 Focus 切片操作

YOLOv5 一共设计了两种 CSP 结构，在主干网络中使用 CSP1_X，在颈部结构中使用 CSP2_X。CSP 结构是借鉴 CSPNet（Cross Stage Paritial Network）[57] 设计的，其目的是降低网络模型中的计算量，CSP 模块将基础层的特征分为两个部分，通过一系列处理将两部分进行拼接，在保证网络准确率的同时降低了计算量。CSP 的结构如图 3-12 所示。

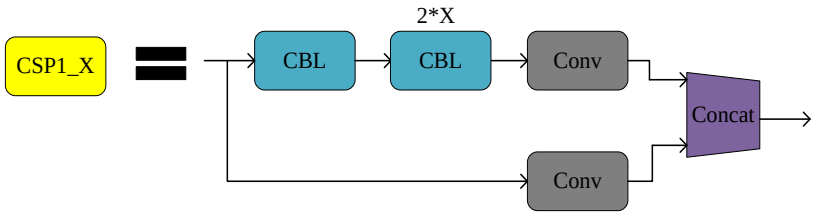
(a) CBL 模块



(b) Resunit 模块



(c) CSP1_X 模块



(d) CSP2_X 模块

图 3-12 CSP 结构

YOLOv5 在颈部结构中使用了 FPN+PAN 的结构，其设计参考了 PANet[58]的架构。FPN 层从上向下传达强语义特征信息，PAN 层自底向上传达强定义特征，两者在不同的层中进行参数聚合，实现了不同尺度的特征融合。

在头部输出端中，只对颈部结构输出的特征图进行卷积操作，而后生成三种对应尺寸的检测输出结果，在头部输出端中的卷积结构仅由三个 1×1 的卷积核构成，其会将输入的特征图通道变成估计尺寸。若输入的特征图网格尺寸为 $S\times S$,经过卷积操作后会生成尺寸为 $S\times S\times 3\times(4+1+n)$ 的张量，3 代表每个网格中包含三个锚框信息，4 代表每个锚框的中心点坐标与宽高信息，1 代表置信

度，n 表示需要检测的类别数量。

3.2.3 轻量化目标检测网络 YOLOv5s-GSConv

目前 YOLOv5 还在不断迭代，v6.x 版本相比较前几个迭代版本，具有更加精简的网络结构。YOLOv5s 为 YOLOv5 的轻量化版本，但是本文主要研究内容为室内动态场景下的 RGB-D SLAM 算法，其需要将模型移植到搭载 RGB-D 摄像头的移动机器人中，因此对于模型的实时性需求较高，故本文将在 YOLOv5s 的基础上对其进行轻量化改进，以满足本论文的研究要求。

本文旨在优化网络结构，实现轻量化处理。为实现这一目标，我们提出了设计轻量级特征提取单元的策略，用以替代 YOLOv5s 中的传统卷积单元。这一思路的核心在于通过简化特征提取过程，在合理的检测精确度内降低网络复杂度。近年来，相关领域的研究人员提出了分组卷积和深度可分离卷积等特征提取单元，以减少传统卷积处理的参数量和计算复杂度。这些轻量级的卷积模块被广泛地应用在各类轻量级网络模型中，如 MobileNet[59]，Xception[60]等。

分组卷积（group Convolution）最开始是为了解决显存不足的情况而被使用在 AlexNet[61]中，它将输入特征图在通道维度上分成几个组，并在每个分组中再使用独立的卷积核进行卷积运算，再将每组输出的特征图进行合并来构建最终的输出，其计算过程如图 3-13 所示。

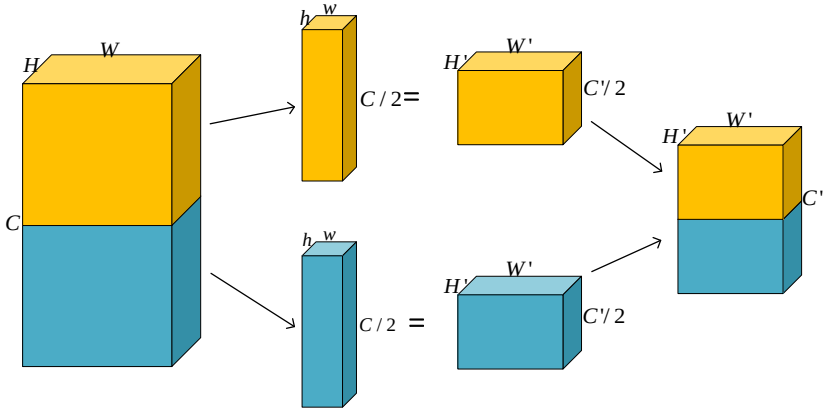
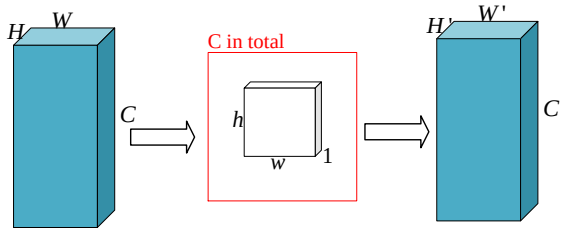


图 3-13 分组卷积

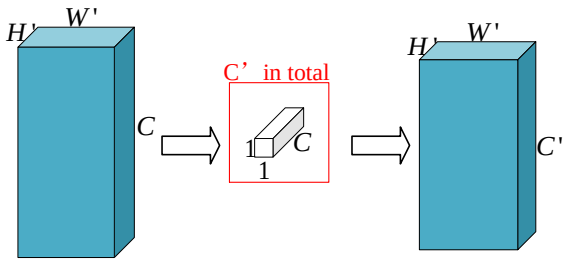
在图 3-13 中，输入特征图的尺寸为$H \times W \times C$，H 和W 表示输入特征图的长和宽，C 表示通道数。输出特征图的尺寸为$H' \times W' \times C'$，若使用传统卷积方式来计算的话，其需要一组个数为C'，尺寸为$h \times w \times C$的卷积核组来进行卷积操作，传统卷积的参数量则为$h \times w \times C \times C'$。分组卷积将输入特征图按照通道

平均分为数个组，在图中分为了两组，再对两组分别使用独立的卷积核进行卷积，此时每个卷积核的尺寸为$h\times w\times C/2$，此时经过卷积后的期望输出通道数为最终输出的一半，所以需要$C'/2$个卷积核，最后将每组的输出进行拼接得到最后的输出，此时的输出与传统卷积得到的输出是一样的，但是分组卷积的参数量为$h\times w\times C/2\times C'/2$，可以看出，分组卷积的参数量相较于传统卷积下降了一半。因此，分组卷积先将输入特征图按照分组进行划分，随后在各分组内部独立进行卷积操作，并最终将各分组的卷积结果进行拼接，这种策略能够显著减少参数量，参数量下降的程度与分组数成反比。但是分组卷积在使用时需要注意其分组数，如果分组数过多会导致特征的丢失或信息的损失，并且每个分组中的卷积核只关注本分组的局部特征，容易导致每组之间的特征信息存在屏蔽和阻塞，从而使整体网络的检测准确性下降。

深度可分离卷积(Depthwise Separable Convolution)[62]是一种特殊的分组卷积方法，它将传统的卷积拆分为两个步骤进行，分别是深度卷积(Depthwise Convolution)和逐点卷积(Pointwise Convolution)，经过两步序列化运算后输出的特征图与传统卷积结果相同，计算过程如图 3-14 所示。



(a)　深度卷积



(b)　逐点卷积

图 3-14 深度可分离卷积

在图 3-14 中，输入特征图的尺寸为$H\times W\times C$，目标输出特征图尺寸为$H'\times W'\times C'$，深度可分离卷积利用C个尺寸为$h\times w\times 1$的卷积核对输入特征图的每个通道进行卷积，得到尺寸为$H'\times W'\times C$的输出特征图，此步骤为深度卷积，其参数量为$h\times w\times C$。在下一步逐点卷积中，使用C'个尺寸为$1\times 1\times C$的卷积核对上一步输出的特征图进行卷积，最终输出与传统卷积输出一样的尺寸为$H'\times W'\times C'$的特征图，这一步的参数量为$C\times C'$。故深度可分离卷积的总参数量为$h\times w\times C+C\times C'$，而传统卷积的参数量为$h\times w\times C\times C'$，计算得出深度可分离卷积方法的参数量下降为传统卷积方法的$1/C'+1/hw$。利用深度可分离卷积替代传统的卷积模型，可以有效降低网络的参数量与计算量，并且深度可分离卷积中的逐点卷积操作有效解决了分组卷积中组间特征信息屏蔽阻塞的问题。但是由于传统卷积同时考虑空间维度特征与通道维度特征，深度可分离卷积将两个维度的特征提取进行拆分然后进行序列化运算，所以在全局特征提取和特征信息融合方面的性能要弱于传统的卷积模块，所以将在网络中使用深度可分离卷积替代传统卷积模块会使网路的精度出现下降。

GSConv[63]在传统卷积方式的基础上结合深度卷积，再进行通道维度的拼接，在减少了网络提取特征的参数量和计算量同时，其对图像整体特征信息的提取和融合性能与传统卷积方法相当。所以使用 GSConv 模块来替代网络中的原始卷积模块，对网络的特征提取能力影响较小且可以大幅降低其参数量和计算量，使整体网络架构实现轻量化。GSConv 的结构如图 3-15 所示。

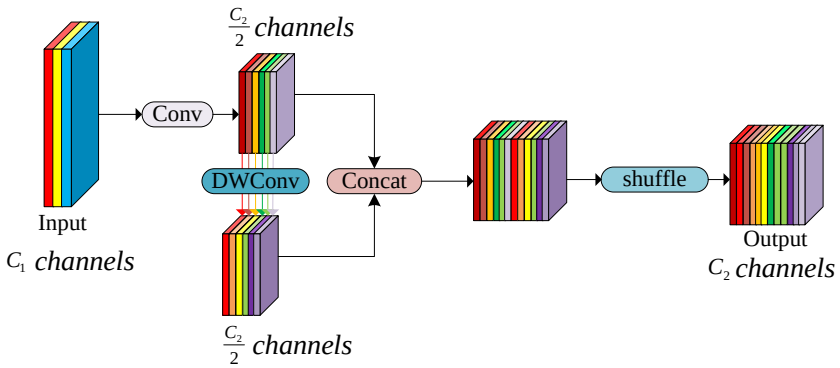


图 3-15 GSConv

如图中所示，GSConv 首先对输入特征图进行传统卷积操作来提取特征信息，将输出特征图的通道维度输出至最终输出特征图通道数的二分之一；将此阶段

输出的特征图利用深度卷积再进行通道维度的特征提取，输出一个通道维度同样为最终输出通道数二分之一的特征图；下一步将两个步骤得到的输出在通道维度进行拼接，最后将其进行通道重排序使传统卷积与深度卷积提取到的特征信息进行混合，得到最终的输出特征图。若输入特征图的尺寸为$H\times W\times C_1$，C_1为通道维度，输出特征图的尺寸为$H'\times W'\times C_2$，若使用大小为$h\times w$的卷积核进行传统卷积操作，那么参数量为$h\times w\times C_1\times C_2$；在 GSConv 模块中，若传统卷积模块的卷积核与深度卷积的卷积核大小相同且都为$h\times w$的话，那么参数量为$h\times w\times C_1\times C_2/2+h\times w\times C_2/2$，所以使用 GSConv 模块可以使参数量减少为传统卷积模块的$(1+C_1)/2C_1$。

3.3 目标检测网络轻量化改进实验

本文在 YOLOv5s 的基础上参考 GSConv 卷积模块对其进行轻量化改进，设计一种 YOLOv5s-GSConv 目标检测网络，并在公开数据集 PASCAL VOC-2007 上进行训练。其中网络模型训练的超参数如下：初始学习率为 0.01；学习率动量为 0.937，权重衰减系数为 0.0005；边框损失函数权重为 0.05；类别损失系数为 0.5；置信度损失系数为 1.0；色调设置为 0.015；饱和度设置为 0.7；亮度设置为 0.4；batch_size 设置为 64。由于本文将人设为室内场景下的默认动态物体，所以在 TUM 数据集上的表现如图 3-16 所示。

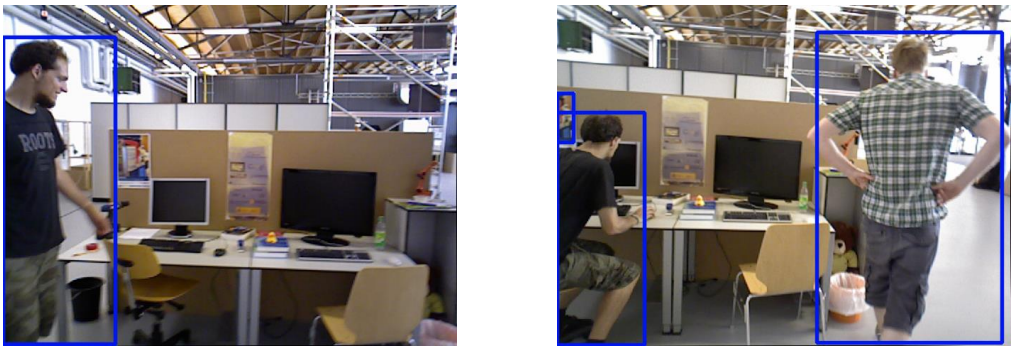


图 3-16 YOLOv5s 检测结果

使用 PASCAL VOC-2007 数据集对 YOLOv5s-GSConv 以及 YOLOv5s 进行测试，其对比指标主要为参数量、计算量、mAP、推理速度和帧率，测试结果如表 3-1 所示。本文设计的 YOLOv5s-GSConv 相比于 YOLOv5s，在推理速度和帧

率上的差距不大，其 mAP 下降了 2.3 个百分点，但是参数量减少了 26.90%，计算量减少了 35.85%。考虑到本文算法是针对室内的动态场景，所以 mAP 的下降在合理范围内。本文设计的轻量级目标检测网路 YOLOv5s-GSConv 基本可以满足 SLAM 系统的实时性需求，同时也有利于算法在移动机器人平台的移植工作。

表 3-1 目标检测网络对比

网络模型	参数量/M	计算量/GFLOPS	mAP0.5/%	推理速度/ms	帧率/FPS
YOLOv5s-GSConv	10.6	10.2	85.4	19.7	43
YOLOv5s	14.5	15.9	87.7	20.2	41

3.4 本章小结

本章分析了卷积神经网络的基本结构、YOLOv5 目标检测网络的结构以及轻量化改进方法。首先，对神经网络的原理以及基本结构进行了详细阐述，利用数学模型来对前向计算和反向传播进行建模，模拟了神经网络不断对输入数据进行学习的过程。其次，对卷积神经网络的组成部分进行了阐述，使用输出层、卷积层、降采样层、全连接层与输出层构建了一个简单的卷积神经网络并分析了各部分的工作原理。最后，对单阶段目标检测网络 YOLOv5 的组成结构进行了说明，并针对本文在室内动态场景下 SLAM 算法研究对其使用 GSConv 进行轻量化改进。

第4章 室内动态场景下 RGB-D SLAM 算法研究

视觉 SLAM 工作在动态场景下最重要的任务是对场景动态特征信息的滤除，基于单阶段目标检测网络结合几何约束来实现场景动态信息的检测和剔除，将单阶段目标检测网络 YOLOv5s 嵌入到 ORB-SLAM2 的跟踪线程中，可以有效检测到场景中的动态物体，并将这些动态物体的特征点剔除。但是只通过目标检测网络来实现动态特征的剔除，会造成特征信息的剔除过度，导致跟踪线程所需特征信息不足，造成跟踪失败。故本文设计一种深度信息随机采样一致性算法(D-RANSAC)，并结合多视图几何约束原理在目标检测网络获取的语义信息基础上来实现动态特征信息的滤除任务。

4.1 动态场景下 RGB-D 算法整体框架

本文在 ORB-SLAM2 框架的 RGB-D 模式下进行改进，来实现 SLAM 系统在动态场景下的工作需求。在 ORB-SLAM2 系统中，为了应对动态环境带来的挑战，本文提出了一种新颖的线程改进与扩展方案。如图 4-1 所示，整体系统框架在原有的基础上增加了 YOLOv5s-GSConv 目标检测网络作为独立的线程。这个新增的线程专门用于对输入的场景图像序列进行实时分析，以检测并识别出预定义的动态物体。在 ORB-SLAM2 的跟踪线程中，系统首先会对场景进行 ORB 特征点的提取。与此同时，YOLOv5s-GSConv 线程会并行工作，对同一场景图像进行动态物体的检测，并生成先验语义信息和动态物体的边界框信息。这些信息随后会被传递给跟踪线程，用于辅助特征点的处理。在跟踪线程中，系统会将提取的特征点信息与来自目标检测线程的先验语义信息进行融合。通过结合 RGB-D 相机提供的场景深度信息，系统能够利用深度信息随机采样一致性算法对特征点进行第一次筛选，有效剔除落在动态物体上的特征点。随后，再借助多视图几何约束方法，对剩余的特征点进行二次筛选，进一步排除可能的动态特征点。跟踪线程处理后的图像帧仅包含静态场景信息。这些经过处理的图像帧会被传入稠密建图线程，用于构建静态稠密点云地图。通过这种方式，SLAM 系统能够更好地适应动态环境，满足在复杂场景下稳定工作的需求。

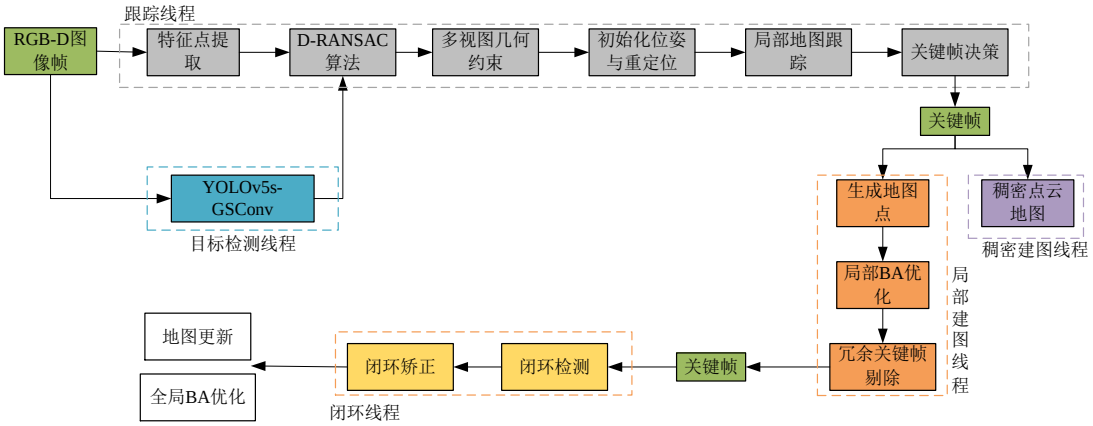


图 4-1 算法整体架构

4.2 深度信息随机采样一致性算法

4.2.1 利用深度信息改进的 RANSAC 算法

ORB-SLAM2 是基于 ORB 特征点来实现定位与建图的 SLAM 系统，其提取的特征点质量关系到整个 SLAM 系统的鲁棒性与精确性。动态特征信息会使系统在计算相机运动轨迹时产生误差漂移。在 DMS-SLAM[64]中，作者将 YOLO 目标检测网络获得的动态框内的所有特征点全部去除，来实现动态信息的筛除。这是一种相对简单的方法，但是当人占场景的三分之二以上的时候，这种方法会剔除场景中的大多数特征点，从而导致 SLAM 系统跟踪所需要的特征点数量不足，出现跟踪失败、系统的鲁棒性下降、相机估计运动轨迹漂移等问题。所以本文将目标检测线程获取的先验语义信息和 RGB-D 相机获取的场景深度信息进行拟合，设计一种深度信息随机采样一致性算法，这种方法可以准确剔除动态框内落在预定义动态物体上的特征点，保留框内的静态背景特征点。本文使用 YOLOv5s-GSConv 作为单独的线程来对输入场景图像序列提取先验语义信息和动态物体框，并将这些信息传入跟踪线程与特征点信息进行拟合，本文将人这一物体预定义为动态物体类别。深度图像(Depth Map)为 RGB-D 相机在拍摄场景彩色图像时，同时拍摄的包含场景物体与传感器之间距离信息的图像，其形式类似于灰色图像，每个像素点的值代表传感器与场景物体的真实距离。RGB-D 相机所获取的 RGB 图像与 Depth 图像是经过传感器校准过的，因此两类图像之间的像素点一一对应，如图 4-2 所示。

图 4-2 深度图像

将 Depth 图像中的场景深度信息与目标检测网络所获得的语义信息相结合，利用 D-RANSAC 算法可以筛选出落在人身上的特征点，而不会将那些位于动态物体框内的静态特征点信息误剔除，从而保留足够的特征点信息来保证位姿估计其他线程的正常工作。D-RANSAC 算法是基于 RANSAC(Random Sample Consensus)[6]算法改进而来，RANSAC 算法也被称为随机采样一致性算法，可以利用迭代的思想在一组含有外点的数据中模拟出正确的数学模型。一般在随机获取的一组数据中，将数据中的信息分为内点数据和外点数据，内点数据是拟合模型集合内的点，而外点数据是对拟合模型无效的点，RANSAC 算法可以将这些外点剔除，从而利用内点数据来使模型的拟合更加准确。一般使用最小二乘法(Least Square Method)从一组数据中拟合理想数学模型，但是最小二乘法是将包括外点数据在内的全部组内数据来进行数据的拟合，在面对从噪声范围较大的数据集中拟合模型的任务时，最小二乘法则不适合应用此类场景。因为 RANSAC 只通过筛选后的内点来拟合模型，所以在面对此类任务时，其模型拟合的准确度更高，如图 4-3 所示。

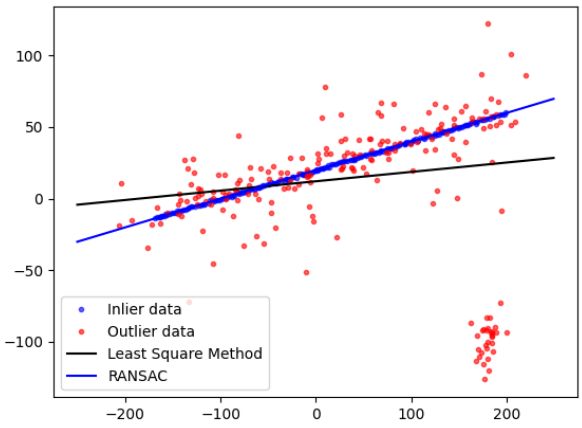


图 4-3 数学模型拟合对比

在图4-3中，黑色直线是使用最小二乘法拟合的直线模型，可以看到明显偏出了正确方向；红色直线是使用 RANSAC 算法几何的直线模型，符合期望的输出结果。本文在 RANSAC 算法的基础上，把一帧图像中由目标检测线程获得的动态物体框内的特征点数据作为整个集合，将落在动态物体上的特征点作为内点，根据内点拟合出的数学模型，将所有内点筛选出来并剔除，这样可以实现保留静态背景特征点的任务。本文 D-RANSAC 算法的使用有一些前提，首先是人需要占据 bounding box 的三分之二以上，不然会误将背景中的静态特征点剔除；其次落在人身上的特征点的深度值差异小于预设阈值，不然使用拟合模型筛选内点时可能找不到全部的内点数据；最后是人与周围背景要有明显深度差异，否则会导致拟合模型将外点筛选出来。

D-RANSAC 算法步骤如下：首先将动态框内的特征点作为一个集合 $P_{dynamic}$，然后进入循环，当前迭代次数 n 小于我们初始化的迭代次数 N_{init} 时，每一次迭代开始时在 $P_{dynamic}$ 中随机选取两个特征点，并且获得这两个点在其深度图像上的深度信息 d_1， d_2，计算 Depth Model，如式(4-1)所示。

$$\mathrm{D}_{\mathrm{mod}}=\frac{\sum_{i=1}^{N}(d_i-\bar{d})^2}{N} \tag{4-1}$$

这里的 D_{mod} 就是期望拟合的深度模型， $\bar{d}$ 表示 $P_{dynamic}$ 内所有点的深度均值，N 的含义是我们计算深度模型时随机选取的点数量，在本文中为 2。

随后遍历$P_{dynamic}$中剩余的点，计算每个点与D_{mod}的差值，当差值小于阈值Th时，将此点加入内点集，更新内点集的数量s和迭代次数n，遍历结束时，将上一次从迭代得出的内点集数量与本次迭代的内点集数量进行比较，取数量大的内点集作为暂时最佳内点集，然后进入下一次迭代。迭代完成时，输出最终的内点集合。

4.2.2 自适应迭代次数的 RANSAC 算法

传统 RANSAC 算法需要提前规定好算法所需的迭代次数，如果迭代次数过大，则会导致计算资源的浪费，若设计的迭代次数过小，则不能获得比较精确的内点集。本文在 D-RANSAC 算法中加入了动态更新迭代次数算法，细节如下：

在$P_{dynamic}$中随机选择k个点来计算数学模型，则内点在全部点中的百分比P：

$$P=\frac{x_{\mathrm{in}}}{x_{\mathrm{in}}+x_{\mathrm{out}}} \tag{4-2}$$

则k个点中至少有一个点为外点的概率为$1-P^k$。在N次迭代的条件下，每一次选择的点集中都存在离散点的概率为$(1-P^k)^N$，用P_{true}来表示从点集中随机选择的点在N次迭代中都是内点的概率，则有以下方程：

$$P_{\mathrm{ture}}=1-(1-P^k)^N \tag{4-3}$$

那么迭代次数N就是

$$N=\frac{\log(1-P_{\mathrm{true}})}{\log(1-P^k)} \tag{4-4}$$

每个点为内点的概率P是先验值，P_{true}是希望通过 RANSAC 计算得出正确数学模型的概率。RANSAC 算法开始时P的准确值是未知的，在初始化时把P设为一个足够大的值，在每次更新深度模型和内点集的时候，利用当前内点集合计算出P,由P更新迭代次数N，由此可以根据由当前计算出的深度模型所获得的内点集合，动态更新 RANSAC 迭代次数，达到更精确的输出结果和更快的计算速度。

4.3 基于几何约束的动态特征点剔除

在上一小节，利用 D-RANSAC 算法基本可以将落在人身上的特征点剔除，但是对于室内场景下仍然存在很多潜在动态物体，例如人可以将椅子、书本等物品进行挪动，本文将人这一类别预定义为动态物体，所以其他潜在动态物品在先验信息中为静态类别，所以当人对其他物品进行移动时，目标检测网络会对这些动态特征信息漏检测。所以仅仅通过目标检测网络并不足以检测场景中所有的动态物体和潜在动态物体，所以本节通过多视图几何约束的方法在 D-RANSAC 算法的基础上再对场景特征信息进行二次筛选和剔除处理。

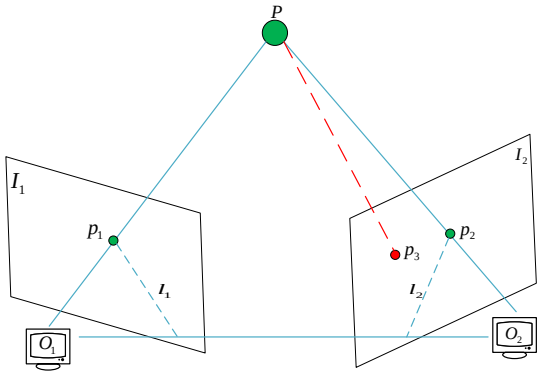


图 4-4 多视图几何

如图 4-4 所示，设摄像机的初始位置为O_1，其成像平面为I_1，经过位姿变换后的相机位置为O_2，成像平面为I_2，设位姿变换的旋转矩阵和平移向量为R，t。在两帧图像中有一组通过特征匹配得到的特征点对p_1与p_2，若特征匹配的结果准确，那么p_1与p_2都是三维空间中的点P在不同成像平面中的投影点。在多视图几何中，射线$\overrightarrow{O_1p_1}$与$\overrightarrow{O_2p_2}$会相交于点P，那么将O_1，O_2，P三点构成的平面称为极平面(Epipolar Plane)，两帧图像的相机中心连线O_1O_2被称为基线(Base Line)，极平面与两个成像平面的交线l_1与l_2被称为极线(Epipolar Line)。假设p_1的归一化坐标为$(u_1,v_1,1)$，p_2的归一化坐标为$(u_2,v_2,1)$，那么根据式(2-7)，p_1与p_2满足表达式：

$$s_1 p_1 = KP, s_2 p_2 = K(RP + t) \tag{4-5}$$

其中 K 代表了相机的内参矩阵，s_1 与 s_2 为成像平面与空间点的深度距离信息，R 和 t 表示两帧图像之间的位姿变换信息。那么可以将归一化平面定义为

$$x_1 = K^{-1} p_1, x_2 = K^{-1} p_2 \tag{4-6}$$

式(4-6)中的 x_1 与 x_2 是 p_1，p_2 在归一化平面上的坐标。将式(4-6)带入式(4-5)可得：

$$x_2 = Rx_1 + t \tag{4-7}$$

在式(4-7)两边叉乘平移向量 t 可得：

$$t \times x_2 = t \times Rx_1 \tag{4-8}$$

随后在式(4-8)两边同时左乘 x_2^T 可以得到：

$$x_2^T t \times x_2 = x_2^T t \times Rx_1 \tag{4-9}$$

因为向量 $t \times x_2$ 与 t 和 x_2 都成垂直关系，所以式(4-9)又可以表示为：

$$x_2^T t \times Rx_1 = 0 \tag{4-10}$$

将式(4-6)带入式(4-10)可以得到：

$$p_2^T K^{-T} t \times RK^{-1} p_1 = 0 \tag{4-11}$$

在多视图几何理论中，定义基础矩阵 $F = K^{-T} t \times RK^{-1}$，那么在理想情况下，两帧图像中匹配的特征点对 p_1，p_2 应该满足以下约束：

$$p_2^T F p_1 = \begin{bmatrix} u_2 & v_2 & 1 \end{bmatrix} F \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = 0 \tag{4-12}$$

基础矩阵 F 可以表达一个空间点在两帧图像间的投影关系，本文利用 RANSAC 算法来找到基础矩阵的最优解。理想状态下，若空间点 P 为静态特征点，那么其在 I_1 上的投影点 p_1 对应在 I_2 上的投影点 p_2 会落在极线 l_2 上，但是由于相机畸变和基础矩阵的误差，所以 p_2 与极线 l_2 的距离在一定阈值范围内也会被认为是静态特征。若点 P 为动态特征点，则 P 投影在平面 I_2 上的实际位置

可能为 p_3，那么 p_3 与极线 l_2 的距离则会超出阈值。本文利用这一理论来实现特征点静动态的区分。极线 l_2 可以利用基础矩阵 F 来表示：

$$l_2 = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = Fp_1 = F \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} \tag{4-13}$$

那么点 p_1 对应在平面 I_2 上的投影点 p_2 到极线 l_2 的距离 D 就可以表示为：

$$D = \frac{p_2^T F p_1}{\sqrt{X^2 + Y^2}} \tag{4-14}$$

若 D 小于预设的距离阈值，则认为该点为静态特征点，否则会识别为动态特征点并将其滤除。

4.4 动态特征点滤除实验

本节在 TUM RGB-D 数据集 Dynamic Objects 类别中的 walking_xyz 序列中评估本文算法的动态特征点滤除性能，并与本文的基础算法 ORB-SLAM2 以及 YOLOv5s-GSConv 结合 ORB-SLAM2 进行对比试验。图 4-5 为 ORB-SLAM2 在面对动态场景时的特征点提取表现，可以看到场景中的动态物体主要为人，ORB-SLAM2 将人身上的特征信息认为是正常特征点，所以这些特征点也会参与位姿估计与后端优化，从而会导致相机估计运动轨迹的漂移。图 4-6 反映的是 ORB-SLAM2 结合轻量化目标检测网络 YOLOv5s-GSConv，将目标检测线程获取的动态物体框内的所有特征点进行剔除的情况。可以看到 YOLOv5s-GSConv 可以准确地获取人这一类别的物体框，并将这一类别识别为动态物体并将框内的所有特征信息进行剔除。此类筛除动态特征的方法较为简单，但是存在剔除过度的问题。当动态物体占据场景中的大部分空间时，暴力滤除动态物体框内的所有特征信息，会导致框内一些有用的静态特征信息丢失，造成后续位姿估计等线程所需特征信息不足，最终引起跟踪失败等问题。图 4-7 为本文在 ORB-SLAM 结合 YOLOv5s-GSConv 的基础上，在跟踪线程中加入 D-RANSAC 算法和多视图几何约束方法来对场景动态信息进行滤除的方案。与图 4-6 对比可以明显看出，本文算法将动态物体框内落在人身上的特征点全部检测出并滤除，将框内的静态背景特征信息保留下来，图中红色特征点为动态物体框内静态特征

点。这样本文算法后续的位姿估计等其他线程所使用的特征信息全部为静态特征点，并且不会出现所需特征信息量过少的情况。以上分析情况表明，利用 YOLOv5s-GSConv 将 SLAM 系统的输入场景进行检测提取动态物体框信息，再利用 D-RANSAC 算法结合多视图几何约束对动态物体框内的特征信息进行检测和剔除，可以准确滤除落在动态物体上的特征信息并保留静态特征信息。

图 4-5 ORB-SLAM2 特征点提取

图 4-6 YOLOv5s+ORB-SLAM2 特征点提取

图 4-7 本文算法特征点提取

4.5 本章小结

本章分析了 RANSAC 算法的原理，在此基础上设计一种 Depth-RANSAC 算法，并结合多视图几何约束方法来实现对场景动态特征信息的剔除，解决了只依靠目标检测网络造成的特征信息剔除过度的问题。本章研究了 RANSAC 算法

的原理并给出 Depth-RANSAC 的算法步骤，在此基础上改进了自适应迭代次数方法。对于场景中少量的潜在动态物体，利用多视图几何约束原理来对其进行筛选和剔除。最后通过实验证明本文的特征信息剔除方法可以有效剔除场景中的动态特征信息并保留大多数静态背景信息。

第5章 室内动态场景下 RGB-D SLAM 算法实验与分析

本文首先对单阶段目标检测网络 YOLOv5s 进行轻量化改进，并将其作为单独线程加入 ORB-SLAM2 系统中以获取场景中的先验语义信息；其次本文结合语义信息和深度图像设计一种深度信息随机采样一致性算法来筛选场景中的动态特征点并剔除，再结合多视图几何约束来进行动态特征二次筛除。本章在相关实验平台中对本文算法进行实验分析，以此验证本文算法的实际性能。本章进行的实验主要分为三部分，第一部分为改进的轻量化目标检测网络 YOLOv5s-GSConv 与原网络之间的性能指标对比，验证其轻量化改进的有效性；第二部分为 D-RANSAC 结合几何约束对动态特征信息的筛除实验；第三部分是将算法在数据集中进行整体定位评估，并与传统 SLAM 算法和主流动态 SLAM 算法进行对比，已验证本文算法在动态环境下的工作性能。

5.1 实验平台与数据集介绍

5.1.1 实验平台介绍

本文的实验硬件平台是一台 CPU 为 Intel i7-8700，GPU 为 NVIDIA GeForce GTX2080Super，显存为 8GB，内存为 64GB，操作系统为 Ubuntu18.04 的计算机；软件平台方面，配备了 CUDA10.1 和 cudnn7 框架来加速 GPU 运算，深度学习库为 Pytorch1.5.1，此外视觉 SLAM 系统运行所需的依赖库还有 Opencv3.4.1、Eigen、Pangolin 等。

5.1.2 数据集介绍

TUM RGB-D 数据集是由德国慕尼黑大学的计算机视觉实验室提出的公开数据集，其 RGB 图像和 Depth 图像都是由 Microsoft 旗下的 Kinect 体感相机在室内环境下进行拍摄。TUM RGB-D 数据集由 39 个序列组成，被分为了测试、手持传感器 SLAM、移动机器人 SLAM、结构与低纹理、动态物体、三维场景重建、验证数据集、标定文件等不同任务场景下的不同类型数据集。数据集中的每个序列都包含有 rgb 和 depth 两个文件夹，rgb 文件夹是当前序列的彩色图像，depth 文件夹是当前序列传感器所获取的深度图像。因为 RGB 图像与 Depth 图像

的采集任务是独立完成的，所以在使用数据集之前需要利用 TUM 提供的 associate.py 程序使两类图像之间根据时间戳对齐。序列目录下的 rgb.txt 和 depth.txt 包含了每张图像采集的时间戳和文件名，groundtruth.txt 包含了外部运动捕捉传感器采集到的真实相机运动轨迹，可以使用官方评估软件结合 groundtruth.txt 文件对本文算法的真实性能进行评估。

本文研究的是室内动态场景下的 RGB-D SLAM 算法，所以使用 TUM 下的 Dynamic Objects 数据集序列对本文算法进行验证。Dynamic Objects 序列中分为 desk_with_person、sitting、walking 三种类别，desk_with_person 记录的是人位于办公桌前与办公物品互动的场景；sitting 类别记录了两个人坐在办公桌前进行简单互动的场景，这一类别主要验证 SLAM 算法在场景中包含少量静态特征信息工作的性能；walking 类别主要记录的使两个人在办公桌前进行大量并快速的移动，主要考验算法对于场景中包含大量且快速移动动态物体的鲁棒性。其中 sitting 和 walking 类别分别有四种后缀，为 xyz、rpy、halfsphere 和 static，这代表了传感器在拍摄图像时的运动方式。xyz 表示相机沿 xyz 轴平移运动；static 表示相机为静止状态；rpy 表示相机沿 xyz 轴进行旋转运动；halfsphere 表示相机沿半球轴运动。

5.1.3 算法精度评价指标

本文用于评价 SLAM 算法的指标为绝对轨迹误差(Absolute Trajectory Error, ATE)与相对位姿误差(Relative Pose Error, RPE)，利用这两种指标可以对 SLAM 算法的定位精度做定量分析。相对位姿误差主要描述的是一定时间戳内估计位姿与真实位姿的误差，相对位姿误差还包含了两种误差，分别是旋转误差和平移误差。绝对轨迹误差是全局的估计轨迹与真实轨迹之间的差值，直接表现算法的精度和估计轨迹的全局一致性。其中每种指标都有其评估参数，包括最大值（Max）、最小值（Min）、均方根误差（Root Mean Squared Error，RMSE）、标准误差（Standard Deviation，STD）、平均值（Mean）、中值（Median）等，这些参数共同衡量了指标的优劣。本文使用 ATE 和 RPE 两种指标以及结合均方根误差和标准误差来衡量本文算法精度并与其他算法进行对比实验。

5.2 定位精度实验与分析

本文算法是基于 ORB-SLAM2 的框架进行研究的，主要对其进行了线程添加和线程改进。在单阶段目标检测网络 YOLOv5s 的基础上进行轻量化改进，设计一种 YOLOv5s-GSConv 网络，并将其作为单独线程与 ORB-SLAM2 结合以获取场景中的动态物体框信息。在 RANSAC 算法的基础上，设计出 D-RANSAC 算法将动态物体框内的特征信息进行筛选并剔除，再结合多视图几何约束方法对动态特征信息进行二次筛除，以实现本文算法在动态场景中的工作性能。为了验证本文改进的算法相对于 ORB-SLAM2 的提升，本节在 TUM 数据集上对本文算法与 ORB-SLAM2 的定位精度进行定量分析对比，其定位精度的提升程度使用式(5-1)表示：

$$\xi = \frac{\alpha - \beta}{\alpha} \times 100\% \tag{5-1}$$

在上式中，ξ 代表算法定位精度提升率，α 表示 ORB-SLAM2 的定位精度误差，β 表示本文算法的定位精度误差。

表 5-1~表 5-3 反映了本文改进算法相比于 ORB-SLAM2 在全局估计轨迹精度、局部估计轨迹的平移与旋转精度方面的对比，其中所有数值单位为米。由于算法每次对于场景 ORB 特征点的提取是随机的，所以每次的匹配结果都不相同，导致定位精度误差也不同。所以本文在对于数据集中每一序列进行实验时，会进行 5 次实验并取平均值作为最终的实验结果，以去除实验中的噪声影响。

表 5-1 绝对轨迹误差对比结果

序列	ORB-SLAM2		DS-SLAM		本文算法		定位精度提升率	
	RMSE	STD	RMSE	STD	RMSE	STD	RMSE	STD
walking_xyz	0.7672	0.4395	0.0247	0.0161	0.0176	0.0106	97.71%	97.59%
walking_rpy	0.8383	0.4692	0.4659	0.2501	0.3198	0.1586	61.85%	66.20%
walking_static	0.3921	0.1634	0.0081	0.0036	0.0103	0.0068	97.37%	95.84%
walking_half	0.5625	0.2927	0.0333	0.0179	0.0277	0.0151	95.08%	94.84%
sitting_xyz	0.0130	0.0062	0.0102	0.0052	0.0127	0.0058	2.31%	6.45%
sitting_static	0.0081	0.0038	0.0065	0.0033	0.0066	0.0032	18.52%	15.79%

表 5-2 相对轨迹误差平移部分对比结果

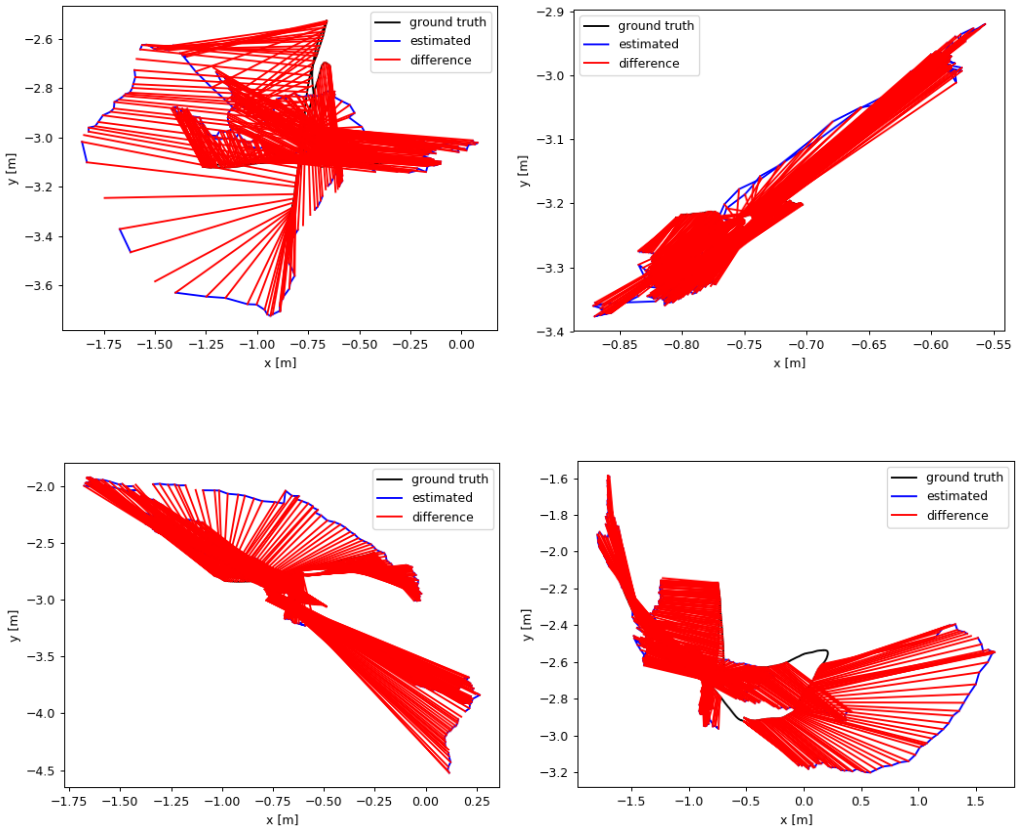
序列	ORB-SLAM2		DS-SLAM		本文算法		定位精度提升率	
	RMSE	STD	RMSE	STD	RMSE	STD	RMSE	STD
walking_xyz	0.0775	0.0739	0.0155	0.0109	0.0119	0.0075	84.64%	89.85%
walking_rpy	0.1007	0.0976	0.0251	0.0172	0.0197	0.0125	84.44%	87.19%
walking_static	0.0141	0.0141	0.0058	0.0036	0.0085	0.0063	39.72%	55.32%
walking_half	0.0342	0.0276	0.0151	0.0100	0.0142	0.0093	58.48%	66.30%
sitting_xyz	0.0184	0.0099	0.0088	0.0050	0.0098	0.0048	46.74%	51.52%
sitting_static	0.0047	0.0029	0.0047	0.0029	0.0042	0.0027	10.64%	6.90%

表 5-3 相对轨迹误差旋转部分对比结果

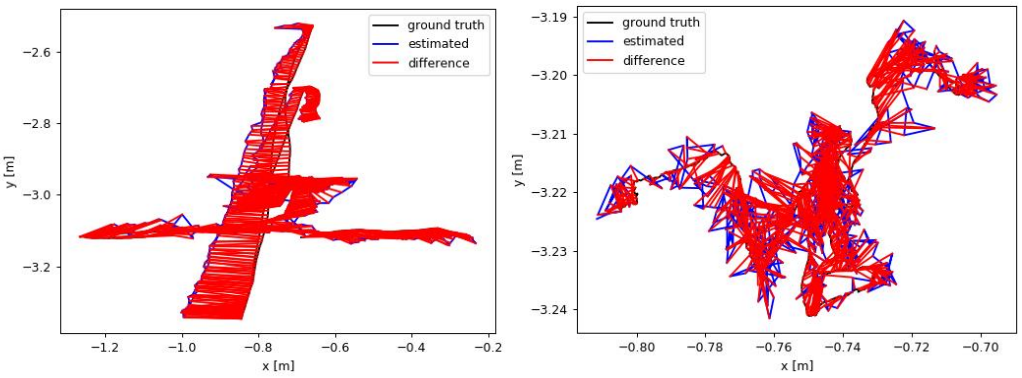
序列	ORB-SLAM2		DS-SLAM		本文算法		定位精度提升率	
	RMSE	STD	RMSE	STD	RMSE	STD	RMSE	STD
walking_xyz	10.6155	5.7296	1.6821	1.2397	0.5514	0.1986	94.81%	96.53%
walking_rpy	13.7847	9.4737	4.1923	3.5656	3.0042	0.3056	78.21%	96.77%
walking_static	8.9133	6.0383	0.0059	0.0033	0.3104	0.1102	96.52%	98.17%
walking_half	9.3693	5.9705	1.0553	0.5518	0.8269	0.2887	91.17%	95.16%
sitting_xyz	0.5835	0.2611	0.6302	0.3297	0.5494	0.2557	5.84%	2.07%
sitting_static	0.2735	0.1215	0.2735	0.1215	0.2486	0.0032	9.10%	97.37%

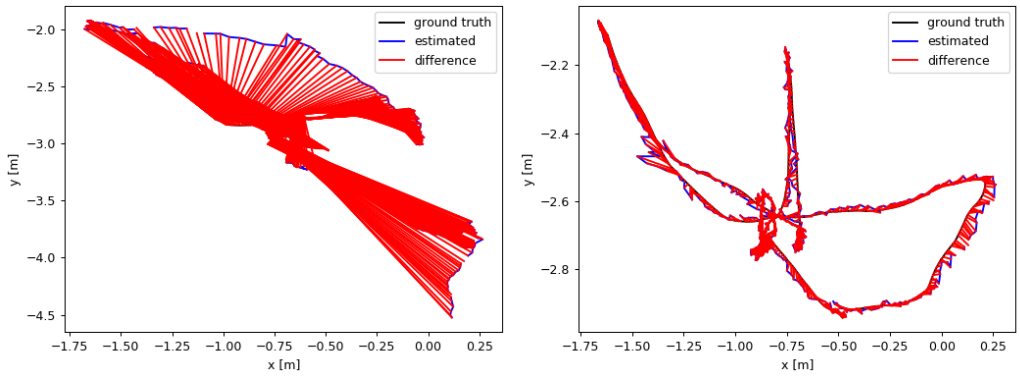
本文使用 Dynamic Objects 序列中的 5 个序列对本文算法、ORB-SLAM2 以及优秀动态场景开源 SLAM 算法 DS-SLAM 进行定位评估实验，其中 sitting_xyz/static 为低动态场景，walking_xyz/rpy/half/static 为高动态场景。表 5-1 为本文算法与 ORB-SLAM2 在绝对轨迹误差方面的对比结果，可以看出，在高动态序列 walking_xyz/half/static 中，本文算法相比较 ORB-SLAM2 在定位精度方面的 RMSE 平均提升率达到了 96.72%，结果表明本文改进算法相比较原始算法，在高动态场景下的定位精度得到了显著提升。但是在高动态序列 walking_rpy 中的定位精度提升率为 61.85%，不如其他高动态序列，这是因为在此序列中，相机围绕 xyz 轴做快速旋转运动，快速的相机运动会导致 SLAM 系统跟踪失败，造成估计的运动轨迹误差变大，定位精度下降等问题。因为 ORB-SLAM2 的理想工作场景为静态环境，并且系统中使用 RANSAC 算法将少量动态特征点作为外点剔除，所以在低动态场景 sitting_xyz/static 序列中，ORB-SLAM2 的定位精度表现良好，但本文算法在 RMSE 的平均提升率也有 10.42%。在与 DS-SLAM 算法的对比中，在相同序列下本文算法取得了与 DS-SLAM 相近的定位精度结果，甚至在某些序列的表现要由于 DS-SLAM。但是总体表现还是略差于 DS-SLAM，这是因为 DS-SLAM 使用的是像素级的语义分割网络 SegNet 来获取场景语义信息，其准确度更高，但是运行速度则下降。结果表明本文利

用 YOLOv5s-GSConv 网络检测场景动态信息，再利用 D-RANSAC 算法和多视图几何约束方法剔除动态信息的方法，可以有效在动态场景中提升算法定位精度。

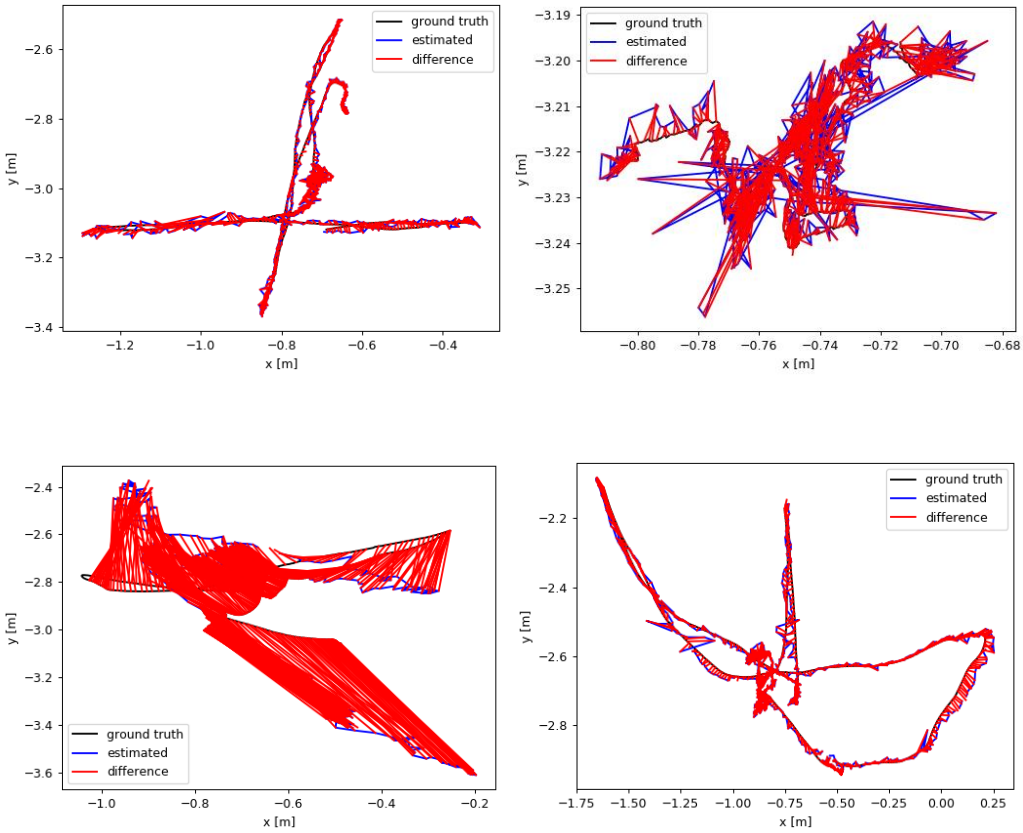


(a)ORB-SLAM2





(b)DS-SLAM



(c)本文算法

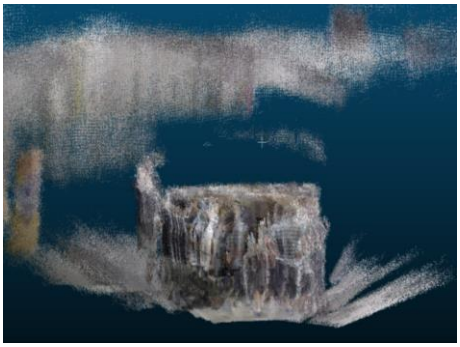
图 5-1 估计轨迹和真实轨迹比较结果

图 5-1 为本文算法、ORB-SLAM2 以及 DS-SLAM 在高动态序列 walking_xyz/static/rpy/half 上的估计轨迹与真实轨迹对比图。图中黑色轨迹为 ground truth，表示为相机的真实运动轨迹；蓝色轨迹称为 estimated，为算法估计的相机运动轨迹；红色部分为 difference，代表估计轨迹与真实轨迹之间的偏差。红色部分面积越大，则表示算法的定位精度越差，红色部分越小就代表算

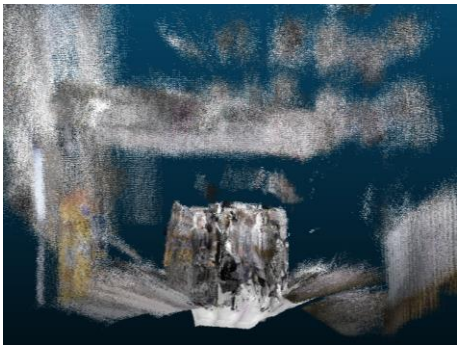
法的定位精度越好。(a)图代表 ORB-SLAM2 的轨迹误差情况，可以看出在高动态场景下，ORB-SLAM2 对于相机运动轨迹的估计误差较大，其算法定位精度受到动态物体的影响较为明显。图(b)与图(c)则是本文算法与 DS-SLAM 在相同序列中的表现，可以明显看出优于 ORB-SLAM2 的表现。在 walking_static/rpy/half 三个序列中，本文算法与 DS-SLAM 的表现相近，而在 walking_xyz 序列中本文算法的红色部分明显小于 DS-SLAM 取得的结果，故表现由于 DS-SLAM。以上结果与表 5-1 的实验结果对应，表明本文算法在动态场景下对于相机运动轨迹的估计对比于原始 ORB-SLAM2 算法有明显改善，与目前主流动态 SLAM 算法表现相当，符合本文的算法设计目标。

5.3 稠密点云地图构建实验

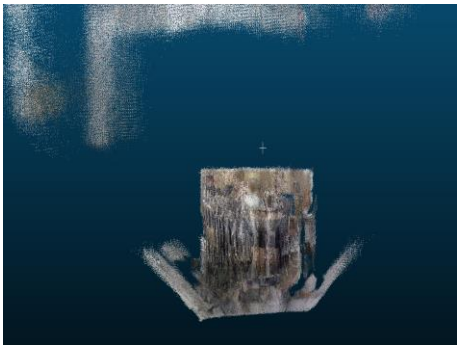
本文在 ORB-SLAM2 的基础上加入稠密建图线程并于本文改进算法在 TUM 数据集中的动态序列上进行实验对比。图 5-2 是 ORB-SLAM2 在动态序列 walking_xyz/rpy/static/half 上的稠密建图结果，由于本文研究的为 SLAM 算法在室内动态场景下的表现，所以场景中的动态物体一般为人。可以看到 ORB-SLAM2 的建图结果中有很多人的“重影”，这是因为人在场景中不停地移动，ORB-SLAM2 将人身上的特征点也作为三维地图点参与到地图的构建中，所以当点云地图中包含了这些动态的信息时，会使建图效果不佳。



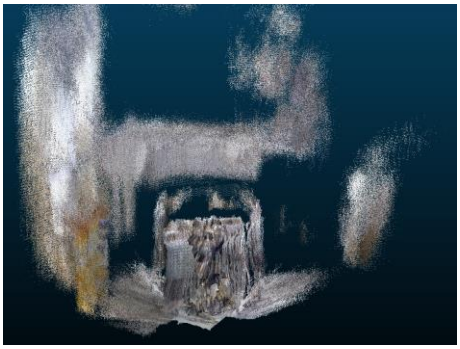
(a)walking_xyz



(b)walking_rpy



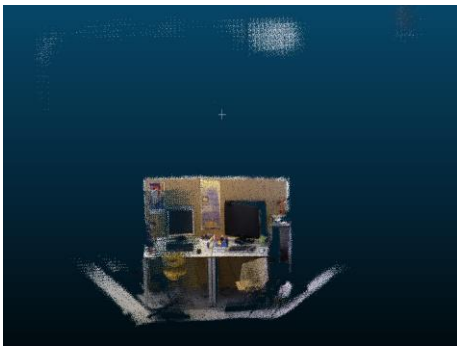
(c)walking_static



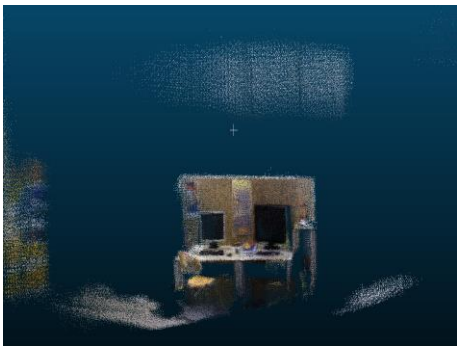
(d)walking_half

图 5-2 ORB-SLAM2 稠密点云地图

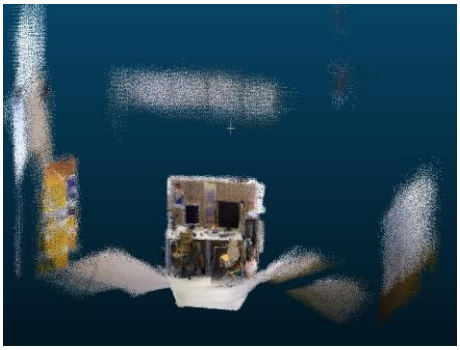
图 5-3 为本文算法稠密建图结果，稠密建图线程所接受的特征信息是经过滤除的静态信息，所以点云地图中不包含动态物体信息。可以看到，图 5-3 中已经基本没有人的“重影”，只保留了静态背景信息。总结来说，本文改进算法相比于 ORB-SLAM2，在动态场景下的建图效果得到了提升，可以构建周围环境的静态稠密点云地图，达到了本文对于算法改进的期望效果。



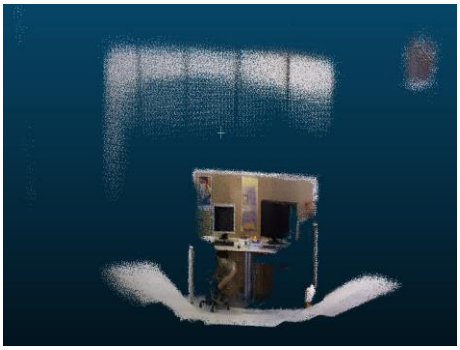
(a)walking_xyz



(b)walking_rpy



(c)walking_static



(d)walking_half

图 5-3 本文算法稠密点云地图

5.4 本章小结

本章对室内动态场景下基于深度学习的 RGB-D SLAM 算法在实验平台上进

行仿真实验，主要在定位精度与稠密建图两个方面对本文算法进行有效性评估。第一节介绍了实验平台的软硬件设施以及数据集；第二节对本文算法以及原算法和主流动态 SLAM 算法进行定位精度对比，结果表明本文算法有效提升了动态场景下的定位精度；第三节展示了本文算法与 ORB-SLAM2 的稠密建图结果，表明本文算法可以有效剔除场景动态信息，构建静态环境三维地图。

第6章　总结与展望

6.1 总结

本文首先对于近年国内外视觉 SLAM 的发展做了概述，传统视觉 SLAM 的理想工作场景为静态场景，这导致当场景中出现大量动态物体时，会造成这些动态特征点被使用在位姿估计和地图构建中，导致最终的运动轨迹估计误差较大。因此动态 SLAM 系统称为近年来的热门研究方向。随着机器学习的发展，利用卷积神经网络结合 SLAM 系统称为动态 SLAM 的理想解决方案。由于 SLAM 算法的实际运行需要很高的实时性，所以有学者利用单阶段目标检测网络 YOLO 来获取场景中的动态物体框并将框内的所有特征点进行剔除，以此达到动态特征点滤除的效果，这种简单的做法会造成滤除过度的问题，导致后续系统工作所需特征信息过少从而造成最终的定位精度误差。针对上述问题，本文在 ORB-SLAM2 的基础上进行改进，针对室内动态场景下的 RGB-D SLAM 算法开展研究，具体工作如下：

(1)在 ORB-SLAM2 的基础上新增目标检测线程来获取场景动态物体框信息，目前 YOLOv5s 网络模型参数量过大，无法满足 SLAM 系统实际工作的实时性需求。本文参考 GSConv 卷积模块对 YOLOv5s 进行轻量化改进，设计出一种 YOLOv5s-GSConv 模型，并将其作为单独的目标检测线程嵌入到 ORB-SLAM2 中。

(2)为了解决直接剔除目标检测线程获取的动态框内的所有特征点所造成的滤除过度的问题，本文参考 RANSAC 算法并结合 RGB-D 相机所获取的场景深度图像，设计出一种 D-RANSAC 算法来对动态物体框内的特征信息进行检测和筛除。由于本文将人这一类别设为默认动态物体，所以 D-RANSAC 算法会筛选出落在人身上的特征点作为内点并将其剔除，这样就将框内的静态背景特征信息保留下来参与后续的位姿估计等工作。

(3)使用多视图几何约束方法来进行动态信息二次滤除。使用目标检测检测网络来获取场景中的动态物体信息，只能将具有自主移动能力的物体设为预定义的动态类别，以此来对物体上的特征点进行筛选和剔除处理。但是对于场景

中的潜在运动物体，例如椅子、书本等可以被动地进行运动的物体，无法使用目标检测网络将其设为默认动态物体。本文使用多视图几何约束的方法，通过判断投影点与极线的距离与预设阈值之间的大小关系，来检测特征点是否为动态特征点或者误匹配点，并将筛选出的外点剔除。

(4)在公开数据集 TUM 上对本文算法进行有效性验证。试验结果表明，本文的改进算法相比较 ORB-SLAM2，在动态环境下具有更好的定位精度表现，相机估计运动轨迹与真实运动轨迹的误差更小，尤其在高动态序列中，其在绝对轨迹误差的 RMSE 和 STD 上的平均提升率达到 96.72%与 96.62%。在与目前优秀的开源动态 SLAM 算法 DS-SLAM 对比中，本文算法的表现与之相当，在部分序列中的表现更加优秀。

6.2 展望

本文在 ORB-SLAM2 的基础上进行改进，使其在动态环境中的表现得到了一定提升，但是本文算法在以下几个方面仍然有优化空间：

(1)本文为了满足 SLAM 算法使用工作中的实时性需求，对 YOLOv5s 进行轻量化改进。但是并未将本文算法部署到移动机器人平台进行实验，后续工作将对本文算法的移植性进行评估，将其部署在实际移动机器人中进行实际工作评估。

(2)本文未进行动态场景下的三维语义建图研究，使用目标检测网络无法进行像素级的语义信息获取，故无法在点云地图中将语义信息进行映射。后续计划将对建图线程单独嵌入轻量化语义分割网络来实现三维语义地图的构建。

(3)本文使用公开数据集来对本文算法进行有效性验证，并没有对实际场景进行数据采集并验证，原因是目前本文还没有完成在 ROS 平台上的移植工作，后续将继续完成此项工作，并将其移植到实物平台中进行验证。

参考文献

[1] 李延真, 石立国, 徐志根等. 移动机器人视觉 SLAM 研究综述[J].智能计算机与应用,2022,12(07):40-45.

[2] 吴涛. 用于移动机器人的视觉 SLAM 综述[J].数据通信,2022,(01):48-51.

[3] Durrant-Whyte H, Bailey T. Simultaneous localization and mapping: part I[C]// IEEE Robotics & Automation Magazine, 2006, 13(3): 108-117.

[4] 刘铭哲, 徐光辉, 唐堂等. 激光雷达 SLAM 算法综述[J].计算机工程与应用,2024,60(01):1-14.

[5] Yuan S, Wang H, Xie L. Survey on localization systems and algorithms for unmanned systems[J]. Unmanned Systems, 2021,9(02):129–163.

[6] Fischler, Bolles, Foley. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography [J]. Communications of the Association for Computing Machinery, 1981,24(6):381-395.

[7] Su P, Luo S, Huang X. Real-time dynamic SLAM algorithm based on deep learning[J]. IEEE Access, 2022, 10: 87754-87766.

[8] Davison A J, Reid I D, Molton N D, etc. MonoSLAM: real-time single camera SLAM[J]. IEEE transactions on pattern analysis and machine intelligence, 2007, 29(6):1052-1067.

[9] McGee L A. Discovery of the Kalman filter as a practical tool for aerospace and industry[M]. National Aeronautics and Space Administration, 1985.

[10] KLEIN G, MURRAY D. Parallel tracking and mapping for small AR workspaces[C]//2007 6th IEEE and ACM international symposium on mixed and augmented reality. IEEE, 2007:225-234.

[11] Li Y, Fan S, Sun Y, etc. Bundle adjustment method using sparse BFGS solution[J]. Remote Sensing Letters, 2018,9(8):789-798.

[12] MUR-ARTAL R, MONTIEL J M M, TARDOS J D. ORB-SLAM: A Versatile and Accurate Monocular SLAM system[J]. IEEE Transactions on Robotics, 2015,31(5):1147-1163.

[13] Rublee E, Rabaud V, Konolige K, etc. ORB: An efficient alternative to SIFT or SURF[C]//2011 IEEE International Conference on Computer Vision (ICCV), 2011:2564-2571.

[14] Rosten, Porter, Drummond. Faster and better: A machine learning approach to corner detection[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32(01): 105-119.

[15] 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.

[16] Campos C, Elvira R, Rodríguez J J G, etc. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam[J]. IEEE Transactions on Robotics, 2021,37(6):1874-1890.

[17] Saputra M R U, Markham A, Trigoni N. Visual SLAM and structure from motion in dynamic environments: A survey[J]. ACM Computing Surveys (CSUR), 2018, 51(2): 1-36.

[18] Xu G, Yu Z, Xing G, etc. Visual odometry algorithm based on geometric prior for dynamic environments[J]. The International Journal of Advanced Manufacturing Technology, 2022, 122(1): 235-242.

[19] Sun Y, Liu M, Meng M Q H. Motion removal for reliable RGB-D SLAM in dynamic environments[J]. Robotics and Autonomous Systems, 2018, 108: 115-128.

[20] Wang R, Wan W, Wang Y, etc. A new RGB-D SLAM method with moving object detection for dynamic indoor scenes[J]. Remote Sensing, 2019, 11(10): 1143.

[21] Li A, Wang J, Xu M, etc. DP-SLAM: A visual SLAM with moving probability towards dynamic environments[J]. Information Sciences, 2021, 556: 128-142.

[22] Fang Y, Dai B. An improved moving target detecting and tracking based on optical flow technique and kalman filter[C]//2009 4th International Conference on Computer Science & Education. IEEE, 2009: 1197-1202.

[23] Liu W, Anguelov D, Erhan D, etc. Ssd: Single shot multibox detector[C]//Computer Vision–ECCV 2016: 14th European Conference, 2016: 21-37.

[24] Mao Q C, Sun H M, Liu Y B, etc. Mini-YOLOv3: real-time object detector for embedded applications[J]. Ieee Access, 2019, 7: 133529-133538.

[25] Zhang L, Wei L, Shen P, etc. Semantic SLAM based on object detection and improved octomap[J]. IEEE Access, 2018, 6: 75545-75559.

[26] Bescos B, Fácil J M, Civera J, etc. DynaSLAM: Tracking, mapping, and inpainting in dynamic scenes[J]. IEEE Robotics and Automation Letters, 2018, 3(4): 4076-

4083.

[27] He K, Gkioxari G, Dollár P, etc. Mask r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2961-2969.

[28] Yu C, Liu Z, Liu X J, etc. DS-SLAM: A semantic visual SLAM towards dynamic environments[C]//2018 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2018: 1168-1174.

[29] 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.

[30] Cheng J, Wang Z, Zhou H, etc. DM-SLAM: A feature-based SLAM system for rigid dynamic scenes[J]. ISPRS International Journal of Geo-Information, 2020, 9(4): 202.

[31] Yuan X, Chen S. Sad-slam: A visual slam based on semantic and depth information[C]//2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020: 4930-4935.

[32] Fan Y, Zhang Q, Tang Y, etc. Blitz-SLAM: A semantic SLAM in dynamic environments[J]. Pattern Recognition, 2022, 121: 108225.

[33] Dvornik N, Shmelkov K, Mairal J, et al. Blitznet: A real-time deep network for scene understanding[C]//Proceedings of the IEEE international conference on computer vision. 2017: 4154-4162.

[34] Zhang X, Rad A B, Wong Y K. Sensor fusion of monocular cameras and laser rangefinders for line-based simultaneous localization and mapping (SLAM) tasks in autonomous mobile robots[J]. Sensors, 2012,12(1): 429-452.

[35] 高翔, 张涛, 刘毅, 等. 视觉 SLAM 十四讲[M]. 北京: 电子工业出版社, 2017: 19.

[36] Kovács A, Szirányi T. Improved harris feature point set for orientation-sensitive urban-area detection in aerial images[J]. IEEE Geoscience and Remote Sensing Letters, 2012, 10(4): 796-800.

[37] Rosten E, Drummond T. Machine learning for high-speed corner detection[C]//Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, 2006: 430-443.

[38] Lowe D G. Object recognition from local scale-invariant features[C]//Proceedings of the seventh IEEE international conference on computer vision. Ieee, 1999, 2: 1150-1157.

[39] Bay H, Ess A, Tuytelaars T, etc. Speeded-Up Robust Features (SURF)[J]. Computer Vision & Image Understanding, 2008, 110(3):346-359.

[40] Calonder M, Lepetit V, Strecha C, etc. Brief: Binary robust independent elementary features[C]//Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, 2010: 778-792.

[41] Aumüller M, Bernhardsson E, Faithfull A. ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms[J]. Information Systems, 2020, 87: 101374.

[42] Longuet-Higgins H C. A computer algorithm for reconstructing a scene from two projections[J]. Readings in Computer Vision, 1987,293(5828):61-62.

[43] Lepetit V, Moreno-Noguer F, Fua P. EPnP: An Accurate O(n) Solution to the PnP Problem[J]. International Journal of Computer Vision, 2009,81(2):155.

[44] Sharp G C, Lee S W, Wehe D K. ICP registration using invariant features[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(1): 90-102.

[45] Sturm J, Engelhard N, Endres F, etc. A benchmark for the evaluation of RGB-D SLAM systems[C]//2012 IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2012:573-580.

[46] Lecun L, Bottou L, Bengio Y, etc. Gradient-based Learning Applied to Document Recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[47] Shao H, Wang S. Deep Classification with Linearity-Enhanced Logits to Softmax Function[J]. Entropy. 2023,25(5):727.

[48] Nabati R, Qi H. Rrpn: Radar region proposal network for object detection in autonomous vehicles[C]//2019 IEEE International Conference on Image Processing (ICIP). IEEE, 2019: 3093-3097.

[49] Girshick R, Donahue J, Darrell T, etc. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2014: 580-587.

[50] Girshick R. Fast r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2015: 1440-1448.

[51] Redmon J, Divvala S, Girshick S, etc. You Only Look Once: Unified, Real-time Object Detection[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016:779-788.

[52] Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.

[53] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[C]//International conference on machine learning. pmlr, 2015: 448-456.

[54] 李泽欣.基于改进 YOLOv4 的图像目标检测算法研究[D].山西大学,2023.

[55] He K. M, Zhang X.Y, Ren S. Q, etc. Deep Residual Learning for Image Recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016:770-778.

[56] Zhu X, Lyu S, Wang X, etc. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2021: 2778-2788.

[57] Wang C. Y, Liao H. Y. M, Wu Y. H, etc. CSPNet: A New Backbone that Can Enhance Learning Capability of CNN[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020:390-391.

[58] Liu S, Qi L, Qin H, etc. Path aggregation network for instance segmentation[C]// IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018:8759-8768.

[59] Howard A, Sandler M, Chu G, etc. Searching for Mobilenetv3[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019:1314-1324.

[60] Chollet F. Xception: Deep learning with depthwise separable convolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 1251-1258.

[61] Ismail Fawaz H, Lucas B, Forestier G, etc. Inceptiontime: Finding alexnet for time series classification[J]. Data Mining and Knowledge Discovery, 2020, 34(6): 1936-1962.

[62] Harjoseputro Y, Yuda I, Danukusumo K P. MobileNets: Efficient convolutional neural network for identification of protected birds[J]. IJASEIT (International Journal on Advanced Science, Engineering and Information Technology), 2020, 10(6): 2290-2296.

[63] Zhao Z, Guo G, Zhang L, etc. A new anti-vibration hammer rust detection algorithm based on improved YOLOv7[J]. Energy Reports, 2023, 9: 345-351.

[64] Liu G, Zeng W, Feng B, etc. DMS-SLAM: A General Visual SLAM System for

Dynamic Scenes with Multiple Sensors[J]. Sensors (Basel, Switzerland), 2019, 19(17):1-20.

攻读硕士期间取得的学术成果

一、读研期间已发表/录用的学术论文

[1]张开平,吴桐. 基于目标检测与深度信息关联的 RGB-D SLAM 算法[J]. 哈尔滨商业大学学报(自然科学版)

[2]吴桐,张开平. 基于动态点线耦合的单目视觉惯性 SLAM 算法[J]. 长春师范大学学报

二、读研期间参与的科研项目

1. 安徽高校协同创新项目《自主智能无人系统视觉感知与控制》