

doi: 10.16576/j.ISSN.1007-4414.2025.05.001

基于特征点匹配和卡尔曼滤波的抖动去除 前处理 VSLAM 算法优化

柴立平¹, 沈国伟¹, 朱利凯², 李 跃²

(1. 合肥工业大学, 安徽 合肥 230009; 2. 中国科学院合肥物质科学研究院, 安徽 合肥 230031)

摘要: 视觉 SLAM 是当前应用广泛的重要导航技术,但在实际应用中常受到多种因素制约,其中画面抖动是影响性能的主要瓶颈之一。为此,该文提出了一种基于 ORB 特征点跟踪与卡尔曼滤波相结合的抖动抑制优化算法。该方法通过分区块匹配特征点计算抖动幅度,并利用卡尔曼滤波对抖动幅度进行拟合处理,进而计算并补偿抖动补偿值,从而实现去除抖动干扰下的 SLAM 高精度建图。实验中以 ORB-SLAM 算法为对照,比较了改进算法与原始算法的运行结果。结果表明:小幅抖动下,改进算法相较于原版算法平均误差和均方根误差最高分别减小 7.68% 和 11.19%,大幅抖动下平均误差和均方根误差最高分别减小 74.89% 和 56.93%。这表明所提出的改进算法在不同抖动条件下均具有良好的防抖效果,并能显著提升视觉 SLAM 的建图精度。

关键词: 视觉 SLAM; ORB-SLAM3; 视频去抖; 特征点匹配; 卡尔曼滤波器

中图分类号: TP39; TP24

文献标识码: A

文章编号: 1007-4414(2025)05-0001-07

Optimization of a VSLAM Algorithm with Jitter Removal Preprocessing based on Feature Point Matching and Kalman Filtering

CHAI Li-ping¹, SHEN Guo-wei¹, ZHU Li-kai², LI Yue²

(1. Hefei University of Technology, Hefei 230009, Anhui, China;

2. Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, Anhui, China)

Abstract: Visual SLAM is a widely used navigation technology, but its performance in practical applications is often constrained by multiple factors. Among them, image jitter is one of the main bottlenecks. To address this issue, this paper proposes a jitter suppression optimization algorithm that combines ORB feature point tracking with Kalman filtering. The method calculates the jitter amplitude by block-based feature point matching, then applies Kalman filtering to fit the amplitude and estimate the compensation value, which is used to correct the jitter. In this way, high-precision SLAM mapping can be achieved under jitter interference. In experiments, the proposed algorithm was compared with the original ORB-SLAM algorithm. Results show that under small-amplitude jitter, the improved algorithm reduces the average error and root mean square error (RMSE) by up to 7.68% and 11.19%, respectively, compared with the original algorithm. Under large-amplitude jitter, the reductions reach up to 74.89% and 56.93%, respectively. These findings demonstrate that the proposed algorithm achieves robust jitter suppression under varying conditions and significantly improves the mapping accuracy of visual SLAM.

Key words: visual SLAM; ORB-SLAM3; video stabilization; feature point matching; Kalman filter

0 引言

同步定位与地图创建(SLAM)是指移动载体在未知环境中,依靠自身传感器,在运动过程中在构建环境地图的同时估计自身位姿的一种算法^[1]。SLAM 算法最早由 Smith Self 和 Cheeseman^[2] 提出,其被认为是机器人自主移动的基础。依据传感器的种类可以将其分为激光 SLAM 和视觉 SLAM,视觉 SLAM 的主要感知传感器为相机。ORB-SLAM(Oriented FAST

and Rotated BRIEF based SLAM)由 Mur-Artal R^[3] 等人提出,是一种基于 ORB(Oriented FAST and Rotated BRIEF)特征点^[4] 的视觉 SLAM,是当下研究最多、综合性能最高的视觉 SLAM 技术之一。笔者以 ORB-SLAM 为主要参考,下文称“原版算法”均指 ORB-SLAM,用于对比验证改进算法相较于原版算法的性能提升幅度。

视觉 SLAM 的运行效果,取决于相机实时采集的

收稿日期: 2024-08-27

基金项目: 国家自然科学基金面上项目: 高速绳移系统能量变化机理及振动控制研究(编号: 51675150);

安徽省自然科学基金面上项目: 高速轴向移动模态密度非均匀结构中频振动建模及预估(编号: 2208085ME130)。

作者简介: 柴立平(1969-),男,安徽滁州人,研究员,主要从事计算机视觉方面的研究工作。

通信作者: 沈国伟(2000-),男,江西九江人,硕士,主要从事视觉 SLAM 方面的研究工作。

环境视频信息质量,包括清晰度、亮度、帧率以及画面稳定程度等方面。实际应用中,在 ORB-SLAM 执行导航定位建图任务过程中,运动产生的结构振动以及运动环境干扰产生的振动均会造成探测相机采集视频出现抖动,使连续视频帧间出现较大差异,甚至会出现卷帘快门的畸变^[5],从而导致在 ORB-SLAM 的特征中不能进行有效地匹配,引起建图误差或无法有效建图。去抖研究是当前的研究挑战之一,常规去抖方法包括基于全局运动估计方法^[6]、基于光流法^[7]及基于特征点跟踪的方法^[8]。近几年来,随着卷积神经网络技术的不断提升,借助神经网络提出的相邻帧插帧法^[9]、边界外试图综合算子法^[10]以及自适应网格法^[11]也相继被提出。在计算机视觉与机器学习不断进步的背景下,如何发展更加快速、高效且精确的去抖算法,利用深度学习、多传感器融合与多方法协同等手段,对实时视频流和复杂运动环境下的算法结构进行优化,并探索新的算法模式,已成为当前的研究热点。

笔者将局部运动估计和特征点跟踪相结合,提出一种基于卡尔曼滤波器^[12]的抖动去除算法。具体方法为:首先采用优化的分区匹配策略,比较相邻帧同一区块内特征点的坐标变化,提取局部运动规律;然后通过各区块加权计算得到整体包含抖动的运动波形,并利用卡尔曼滤波器对该波形进行滤波整定,获得动态状态下连续平滑的运动曲线;最后逐帧计算原始抖动波形与平滑运动波形的差值,将其作为相机抖动补偿值,调整固定大小的取景框位置,使中央画面保持稳定,从而实现动态去抖效果。

1 基本原理

文中的优化算法包括输入、特征点提取与匹配、抖动状态分析与计算、抖动矫正、SLAM 处理与建图共计 5 个流程单元。在 ORB-SLAM 前端完成抖动图像的抖动拟制。在抖动状态分析与计算阶段,对图像进行优化匹配处理,以中央区域特征点为基准进行帧间匹配,并计算匹配点的坐标差,从而得到帧间抖动状态参数。随后,利用卡尔曼滤波器对该参数进行处理,获得动态条件下连续、平滑的运动曲线。通过计算抖动参数与平滑参数的差值,得到抖动补偿值,并通过移动取景框实现图像矫正。最终,将矫正后的图像输入至 SLAM 系统,实现高精度建图与实时导航。如图 1 所示为添加抖动矫正算法后的 ORB-SLAM3 算法完整流程,其中红色框线部分为该算法核心内容。

1.1 特征点提取与匹配

局部运动估计依赖于特征点的提取与匹配。特征点通常是指在图像或模式中具有显著性或独特性的位置或区域,一般由关键点和对应描述子共同构成。目前,学界主流认可且应用于实际的特征有 SIFT

特征(Scale Invariant Feature Transform)^[13]、SURF 特征(Speeded-Up Robust Features)^[14]和 ORB 特征^[4]。其中,ORB 特征点在保证了旋转和尺度不变的同时,计算效率也远远高于前两者,使得其在视觉 SLAM 算法中应用广泛。ORB、SURF、SIFT 三种特征点提取的耗时分别为 14.5、224.7、746.1 ms^[15]。

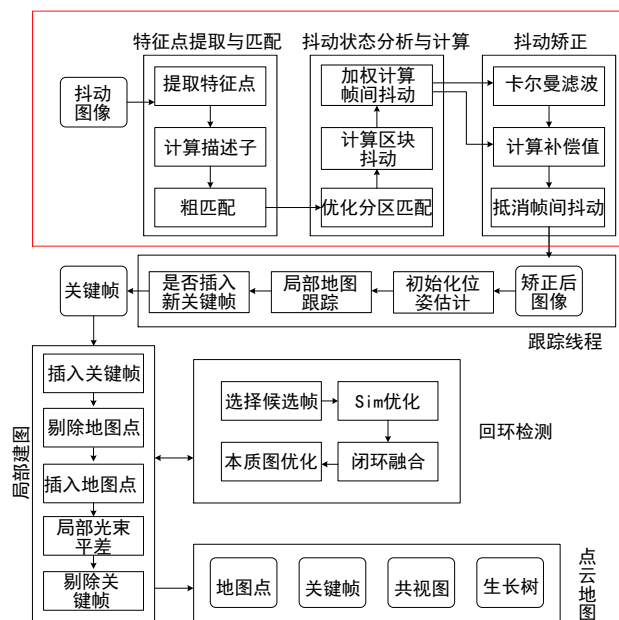


图 1 算法流程图

ORB 特征点由关键点和描述子两部分组成,其关键点称为 Oriented FAST 角点,是一种改进的 FAST 角点,具有旋转不变性;其描述子称为 BRIEF 描述子。

FAST 是一种角点,主要对图像灰度变化明显的区域进行检测,其核心思想是将像素点与其周围的像素点进行灰度比较,若差值大于设定阈值,体现在图像层面则是该点过暗或过亮,即该点有可能为角点。如图 2 所示为 FAST 特征点提取示意。

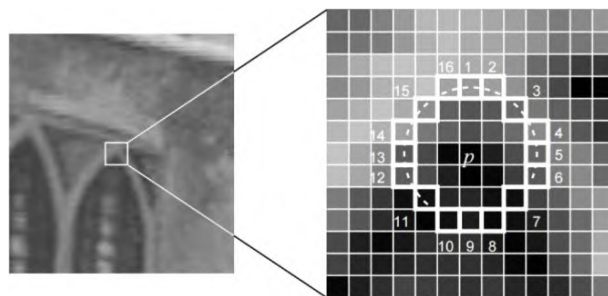


图 2 FAST 关键点

FAST 角点不具备尺度不变性和旋转不变性。ORB 特征点通过构建图像金字塔和灰度质心法^[16],使得 FAST 角点具有了尺度不变性和旋转不变性,从而大大提高了鲁棒性。这种改进后的 FAST 称为 Oriented FAST 关键点。

在关键点提取完成之后,需要对每个关键点进行描述子计算。描述子描述了对应关键点周围的像素信息。BRIEF 是一种二进制描述子,其编码了关键点附近按一定规则选取的像素点对。通常计算 BRIEF 描述子会通过选取 128 对像素点对,最后得到 128 维由 0 和 1 组成的向量^[17]。

特征点匹配是视觉 SLAM 算法中的关键步骤。通过在图像中按照统一标准提取的特征点平面坐标,可以与空间物体的三维坐标建立对应关系。利用相邻帧间特征点的匹配关系,可以获取相机位姿的变化信息。同时,相机抖动本身也属于相机运动,需要通过特征点匹配获取抖动信息,以实现抖动矫正。常用的匹配方法是暴力匹配,即将上一帧的每个特征点描述子与下一帧所有特征点描述子进行比较,然后根据匹配结果排序,选择最相近描述子作为对应匹配点。对于 BRIEF 等二进制描述子,通常使用汉明距离进行匹配,即计算两个二进制字符串中不同位的数量。如图 3 所示为相邻帧间依据汉明距离的匹配结果。



图 3 匹配结果

该算法在局部运动估计中依赖特征点的坐标变换,通过坐标差值计算抖动幅度。因此,特征点匹配的准确性会直接影响抖动幅度的计算,从而影响后续抖动去除的精度。特征点匹配对象来自相邻两帧图像,由于两帧间时间间隔较短,相机位置偏移幅度较小,特征点坐标变化也较小且整体趋势一致。在此基础上,特征点匹配正确性的直观表现为匹配线条大多呈近似平行状态。然而,通过观察粗匹配结果可发现,多数平行线条中仍存在较多交叉的误匹配线条,因此需要对粗匹配结果进行进一步优化处理。

1.2 抖动状态分析与计算

1.2.1 优化分区块匹配

对于粗匹配的误匹配情况,考虑将图像划分为区块,通过相邻帧同区块匹配,将待匹配对象限制在一个较小邻域范围内,提升了匹配的准确度。相较于对两张整幅图像所有特征点进行暴力匹配,分区块匹配的方式减少了计算次数,提升了效率。考虑相机抖动会导致图像的边缘部分在下一帧移出视野外,图像边缘的特征点均应该舍弃,划分的区块应在图像中央区域。该算法选择图像中央区域 5 块大小为 100×100 像素的区块。另外,在图像具有较多相似物体的情况

下,分区块匹配仍有较多误匹配存在,需要进一步优化匹配。通过设置最小汉明距离阈值的方式可以很好地将这一部分误匹配筛选出去。以暴力匹配的所有结果中最小的汉明距离为参照,设置 2 倍的最小汉明距离为阈值。具体筛选方式如式(1)所列:

$$\begin{cases} 1 & \text{dist} \leq \text{Max}(\text{dist}_{\min} \times 2, \text{limit}) \\ 0 & \text{dist} > \text{Max}(\text{dist}_{\min} \times 2, \text{limit}) \end{cases} \quad (1)$$

式中: dist 表示每组匹配间描述子的汉明距离; dist_{\min} 表示所有匹配中最小的汉明距离; Max 表示取较大值运算; limit 表示下限参数,避免因匹配特征点的汉明距离最小值过小而导致阈值设置过低,从而保留的匹配点数量过少的情况,经验数值多为 20~40 之间; 1 表示为优秀匹配,予以保留; 0 表示误匹配,从结果中删去。如图 4 所示为优化后的分区块匹配结果。

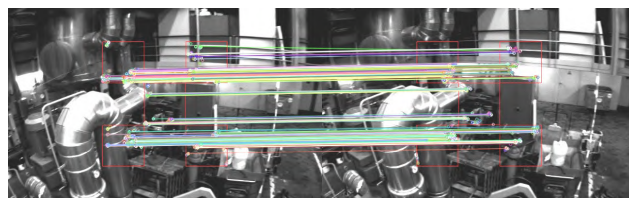


图 4 优化分区块匹配

可以看出,优化后的匹配结果数量相较粗匹配减少了许多,且均为相对平行线条,可以认为均为正确匹配,准确率大幅提升。

1.2.2 计算帧间抖动

两帧间相机产生的运动主要通过对对应特征点在各自帧的坐标差值得出。坐标差值如图 5 所示。

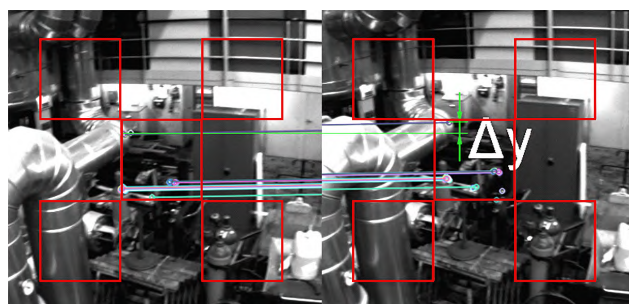


图 5 坐标差值

对每个区块中的匹配点对进行坐标差求值,如式(2)所列:

$$\begin{cases} \Delta y_t^N = \frac{\sum_{i=1}^n (y_{t-1}^{N,i} - y_t^{N,i})}{n}, t > 1 \\ \Delta x_t^N = \frac{\sum_{i=1}^n (x_{t-1}^{N,i} - x_t^{N,i})}{n}, t > 1 \\ \Delta y_t^N = 0, \Delta x_t^N = 0, t = 1 \end{cases} \quad (2)$$

式中: Δy_t^N 和 Δx_t^N 为第 t 帧和第 $t-1$ 帧中第 N 区块的坐标差值; $y_t^{N,i}$ 和 $x_t^{N,i}$ 为第 t 帧第 N 区块的第 i 个匹配点

坐标; n 为第 N 区块匹配点对数。

对每组相邻帧进行坐标差求值,会得到 5 组有 5 个区块的平均坐标差。考虑 5 个区块的分布情况,对 5 个区块分配不同权重。将中央区块赋予最高权重,周围 4 个区块赋予等值的较低权重,将加权结果作为运动幅度,如式(3)所列:

$$\begin{cases} d_{t,y} = p\Delta y_t^1 + q \sum_{N=2}^5 \Delta y_t^N \\ d_{t,x} = p\Delta x_t^1 + q \sum_{N=2}^5 \Delta x_t^N \\ p + 4q = 1 \end{cases} \quad (3)$$

式中: $d_{t,y}$ 和 $d_{t,x}$ 为第 t 帧和第 $t-1$ 帧的帧间抖动幅度; p 为中央区块的权重; q 为周围区块占比。对一段时间内的帧间抖动进行如式(4)所列的累加计算:

$$\begin{cases} Y_{t+\Delta t} = \sum_{i=t}^{t+\Delta t} d_{i,y} \\ X_{t+\Delta t} = \sum_{i=t}^{t+\Delta t} d_{i,x} \end{cases} \quad (4)$$

式中: $Y_{t+\Delta t}$, $X_{t+\Delta t}$ 分别代表第 $t + \Delta t$ 帧 y 轴方向和 x 轴方向的波形值。通过如上累加计算,可以得出这段时间相机运动波形。

1.3 抖动矫正

1.3.1 卡尔曼滤波处理

卡尔曼滤波器将状态、测量以及其中产生的噪声均视作服从高斯分布,通过建立状态方程和观测方程,利用状态方程和当前时刻的最优状态估计值,预测下一时刻的状态估计值,并将传感器测量值与预测的状态值进行比较,之后利用卡尔曼增益来修正预测值,得到更准确的状态估计。通过不断迭代预测和更新步骤,卡尔曼滤波器可以提供最优的系统状态估计,并能够有效地处理传感器噪声和系统动态变化。

卡尔曼滤波器主要工作原理可以分为状态预测和状态更新两大步骤,并对两个步骤循环迭代。预测步骤如式(5)所列:

$$\begin{cases} \hat{x}_k^- = F\hat{x}_{k-1} \\ P_k^- = FP_{k-1}F^T + Q \end{cases} \quad (5)$$

式中: \hat{x}_k^- 表示预测结果; F 表示状态矩阵,用于描述相机状态; \hat{x}_{k-1} 表示上一次的预测状态; P_k^- 表示预测状态的协方差; P_{k-1} 为上一次迭代的估计量协方差; Q 代表预测过程中出现的过程误差, P 和 Q 的初始化由手动设置。

预测之后需要将预测结果用于更新下一次迭代的预测状态,更新步骤如式(6)所列:

$$\begin{cases} K_k = \frac{HP_k^-}{HP_k^-H^T + R} \\ \hat{x}_k = H\hat{x}_{k-1} + K_k(Z - H\hat{x}_{k-1}) \\ P_k = (1 - K_kH)P_k^- \end{cases} \quad (6)$$

式中: H 代表观测矩阵; R 表示测量值不确定性的协方差; Z 代表当前迭代次数的观测值。此算法中将相机运动中的微小抖动视作噪声,利用卡尔曼滤波器将其去除以实现波形平滑,将波形值 $Y_{t+\Delta t}$ 作为观测值代入,可对抖动波形进行滤波处理。

1.3.2 计算补偿值

将卡尔曼滤波结果视作相机的主运动方向,则原始波形与卡尔曼滤波波形之间的差值即为抖动补偿值,计算公式如式(7)所列:

$$\begin{cases} y'_{\text{offset}} = Y_t - \hat{y}_t \\ x'_{\text{offset}} = X_t - \hat{x}_t \end{cases} \quad (7)$$

式中: y'_{offset} 和 x'_{offset} 分别为第 t 帧 y 轴方向和 x 轴方向补偿值; Y_t 和 X_t 分别为第 t 帧 y 方向和 x 方向波形值; \hat{y}_t 和 \hat{x}_t 分别为第 t 帧 y 方向和 x 方向滤波值。

1.3.3 抵消帧间抖动

设置可移动取景框,随着抖动移动取景框以保证在一组连续帧中,画面物体保持在画面相对固定区域,以实现画面矫正。如图 6 所示。

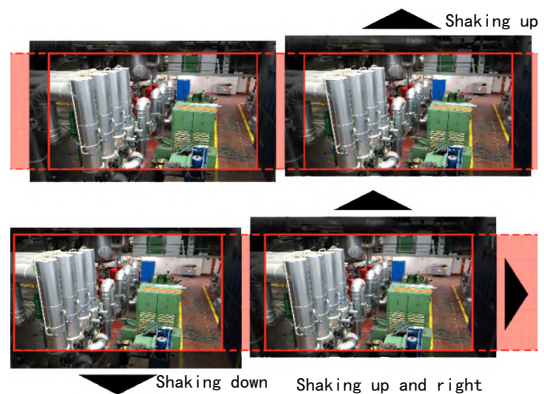


图 6 抖动矫正

设置固定大小的可移动取景框,其中心点坐标方程如式(8)所列:

$$\begin{cases} C(x_t, y_t) = C \begin{pmatrix} x_{t-1} + x'_{\text{offset}} \\ y_{t-1} + y'_{\text{offset}} \end{pmatrix}, t > 1 \\ C(x_t, y_t) = \left(\frac{1}{2}X, \frac{1}{2}Y \right), t = 1 \end{cases} \quad (8)$$

式中: $C(x_t, y_t)$ 为第 t 帧取景框中心点坐标; X 、 Y 为原始图像的尺寸大小。

在对画面矫正的过程中,当出现补偿值较大导致取景框移出画面外的情况时,为保证图像大小的统一性,传统去除抖动算法会在边缘无图像区域填充纯色以保证图像大小的统一性。文中算法在移动取景框前对原始图像边缘进行扩展处理,保证了图像大小统一性的同时还使视觉层面不会产生割裂感,并且扩展部分并不会对特征点的提取产生影响。扩展前后特征点提取对比如图 7 所示。

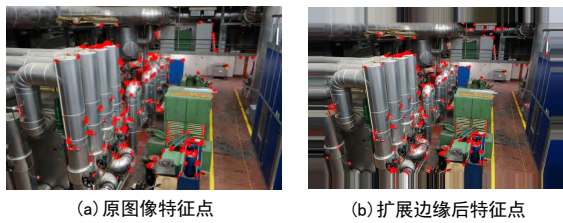


图7 扩展前后特征点

通过式(8)可以将计算所得抖动幅度转换为取景框的移动大小,移动取景框对相机抖动造成的画面偏移做出补偿,使得一组连续帧中,物体保持在画面相对固定区域,达到矫正抖动的效果。

2 实验验证

2.1 实验设计

为验证所提算法在视觉 SLAM 运行过程中对视频抖动的优化效果,文中选用公开仿真数据集 EuRoC 作为实验平台,分别运行主流的 ORB-SLAM 算法与抖动矫正优化后的算法。随后,借助 EVO(Evaluation of Odometry)工具,将两种算法的运行结果与数据集的真值轨迹进行对比分析,从误差角度评价优化算法在实际运行中的效果与优势。

EuRoC 是由 Autonomous Systems Lab 提供的公开数据集,包含室内、室外多场景下运行的数据包,并提供通过外部定位方式记录的相机真实的运动轨迹和位姿,作为对照标准^[18],是目前研究人员常采用的数据集之一。

文中采用 MH01 和 V101 数据包。MH01 数据包场景是较为复杂的工厂内部,较为符合多数工厂机器人实际工作场景,选用此数据包可以最大程度模拟真实情况;V101 数据包场景为室内狭小场景,具有丰富特征点。同时,为了模拟实际运行中由于结构振动、路面环境不平、采集相机自身噪声等影响而产生的抖动,需要对数据包添加抖动以验证算法对于抖动的矫正效果。综合考虑多种因素产生的抖动特点,对数据包添加 y 方向均匀分布的 ± 10 像素抖动,模拟因路面不平整产生的小幅抖动;添加 y 方向均匀分布的 ± 15 像素抖动,模拟因机械结构振动产生的大幅抖动;添加服从 $X \sim N(10, 3)$ 高斯分布的抖动幅度,模拟多数场景下产生的自然抖动。之后对添加抖动后的数据分别运行原版 ORB-SLAM 算法和抖动优化算法,再借助 EVO 工具对实验结果进行分析、比对。

文中以轨迹长度(Path Length)和绝对轨迹误差(Absolute Pose Error, APE)为主要评价指标。轨迹长度直接反映了不同算法运行结果和真实值之间的差距,可以从总体层面评价算法的优劣性;APE 主要计算相机估计位姿和真实位姿之间的变换,具体计算如式(9)所列:

$$\begin{cases} E_i = P_{\text{truth},i}^{-1} P_i \\ APE_i = \| \text{trans}(E_i) \| \end{cases} \quad (9)$$

式中: $P_{\text{truth},i}^{-1}$ 表示第*i*时刻的真实位姿; P_i 表示第*i*时刻的估计位姿; E_i 表示估计位姿和真实位姿之间的变换矩阵; trans 表示真实值与估计值之间的轨迹误差部分。之后可以使用 APE 均方根数值比较两种算法结果的差异,具体计算如式(10)所列:

$$RMSE(APE) = \left(\frac{1}{m} \sum_{i=1}^m APE_i^2 \right)^{\frac{1}{2}} \quad (10)$$

分别计算原版算法和文中算法在运行添加抖动后数据包所得实验结果与真实轨迹绝对轨迹误差和其他相关数值,以判断在抖动环境中文中算法相较于原版算法的优劣。

2.2 实验结果

在向数据集添加抖动后,分别运行原版算法和文中改进后算法,记录滤波处理前后的运动波形,通过波形可以直观看出文中算法对于原本大幅度抖动的波形的去除效果。波形图如图8所示。

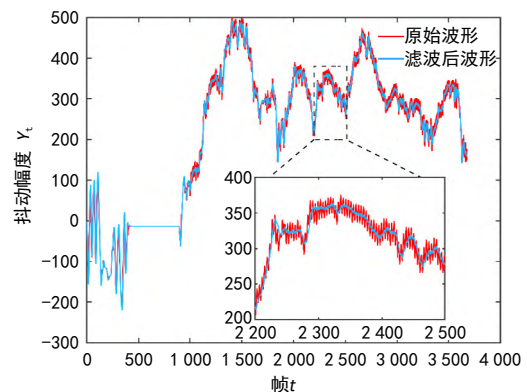
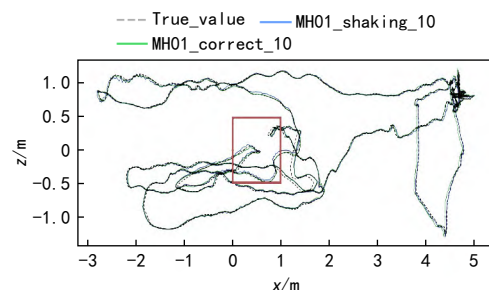


图8 滤波前后波形

分析图8可以看出,原始波形抖动幅度较大,模拟的是未作任何抖动处理的原版算法应用的工作场景,滤波后的波形是文中所提算法去除抖动后的运动波形,整体更加平滑。

将两种算法运行所得的轨迹路线文件,使用 EVO 工具转换为可视化轨迹图像,结果如图9~11所示。其中虚线代表真实值轨迹,蓝色线表示原版算法轨迹,绿色线表示抖动去除后轨迹。



(a) 全局轨迹图

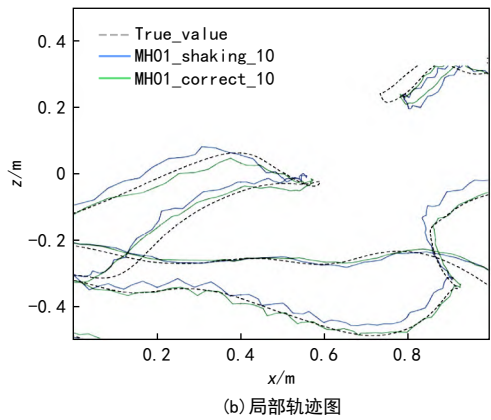


图 9 10 像素抖动轨迹图

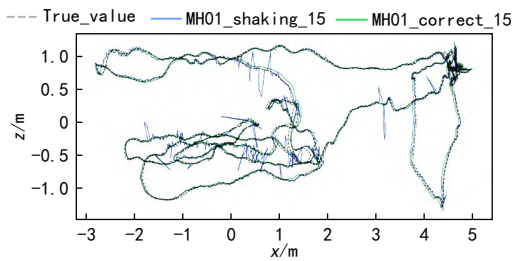
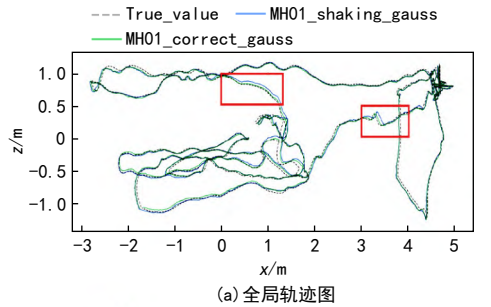
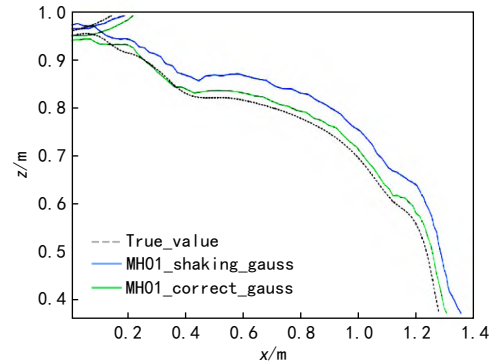


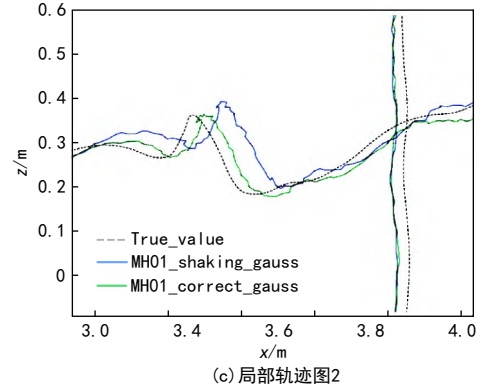
图 10 15 像素抖动轨迹图



(a) 全局轨迹图



(b) 局部轨迹图1



(c) 局部轨迹图2

图 11 高斯抖动轨迹图

观察 3 组抖动矫正前后轨迹图可以看出, 10 像素抖动经过文中算法矫正抖动后的轨迹在拐角处更加贴近真实值轨迹; 15 像素由于抖动幅度较大, 原版算法运行过程中存在多次追踪丢失的情况, 这一点在轨迹图像中可以得到直观反映, 抖动轨迹线条有多处出现大幅偏离真实值的情况; 对于高斯抖动, 矫正后的轨迹在平直路段和拐角路段均更贴近真实值轨迹。

分别对两组数据集不同抖动条件下计算轨迹长度。如表 1 所列为两种算法在不同数据集下运行所得轨迹长度与真值之间的差异情况。

表 1 轨迹长度

数据	算法	轨迹长度/m	误差率/%
MH01 (True value 80.626 m)	10 像素 抖动 ORB-SLAM3	84.805	5.18
	文中算法	83.349	3.38
	15 像素 抖动 ORB-SLAM3	113.699	41.02
	文中算法	84.461	4.76
	高斯 抖动 ORB-SLAM3	82.934	2.86
	文中算法	81.704	1.34
V101 (True value 58.592 m)	10 像素 抖动 ORB-SLAM3	67.566	15.32
	文中算法	61.102	4.28
	15 像素 抖动 ORB-SLAM3	162.429	177.22
	文中算法	63.936	9.12
	高斯 抖动 ORB-SLAM3	61.541	5.03
	文中算法	60.539	3.32

从表中可以看出, 文中算法对抖动进行矫正有较好效果。从轨迹长度可以看出, 由于抖动的存在, ORB-SLAM3 算法受到扰动程度较大, 得出的轨迹长度与真实值有较大差别。其中, 在 15 像素抖动条件下, 原版算法在 V101 数据集的运行过程中出现多次跟踪丢失, 导致计算得到的轨迹长度误差较大, 已不具备参考价值。除该情况外, 其余实验组中, 文中算法所得轨迹长度均更接近真实值, 误差率相比原版算法降低了 30%~70%。

分别对两组数据集不同抖动条件下计算绝对轨迹误差。如表 2 所列为不同数据集在两种算法运行结果下的绝对轨迹误差统计数据。

在两组数据集添加不同幅度的抖动条件下, 文中算法相较于原版 ORB-SLAM3 算法, 误差均有一定程度地缩小。对于 MH01 数据集, 10 像素抖动下最大误差缩小了 13.34%, 平均误差缩小了 7.68%, 均方根误差缩小了 9.36%; 15 像素抖动下最大误差缩小了 83.47%, 平均误差缩小了 3.62%, 均方根误差缩小了 19.82%; 高斯抖动下最大误差缩小了 15.02%, 平均误差缩小了 7.13%, 均方根误差缩小了 11.19%。对于 V101 数

据集, 10 像素抖动下最大误差缩小了 45.55%, 平均误差缩小了 2.51%, 均方根误差缩小了 3.09%; 15 像素抖动下最大误差缩小了 92.68%, 平均误差缩小了 56.93%, 均方根误差缩小了 74.89%; 高斯抖动下最大误差缩小了 12.23%, 平均误差缩小了 0.91%, 均方根误差缩小了 1.04%。其中, 文中算法相较于原算法在两组数据的 15 像素抖动条件下提升最为显著。通过对运行画面的观察可以发现, 由于 15 像素抖动幅度较大, 原算法在运行过程中多次出现跟踪丢失现象, 导致轨迹误差大幅增加; 而文中算法在抖动矫正后未出现跟踪丢失, 从而有效减小了误差。

表 2 绝对轨迹误差统计数据

数 据	算 法	Max	Mean	RMSE
MH01	10 像素 ORB-SLAM3	0.123 011	0.043 242	0.049 507
	抖动 文中算法	0.106 605	0.039 919	0.044 873
	15 像素 ORB-SLAM3	0.721 949	0.042 587	0.057 742
	抖动 文中算法	0.119 365	0.041 047	0.046 296
	高斯 ORB-SLAM3	0.152 814	0.055 252	0.063 057
	抖动 文中算法	0.129 860	0.051 310	0.055 999
V101	10 像素 ORB-SLAM3	0.289 780	0.084 897	0.090 972
	抖动 文中算法	0.157 776	0.082 767	0.088 162
	15 像素 ORB-SLAM3	3.520 441	0.206 997	0.375 445
	抖动 文中算法	0.257 819	0.089 153	0.094 289
	高斯 ORB-SLAM3	0.181 872	0.085 084	0.090 182
	抖动 文中算法	0.159 625	0.084 311	0.089 247

由表中数据可以看出, 在处理两组数据集不同抖动条件数据时, 文中算法运行所得绝对轨迹误差的各项统计数据都优于原版 ORB-SLAM3 算法, 因此文中算法对不同场景下的抖动均有一定优化效果。

3 结 论

实际应用中, 抖动造成的相机画面抖动成为制约视觉 SLAM 精确度的一大瓶颈。文章以 ORB 特征点为锚点, 在 VSLAM 前端数据采集部分添加处理算法, 提出了一种基于卡尔曼滤波器的防抖算法, 所得结论如下。

(1) 提出了一种分区匹配的方法。该算法在抖动矫正过程中依赖于匹配特征点坐标变化的计算, 而分区匹配策略显著提升了特征点匹配的准确性, 从而为后续的抖动矫正提供了更加可靠的依据。

(2) 基于卡尔曼滤波器实现抖动去除。通过对抖动条件下的运动波形进行分析, 并采用卡尔曼滤波器对其进行平滑处理, 能够有效抑制数据中的抖动成分, 表现出显著的矫正效果。

通过实验验证, 对比了两种算法在含有抖动的数据包上的运行结果。在抖动幅度较小时, 文中算法相

较于原版算法的平均误差和均方根误差最高分别降低 7.68% 和 11.19%; 而在大幅抖动条件下, 文中算法的优势更加显著, 平均误差和均方根误差最高分别降低 74.89% 和 56.93%。实验结果表明, 文中算法对 ORB-SLAM3 的性能具有显著优化效果。同时, 由于该算法仅在前端数据采集环节进行了改进, 因此可以合理推断, 其对多种类型的 VSLAM 算法框架均具有潜在的提升作用。

参考文献:

- [1] 顾照鹏, 刘 宏. 单目视觉同步定位与地图创建方法综述 [J]. 智能系统学报, 2015, 10(4): 499-507.
- [2] 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(6): 1052-1067.
- [3] 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.
- [4] Rublee E, Rabaud V, Konolige K, et al. ORB: An efficient alternative to SIFT or SURF[C]. International Conference on Computer Vision, 2011: 2564-2571.
- [5] 杨涛涛. 基于全景拼接的视频防抖算法设计 [D]. 成都: 电子科技大学, 2018.
- [6] 任正玮. 全景电子稳像中全局运动估计及运动滤波方法研究 [D]. 长春: 长春理工大学, 2018.
- [7] 熊 炜, 王传胜, 李利荣, 等. 结合光流法和卡尔曼滤波的视频稳像算法 [J]. 计算机工程与科学, 2020, 42(3): 493-499.
- [8] 王传胜. 基于特征跟踪的视频稳像算法研究 [D]. 武汉: 湖北工业大学, 2020.
- [9] Yu J, Ramamoorthi R. Robust video stabilization by optimization in cnn weight space[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 3800-3808.
- [10] Yu J, Ramamoorthi R. Learning video stabilization using optical flow[C]. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 8159-8167.
- [11] Zhao M, Ling Q. Adaptively meshed video stabilization[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2020, 31(9): 3504-3517.
- [12] Kalman R E. A new approach to linear filtering and prediction problems[J]. Journal of Basic Engineering, 1960(82D): 35-45.
- [13] Lowe D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60(2): 91-110.
- [14] Bay H, Tuytelaars T, Van Gool L. Surf: Speeded up robust features [C]. European conference on computer vision, 2006: 404-417.
- [15] 傅柳军. 面向动态环境的 ORB-SLAM3 优化 [D]. 武汉: 武汉大学, 2021.
- [16] Rosin P L. Measuring corner properties[J]. Computer Vision and Image Understanding, 1999, 73(2): 291-307.
- [17] Calonder M, Lepetit V, Strecha C, et al. BRIEF: Binary robust independent elementary features[J]. Springer, Berlin, Heidelberg, 2010: 778-792.
- [18] 杨文韬. 基于视觉语义 SLAM 的无人机路径规划算法研究 [D]. 济南: 山东大学, 2023.