Dynamic Environments and Robust SLAM: Optimizing Sensor Fusion and Semantics for Wheeled Robots

Jaafar Mahmoud, Andrey Penkovskiy
ITMO University
Saint Petersburg, Russia
jaafar.a.mahmoud, aapenkovskiy@itmo.ru

***Abstract*—This paper proposes an approach to enhance the robustness and accuracy of visual simultaneous localization and mapping (SLAM) for ground wheeled mobile robots in dynamic environments. The proposed method incorporates encoder measurements to establish optimization constraints in bundle adjustment. To further improve robustness, a geometric technique utilizing KMeans clustering with epipolar constraints and the SegNet [1] for semantic segmentation is employed to filter out features detected on moving objects. These modifications are integrated into the state-of-the-art SLAM system ORB-SLAM3 [2] and demonstrate superior accuracy and real-time performance compared to the baseline approach. The effectiveness of the proposed method is demonstrated through multiple OpenLoris and IROS Lifelong SLAM competition scenarios.**

I. Introduction

Simultaneous localization and mapping (SLAM) systems for mobile robots have emerged as a critical area of research due to their numerous applications in logistics, monitoring, and inspection. Autonomous navigation is a crucial component of these applications, and SLAM systems play a vital role in achieving efficient task performance.

SLAM systems can be broadly categorized into two types: **filter-based** and **graph-based** approaches [2]–[10]. Filter-based methods use a multi-state constraint Kalman filter [11], which is an extended Kalman filter that fuses sensors in SLAM. However, the computational cost of computing Jacobians for large maps limits the algorithm's scalability in large environments. On the other hand, graph optimization-based algorithms have gained popularity in recent years as they are easier to implement using open-source frameworks such as [12]–[15] and can handle a larger set of sensors.

Here we introduce a modified version of graph-based ORB-SLAM3 [2]. It is a well-developed and widely used state-of-the-art approach that has been tested over different camera models (mono, stereo) with different set of lenses (pinhole, fish-eye) with/without the IMU sensor showing stunning performance. Therefore, we took this approach as the baseline for our research.

However, the ORB-SLAM3 system can be not robust enough for certain conditions when mobile robots are operating in dynamic environments. Thus, here we aim to improve its accuracy and robustness by applying the power of optimization techniques, sensor fusion, and incorporating semantics of the scene.

The fusion of odometric measurements and optical frames has been shown to enhance the accuracy and robustness of visual simultaneous localization and mapping (SLAM) systems, particularly in challenging low-illumination conditions. The objective of this study is to investigate the efficacy of fusing optical frames with encoders, as compared to using an inertial measurement unit (IMU), for wheeled mobile robots. For example, it was shown in [16] that for Visual Inertial SLAM (VINS) scale parameter and robot's global orientation becomes unobservable, when it performs even basic movement with constant acceleration or simply does not rotate during the inertial initialization period. This phenomena leads to a significant loss in accuracy of VINS pose estimation result. On the other hand, several recent works elaborate on fusion of optical data with encoders' measurements. DRE-SLAM [17] addresses the task of building a static map, while odometric measurements from encoders are tightly coupled with optical data using graph optimization based method. SE2CLAM [18] implemented visual SLAM for SE(2) planar motion as a unary constraint on SE(3) robot pose. SE2LAM [19] proposed a novel constraint SE2-XYZ that allow to parameterize robot pose on SE(2) along considering the out-of-SE(2) motion perturbation.

Our approach builds upon the SE2LAM [19] methodology, which has been demonstrated to effectively mitigate potential drift in visual odometry and perform well in static environments. However, real-world scenarios often involve moving objects, making it critical to enhance the robustness of SLAM systems by removing such objects. To address this, we incorporate an outlier rejection algorithm to complement our system.

In general, there are two types of methods for the outliers' rejection algorithm. **Geometric-based** methods are quick and important for improving the accuracy of border enhancement, because low-level information (pixels' value) is processed directly. **AI-based** methods provide processing of a higher-level visual information by detecting, classifying, and semantically annotating objects in the scene like people, cars, or pets. State-of-the-art approaches combine both methods and can be considered as **geometric-AI**.

For example, DynaSLAM [20] uses **Mask R-CNN** to detect possibly moving objects, along with multi-view geometry based method (epipolar constraint). DS-SLAM [21] uses semantic segmentation model (SegNet [1]), along with moving consistency check and is based on ORB-SLAM2 [7]. DreSlam [17] utilizes an object detection model (YOLO) along with K-means clustering method for segmentation over the depth data from the RGB-D sensor. Detect-Slam [22] uses a DNN-based object detector, along with propagating probabilities of features (probability of detecting this feature on a moving object). We propose the solution based on the work of DS-SLAM by implementing similar model for the most recent ORB-SLAM3 algorithm. Key-frames are segmented using SegNet, and moving objects are detected semantically. We process the depth data using a K-mean clustering algorithm to obtain a higher level model, i.e. by detecting moving clusters instead of detecting just moving feature points. It is suitable for the outliers located on static objects that are moved (e.g. moved chair). We additionally filter the good matches by epipolar constraint, which helps to remove bad matches, that were not detected on a moving object or a cluster.

The main contributions of this work are an extended version of ORB-SLAM3 system that outperforms recent solutions in terms of robustness and accuracy of localization and mapping for mobile robots operating in dynamic environments by introducing:

1) an algorithm for reducing drift in robot localization that utilizes wheel odometry and optical frames fusion together with constrained optimization in the SE(2) space in the process of camera pose estimation and bundle adjustment;
2) geometric-AI method for outliers' rejection that detects moving objects and excludes corresponding points at the stage of visual feature extraction.

To evaluate the effectiveness of our proposed solution, we conducted several steps. First, we utilized the OpenLoris dataset [23], an open-source benchmarking dataset that provides recordings of stereo fish-eye cameras, RGB-D sensors, encoders, and IMU measurements of a mobile wheeled robot in various scenarios (e.g., cafes, corridors, rooms, and markets) that include moving objects. Secondly, we employed the TUM-RGBD dataset [24], which is an open-source dataset that has been used in previous works to assess real-time performance, robustness, and accuracy benchmarks, specifically for evaluating outlier rejection algorithms. The remainder of the paper is structured as follows: Section 2 describes the proposed modifications to the ORB-SLAM3 system that involve the fusion of odometric measurements with optical frames and the optimization process. Section 3 presents the algorithm for the outlier rejection model and its implementation in ORB-SLAM3. Section 4 presents the evaluation and comparison of our proposed solution with other state-of-the-art algorithms in terms of accuracy and execution speed. Finally, Section 5 concludes our work and discusses future directions.

II. Fusing encoders' measurements with optical frames

This section describes two types of constraints used in the optimization process. Here, we state their implementation in the ORB-SLAM3 system in the bundle adjustment function.

A. Optimization Constraints on SE(2)

1) The projection Constraint: The feature-based SE(2)-XYZ [19] benefits with encapsulating the out-of-SE(2) motion perturbation and directly parameterizes the robot's poses on SE(2). The projection equation from a landmark $\mathbf{l}_\ell$ w.r.t robot body coordinate system to the image plane is:

$$\mathbf{u}\left(\nu_i, \mathbf{l}_\ell\right) = \Pi({}^{C}\mathbf{R}_B \mathbf{R}_i^T \left(\mathbf{l}_\ell - \mathbf{p}_i\right) + {}^{C}\mathbf{p}_B) + \eta_u \tag{1}$$

in which $\eta_u \sim \mathcal{N}\left(\mathbf{0}, \sigma_u^2 \mathbf{I}_2\right)$ is the projection uncertainty, and $[{}^{C}\mathbf{R}_B | {}^{C}\mathbf{p}_B]$ are calculated from extrinsic calibration of the camera with respect to the body frame.
The out-of-SE(2) motion [19] includes two parts: translation perturbation along z as $\eta_z \sim \mathcal{N}\left(\mathbf{0}, \sigma_z^2\right)$ and rotation perturbation *xy* as $\boldsymbol{\eta}_{xy} \sim \mathcal{N}\left(\mathbf{0}_{2x1}, \boldsymbol{\Sigma}_{\theta_{xy}}\right)$. Therefore, the pose can be written as:

$$\mathbf{R}_i \leftarrow \text{Exp}(\underbrace{\left[\boldsymbol{\eta}_{\theta_{xy}}^T 0\right]^T}_{\boldsymbol{\eta}_\theta})\mathbf{R}_i, \quad \mathbf{p}_i \leftarrow \mathbf{p}_i + \underbrace{\left[\begin{array}{ccc} 0 & 0 & \eta_z \end{array}\right]^T}_{\boldsymbol{\eta}_z} \tag{2}$$

then the projection equation (1) becomes

$$\begin{aligned} &\mathbf{u}\left(\boldsymbol{\nu}_i, \mathbf{l}_\ell\right) \\ &\quad = \pi\left({}^{C}\mathbf{R}_B \mathbf{R}_i^T \, \mathrm{e}^{(-\boldsymbol{\eta}_\theta)} \left(\mathbf{l}_\ell - \mathbf{p}_i - \boldsymbol{\eta}_z\right) + {}^{C}\mathbf{p}_B\right) + \boldsymbol{\eta}_u \\ &\quad \approx \pi\left({}_{C_i}\mathbf{l}_\ell\right) + \mathbf{J}_{\boldsymbol{\eta}_\theta} \mathbf{u}_{\boldsymbol{\theta}} \boldsymbol{\eta}_\theta + \mathbf{J}_{\boldsymbol{\eta}_{\boldsymbol{z}}}^{\mathbf{u}} \boldsymbol{\eta}_z + \boldsymbol{\eta}_u \\ &\quad = \pi\left({}_{C_i}\mathbf{l}_\ell\right) + \delta\boldsymbol{\eta}_u \end{aligned} \tag{3}$$

where $\delta\boldsymbol{\eta}_u$ is a synthetic zero-mean noise, and $\mathrm{e}^{(-\boldsymbol{\eta}_\theta)} = Exp\left(-\boldsymbol{\eta}_\theta\right)$. The noise η_θ, η_z and η_u is not interdependent, thus we can compute the covariance matrix as

$$\boldsymbol{\Sigma}_{\delta\boldsymbol{\eta}_u} = \mathbf{J}_{\boldsymbol{\eta}_\theta}^{\mathbf{u}} \Lambda_{12} \boldsymbol{\Sigma}_{\theta_{xy}} \Lambda_{12}^T \mathbf{J}_{\boldsymbol{\eta}_\theta}^{\mathbf{u}T} + \sigma_z^2 \mathbf{J}_{\boldsymbol{\eta}_z} \mathbf{e}_3 \mathbf{e}_3^T \mathbf{J}_{\boldsymbol{\eta}_z}^{\mathbf{u}T} + \sigma_u^2 \mathbf{I}_2 \tag{4}$$

As the matching feature of the corresponding landmark is represented in pixels ${}^{i\ell}\boldsymbol{u}$, we can formulate our re-projection error as:

$$e^{i\ell} = \pi({}_{C_i}\boldsymbol{l}_\ell) - {}^{i\ell}\boldsymbol{u} \tag{5}$$

Graph optimization method is used for the minimization of the error stated in (5). The information matrix for (5) is the inverse of the covariance matrix $\boldsymbol{\Sigma}_{\delta\boldsymbol{\eta}_u}$. The Jacobian matrix of $e^{i\ell}$ are

$$\begin{aligned} \mathbf{J}_i^{i\ell} &= \left[\frac{\partial^{i\ell}\mathbf{e}}{\partial \mathbf{r}_i} \ \frac{\partial^{i\ell}\mathbf{e}}{\partial \phi_i}\right], \\ \frac{\partial^{i\ell}\mathbf{e}}{\partial \mathbf{r}_i} &= -\mathbf{J}^\pi \left({}_{C_i}\mathbf{l}_\ell\right)^C \mathbf{R}_B \mathbf{R}_i^T \Lambda_{12} \\ \frac{\partial^{i\ell}\mathbf{e}}{\partial \phi_i} &= \mathbf{J}^\pi \left({}_{C_i}\mathbf{l}_\ell\right)^C \mathbf{R}_B \mathbf{R}_i^T \left(\mathbf{1}_\ell - \mathbf{p}_i\right)^\wedge \mathbf{e}_3 \\ \mathbf{J}_\ell^{i\ell} = \frac{\partial^{i\ell}\mathbf{e}}{\partial \mathbf{l}_\ell} &= \mathbf{J}^\pi \left({}_{C_i}\mathbf{l}_\ell\right)^C \mathbf{R}_B \mathbf{R}_i^T \end{aligned} \tag{6}$$

This projection error is used in graph optimization for bundle adjustment as the projection edge (Figure 1).

2) The odometric edge: Inspired by the work on preintegrated IMU on SE(3) by [25], se2lam have formulated the preintegration of encoder's measurements on SE(2). From the motion model of the wheel encoder, we get the robot body pose $\boldsymbol{\nu}_i$ and $\boldsymbol{\nu}_j$ between two consecutive timestamps *k, k+1*, respectively. The preintegrated measurement (ϕ - rotation and $\boldsymbol{r}$ - translation) and the corresponding noises between key-frame *i*, *j* are formulated as:

$$\begin{aligned} {}^{i}\phi_j &:= {}^{i}\tilde{\phi}_j - \delta^{i}\phi_j \\ {}^{i}_{i}\mathbf{r}_j &:= {}^{i}_{i}\tilde{\mathbf{r}}_j - \delta^{i}_{i}\mathbf{r}_j \end{aligned} \tag{7}$$

The propagation of the integrated noise $\delta^{i}\phi_j$, $\delta^{i}_{i}\mathbf{r}_j$ is written in compact form as in [19]

$$\begin{gathered} \left[\begin{array}{c} \delta^{i}_{i}\mathbf{r}_{k+1} \\ \delta^{i}\phi_{k+1} \end{array}\right] := \delta^{i}\boldsymbol{\nu}_{k+1} = \mathbf{A}_k\delta^{i}\boldsymbol{\nu}_k + \mathbf{B}_k\boldsymbol{\eta}_{\nu k}, \\ \mathbf{A}_k = \left[\begin{array}{cc} \mathbf{I}_2 & \Phi\left({}^{i}\tilde{\phi}_k\right)1^{\times}\tilde{\mathbf{r}}_k \\ \mathbf{0} & 1 \end{array}\right], \mathbf{B}_k = \left[\begin{array}{cc} \Phi\left({}^{i}\tilde{\phi}_k\right) & \mathbf{0} \\ \mathbf{0} & 1 \end{array}\right] \end{gathered} \tag{8}$$

Hence, the covariance of odometric measurement $\delta^{i}\boldsymbol{\nu}_k$ can be propagated at each step:

$$\boldsymbol{\Sigma}_{\delta^{i}\nu_{k+1}} = \mathbf{A}_k\boldsymbol{\Sigma}_{\delta^{i}\nu_k}\mathbf{A}_k^T + \mathbf{B}_k\boldsymbol{\Sigma}_{\nu k}\mathbf{B}_k^T \tag{9}$$

We can now formulate the error function of the preintegrated odometric measurement as follows:

$${}^{ij}\mathbf{e} = \left[\begin{array}{c} \Phi\left(-\phi_i\right)\left(\mathbf{r}_j - \mathbf{r}_i\right) \\ \phi_j - \phi_i \end{array}\right] - \left[\begin{array}{c} {}^{i}\tilde{\mathbf{r}}_j \\ {}^{i}\tilde{\phi}_j \end{array}\right] \tag{10}$$

where its information matrix is the inverse of the covariance term $\boldsymbol{\Sigma}_{\delta\boldsymbol{\eta}_u}$. The Jacobian of error function is:

$$\begin{aligned} \mathbf{J}_i^{ij} &= \frac{\partial^{ij}\mathbf{e}}{\partial\boldsymbol{\nu}_i} = \left[\begin{array}{cc} -\Phi\left(-\phi_i\right) & -\Phi\left(-\phi_i\right)1^{\times}\left(\mathbf{r}_j - \mathbf{r}_i\right) \\ \mathbf{0} & -1 \end{array}\right] \\ \mathbf{J}_j^{ij} &= \frac{\partial^{ij}\mathbf{e}}{\partial\boldsymbol{\nu}_j} = \left[\begin{array}{cc} \Phi\left(-\phi_i\right) & \mathbf{0} \\ \mathbf{0} & 1 \end{array}\right] \end{aligned} \tag{11}$$

The preintegrated odometric measurements are used in graph optimization for the odometric edge (Fig. 1).

B. Implementation of visual odometric optimization edges in ORB-SLAM3:

The ORB-SLAM3 system utilizes a graph optimization method, wherein the error function is formulated as a nonlinear least-squares problem and subsequently minimized using techniques such as Gauss-Newton or Levenberg–Marquardt. In ORB-SLAM3, this optimization process is incorporated in the Bundle Adjustment (BA) function, which optimizes the local window of key-frames' poses and the associated local map points.

The ORB-SLAM3 system [6] operates on parallel threads, including the Tracking, Mapping, and Loop Closing threads, along with an additional thread for performing global bundle adjustment. Specifically, **1**) the Tracking thread serves as the front-end of the system, processing data from sensors for the initial pose estimation and determining when to update the map by adding new key-frames or points based on the optical frames from the camera and other sensor data. **2**) The Mapping thread processes the key-frames and map points, with bundle adjustment executed every time a new key-frame is added to the map. **3**) The Loop Closing thread detects loops during tracking to correct optical drift.

In the following section, we describe the modifications we made to the ORB-SLAM3 system.

1) ***In the tracking thread:*** The Tracking thread in the ORB-SLAM3 system functions as the front end of the system and is responsible for reading odometric measurements from encoders and converting them into odometric poses. The ticks from encoders can be transformed into SE2 poses using the motion model of encoders at specific timestamps. To synchronize the optical frames' timestamps with the odometric data, we perform offline or online interpolation.

The SE2 poses can be interpolated linearly over the X, Y axes and linearly over the angles around the Z-axis, although this is not the most efficient method. To address this, we use the **slerp** method, which interpolates the Quaternion representation of the data.

To build the first map, the system needs to be initialized after processing the encoder data. We modified the initialization process by incorporating odometric measurements with monocular frames since the scale is unobservable to the monocular vision alone. The optical criteria for successful initialization are satisfied when two optical frames have sufficient parallax for the first triangulation between their matches. Instead of calculating the Fundamental or Essential matrix as in standard ORB-SLAM3, we use odometric poses to estimate the first transformation between these two optical frames.

After successful initialization, the odometric poses will act as the first estimates of the tracking pose. The main advantage of odometric poses is that they provide a reliable relative estimate between frames. These poses are prone to a neglected drift that will be corrected by bundle adjustment or loop closure. The odometric poses without optical correction will drift after some time (even if no slipping happens in the wheels). However, they provide an odometric trajectory in bad optical situations e.g. insufficient illumination, low-textured planes, etc. This state is called in the ORB-SLAM3 system as the **Recently-Lost**. It is used basically with the IMU, so we modified it to be used also with encoders' measurements.

The last rule of the tracking thread is to decide whether a new Key-frame is needed or not. In ORB-SLAM3, there are only optical criteria. We also added movement criteria, because the pre-integrated measurements between key-frames will drift if they are far from each other. So we add new KF when transnational movement $> 10cm$ or rotational angle $> 10°$) based on the odometric measurements.

2) ***In the mapping thread:*** The mapping thread optimizes the key-frames using bundle adjustment. We implemented a modified version of BA function in ORB-SLAM3 where the camera parameterized directly on SE(2) and the out-of-SE(2) motion perturbation is also considered. The graph is structured as depicted in Fig. 1 using two edges mentioned in II-A. The local window in our case consists of 10 succeeding key-

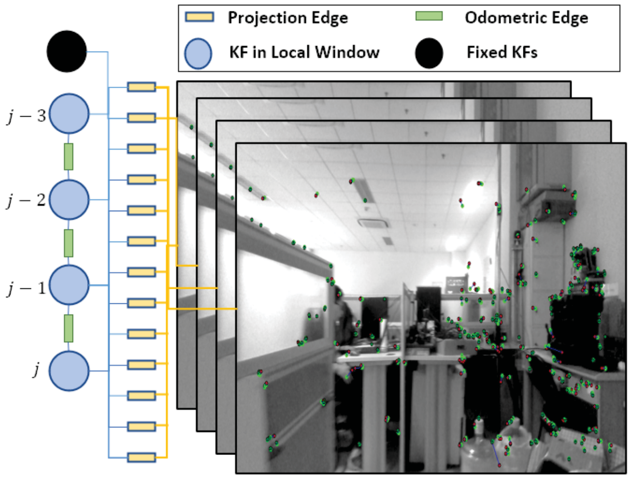

Fig. 1. Graph structure in Bundle Adjustment

frames connected subsequently by odometric Pre-integration constraint (Fig. 1: The odometric edge). The local window also includes all the map points that are seen from those Key-frames through observations. These observations represent the projection error modeled using an edge in the graph optimization, which connects Key-frame with a map point (Fig 1: The projection edge). The bundle adjustment also includes all Key-frames that observe the map points but are not included in the local window as fixed optical constraints (figure 1). We also modified the Key-frame culling process, which re-moves redundant Key-frames after using them in optimization. We cull a key-frame located in the current local window as long as its removal still satisfies a convenient interval of translation or rotation between previous and next key-frames. Key-frames located outside the local window are culled as in the standard ORB-SLAM3 system.

*3) **In the loop closing thread**:* Loop closure process is purely optical process. It detects cases where the robot has traversed in some old recognizable places. If the system detected some loop, then the closure process is done over 3 steps:

- Correcting local key-frames using **SIM3 Solver** satisfying the detected loop.
- Optimizing the essential graph, which preserve the relative constraints between key-frames. We modified here by adding subsequent constraints of odometric preintegrated measurements over the current local window.
- Global Bundle Adjustment: Here we apply global BA with the edges described in II-A.

III. Outliers Rejection Model:

The fast development of visual processing of scenes, either by low-level image processing techniques or by AI approaches, has led to a great contribution to SLAM community. Visual processing can be classified into two classes: geometric-based and AI-based. Geometric-based approaches use low-level information in the image like pixel values (either it is a color image or a depth image), where AI-based methods can utilize a higher level of information like objects in the scene.

SLAM expects the surrounding scene to be static. ORB-SLAM3, similar to other basic SLAM systems, is not robust to dynamic environments, and some additional models should be implemented to filter out the moving objects. We realized a geometric-AI model for detecting moving objects and remove optical features located on them.

1) The AI-Based approach: We used the semantic segmentation model **SegNet** [1] for detecting and segmenting the possibly moving objects. SegNet classifies objects semantically in the scene. We filter out detected people, animals, cars, buses, bicycles, etc... The SegNet model can process images at moderate frequencies, and it can perform real-time segmentation to some level. However, We applied the SegNet segmentation only over the key-frames. When the criteria of creating new key-frame in the tracking thread are satisfied, we process the image in the SegNet model and remove all key points that are located on possibly moving objects (Fig. 4).

2) The Geometric-Based approach: As SegNet provides a real-time segmentation and detection of possibly moving objects, the accuracy of the segmentation is not high. That's why we need to detect the points that are moving and not detected by SegNet.

The RGBD sensor provides a colored monocular image with the corresponding depth data. In the case where the mobile robot has an RGBD sensor, we do geometric clustering using the K-Means algorithm over the depth data. The clustering operation can't identify whole objects, but parts of an object as depicted in figure 3. We search for possibly moving clusters by calculating the ratio of optical outliers located in this cluster. If the ratio of outliers ($>$ **50 %**), it means that the cluster is moving, and we remove all the optical features (Inliers & Outliers) located in this cluster. **Outliers** are defined by projection error on the new candidate key-frame of recognized optical features from the local map. If the projection error is bigger than some threshold (3-7) pixels, it is considered an outlier. This geometric filter is sensitive to moving objects not detected by the semantic segmentation model. Specifically, It detects optical features located on objects moved by people for example, which are considered static by the SegNet, i.e. a moved chair. It detects also the optical features on the borders of a moving object, due to not accurate segmentation (figure 3). We also show in table III, that the model preserve the real-time performance of ORB-SLAM3, and working as fast or faster than other models. As our objective is not to compare with running time of other approaches and on other devices, but to show the capability that the proposed solution operates in real-time.

The geometric model also includes the epipolar constraint between two subsequent key-frames for filtering not correctly matched points (relatively-moving features won't satisfy the epipolar constraint), in case these points were not detected on moving object neither on moving cluster.

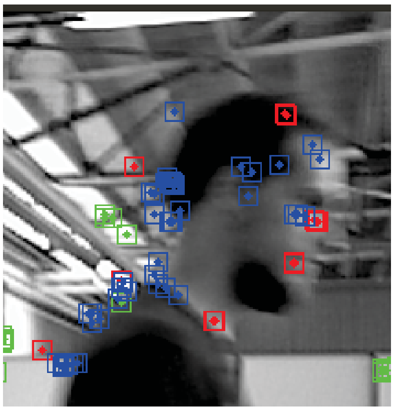

Fig. 2. Detecting Points on moving objects (Person)

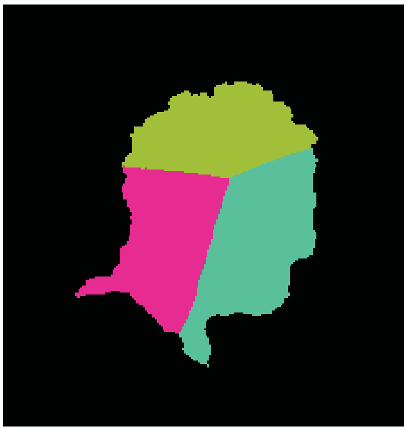

Fig. 3. KMeans - Geometric-based approach (High border details)

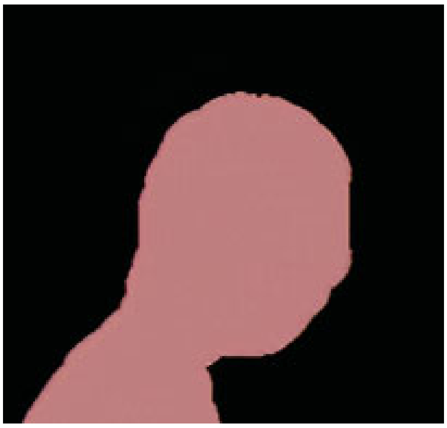

Fig. 4. SegNet Model: AI-based approach (Semantic segmentation and object detection)

IV. Evaluation

We split the evaluation into two sets of experiments: First, we test the optimization constrained modification over the ORB-SLAM3 system and compare it with the monocular inertial results over the OpenLoris dataset [23]. and second, we test the efficiency and robustness of the dynamic model independently from the constrained optimization on TUM-RGBD [24]. Both took place on Intel i7 CPU, RTX2060 (Notebook) GPU, RAM 16 Gb.

1) Evaluation of Constrained Optimization: We chose the OpenLoris dataset for evaluation for several purposes. The OpenLoris dataset is universal, including a diverse set of sensors (monocular (pinhole, fish-eye) /Stereo (fish-eye) vision, gyroscope, accelerometer, encoders, RGBD sensor). OpenLoris is also a popular platform for competitions of SLAM for mobile robots[1]. We couldn't find any published results of ORB-SLAM3 (monocular inertial case) for the OpenLoris dataset. So, we tested ORB-SLAM3[2] with monocular pinhole inertial default settings. We found out that ORB-SLAM3 is not working fine, because we got bad trajectories. So, we decided to test ORB-SLAM3 with monocular **fish-eye** inertial default settings. Fish-eye model provides bigger FoV and therefore more robust and accurate tracking. In Table I, we provide the average of (rotational/translations) RMSE error[3] of 5 executions of ORB-SLAM3 system on the OpenLoris dataset. We found that one of the main factors of the failure of the State-of-The-Art ORB-SLAM3 system on OpenLoris dataset is the bad inertial initialization, due to unobservable scale and the constant acceleration (rotational only/transitional only) movement of the mobile robot [16]. We tested our modification over the ORB-SLAM3 system using the constrained optimization (monocular fish-eye & wheel odometry) also by running 5 executions for each sequence and calculating the average (transitional) ATE error, showing more accurate and robust results over different sequences and places (Table I). The most noticeable improvement is depicted in figure 5, where our trajectory over Z axis shows better accuracy and robustness than ORB-SLAM3. We also provide in the Table I results of the 1st place of the **IROS 2019 Lifelong Robotic Vision Challenge: Lifelong SLAM**, were all sensors are available to use in the challenge of OpenLoris. The Table I shows that our results are robust and in most cases better than the best results achieved in the challenge.

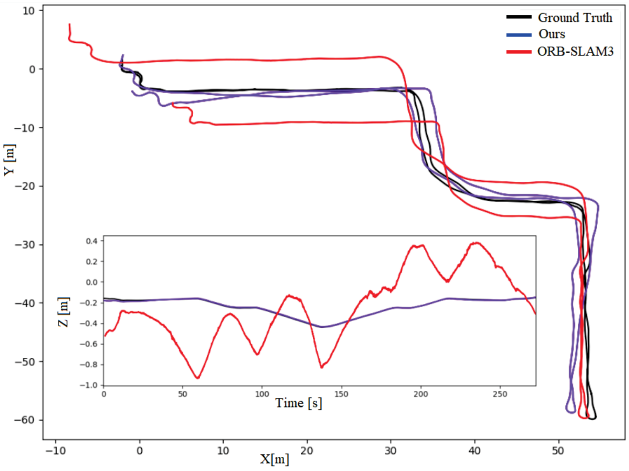

Fig. 5. XY trajectory of Corridor 01, with Z trajectory (Bottom Left)

2) Evaluation of The Outliers' Rejection model in Dynamic Environments: For evaluation of our outliers' rejection model, we tested on the TUM-RGBD dataset [24]. It provides RGBD sequences with different scenarios of moving objects in front of a static or moving camera. TUM-RGBD is also suitable to compare with other models in the literature by robustness, accuracy, and execution cost. We tested our modification of the ORB-SLAM3 system without the constrained optimization

[1] https://competitions.codalab.org/competitions/21504#result
[2] V0.3: Beta version, 7 September 2021
[3] https://docs.openvins.com/eval-error.html

TABLE I. Performance comparison of the ATE (position) (m.) between Ours (Monocular-Wheel Odometry), ORB-SLAM3 (Monocular-Inertial) and the 1st place winner of competition on OpenLoris Dataset [23]

Sequence	ORB-SLAM3	1st place	Ours
office1-1	0.063	0.172	**0.030**
office1-2	0.076	0.936	**0.036**
office1-3	0.028	0.732	**0.022**
office1-4	0.080	0.673	**0.045**
office1-5	0.227	0.515	**0.083**
office1-6	0.067	0.459	**0.026**
office1-7	0.060	0.854	**0.030**
Avg	0.086	0.620	0.039
home1-1	0.424	**0.172**	0.223
home1-2	0.371	**0.240**	0.301
home1-3	0.351	0.158	**0.118**
home1-4	0.299	0.171	**0.091**
home1-5	0.257	0.254	**0.077**
Avg	0.340	0.206	0.162
cafe1-1	**0.102**	0.232	0.201
cafe1-2	**0.115**	0.230	0.432
Avg	0.108	0.231	0.317
corridor1-1	5.040	**1.032**	1.841
corridor1-2	6.119	**0.675**	2.251
corridor1-3	11.273	1.320	**0.190**
corridor1-4	2.916	1.270	**0.244**
corridor1-5	4.464	0.964	**0.442**
Avg	5.962	1.052	0.993
market1-1	15.771	**1.073**	2.712
market1-2	9.126	**1.216**	5.381
market1-3	11.431	**1.432**	4.534
Avg	12.109	1.240	4.209

TABLE II. Comparison of the RMSE RPE in translation drift over TUM-RGBD [24] dataset. Best results are highlighted in bold and the second-best are underlined.

Sequence	Translation RPE (m/s)			
	ORB_SLAM3 (RGB-D)	DS-SLAM	Detect Slam	Ours (G+AI)
walking_xyz	1.251107	0.0333	**0.0241**	0.046710
walking_rpy	1.517069	0.1503	0.2959	**0.040565**
walking_half	1.055122	**0.0303**	0.0514	0.040534
walking_static	0.553423	**0.0102**	-	0.011880
sitting_xyz	**0.016737**	-	0.0201	**0.016970**
sitting_rpy	**0.031859**	-	-	0.032727
sitting_half	0.037480	-	**0.0231**	0.024585
sitting_static	0.015077	-	-	**0.009412**

TABLE III. Overview of the running time for the proposed solution and other existing approaches, showing that it works in real-time

Methods	AI	Geometric	Tracking	Hardware
ORB SLAM3	-	-	22.2322	CPU only
DS-SLAM	75.64	47.38	148.53	Intel i7 CPU P4000 GPU
Dyna SLAM	884.24	589.72	1144.93	Titan X GPU
Detect$_4$ Slam	310.0	20.0	-	Intel i7-470 GTX960M GPU
Ours	69.3642	26.6683	22.837	Intel i7 CPU RTX2060 GPU

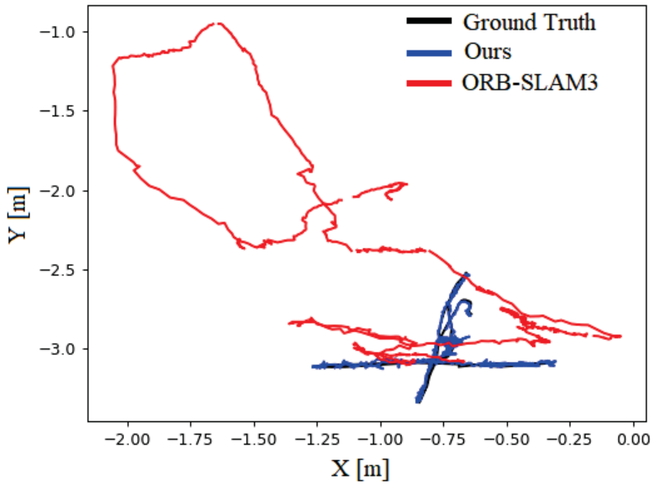

Fig. 6. Trajectory comparison on TUM-RGBD walking-XYZ sequence

part, to see independently how well the model is performing. This outliers' rejection model provides ORB-SLAM3 with robustness and accuracy in the case of tracking in a dynamic environment. We show in Table II, the results of our model compared to ORB-SLAM3 and other works in the literature. It is shown that our model is robust over all the sequences we tested, and gets either the best or second best accuracy in most cases. We also show two trajectories of ORB-SLAM3 with/without our dynamic model (figure 6), and it is clear that ORB-SLAM3 system is not robust to dynamic environments and it works better with our modifications. We also show in table III that our modification preserve the real-time performance of the ORB-SLAM3 system.

V. Conclusion

We addressed the robustness of the visual SLAM systems designed for ground wheeled robots operating in real-world like dynamic environments. We benefits from the constrained movement on Z-axis by adding constraints based on encoder measurements in the optimization process, showing that Visual Encoder SLAM is more robust than Visual Inertial one for wheeled robots. For more robustness, we realized an outliers rejections model for detecting moving features. This model consists of Geometric and AI-based methods that are complementary for achieving high robustness.

The work was built upon the state-of-the-art ORB-SLAM3 system and has been tested on OpenLoris and TUM-RGBD showing better results in terms of robustness and accuracy, while still performing SLAM in real-time.

In the future, fusing wheel odometry with inertial data for the robust inertial initialization and overcoming wheel slippage error will be implemented to achieve a better state estimation providing more accuracy for visual SLAM system in mobile robots. The semantic model can be extended for improving place recognition and performing better loop closing.

REFERENCES

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," 2016.

[2] "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam."

[3] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[4] T. Schneider, M. T. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "maplab: An open framework for research in visual-inertial mapping and localization," *IEEE Robotics and Automation Letters*, 2018.

[5] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A research platform for visual-inertial estimation," in *Proc. of the IEEE International Conference on Robotics and Automation*, Paris, France, 2020.

[6] "Orb-slam: A versatile and accurate monocular slam system," vol. 31.

[7] "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," vol. 33.

[8] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[9] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[10] V. Usenko, N. Demmel, D. Schubert, J. Stueckler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters (RA-L) & Int. Conference on Intelligent Robotics and Automation (ICRA)*, vol. 5, no. 2, pp. 422–429, 2020.

[11] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 3565–3572.

[12] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org, 2010.

[13] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 3607–3613.

[14] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," 2012.

[15] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, "Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group," 2017.

[16] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis, "Vins on wheels," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5155–5162.

[17] D. Yang, S. Bi, W. Wang, C. Yuan, W. Wang, X. Qi, and Y. Cai, "Dre-slam: Dynamic rgb-d encoder slam for a differential-drive robot," *Remote Sensing*, vol. 11, no. 4, 2019.

[18] F. Zheng, H. Tang, and Y.-H. Liu, "Odometry-Vision-Based Ground Vehicle Motion Estimation With SE(2)-Constrained SE(3) Poses," *IEEE Trans. Cybernetics*, vol. 49, no. 7, 2019.

[19] F. Zheng and Y.-H. Liu, "Visual-Odometric Localization and Mapping for Ground Vehicles Using SE(2)-XYZ Constraints," in *Proc. IEEE Int. Conf. Robot. Autom (ICRA)*, 2019.

[20] "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," vol. 3.

[21] "Ds-slam: A semantic visual slam towards dynamic environments."

[22] F. Zhong, S. Wang, Z. Zhang, C. Chen, and Y. Wang, "Detect-slam: Making object detection and slam mutually beneficial," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1001–1010.

[23] X. Shi, D. Li, P. Zhao, Q. Tian, Y. Tian, Q. Long, C. Zhu, J. Song, F. Qiao, L. Song, Y. Guo, Z. Wang, Y. Zhang, B. Qin, W. Yang, F. Wang, R. H. M. Chan, and Q. She, "Are we ready for service robots? the openloris-scene, datasets for lifelong slam," 2019.

[24] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, 10 2012.

[25] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.