Solving Short-Term Relocalization Problems In Monocular Keyframe Visual SLAM Using Spatial And Semantic Data

Azmyin Md. Kamal[1], Nenyi Kweku Nkensen Dadson[1], Donovan Gegg[1], Corina Barbalata[1,*]

Abstract— In Monocular Keyframe Visual Simultaneous Localization and Mapping (MKVSLAM) frameworks, when incremental position tracking fails, global pose has to be recovered in a short-time window, also known as short-term relocalization. This capability is crucial for mobile robots to have reliable navigation, build accurate maps, and have precise behaviors around human collaborators. This paper focuses on the development of robust short-term relocalization capabilities for mobile robots using a monocular camera system. A novel multimodal keyframe descriptor is introduced, that contains semantic information of objects detected in the environment and the spatial information of the camera. Using this descriptor, a new Keyframe-based Place Recognition (KPR) method is proposed that is formulated as a multi-stage keyframe filtering algorithm, leading to a new relocalization pipeline for MKVSLAM systems. The proposed approach is evaluated over several indoor GPS denied datasets and demonstrates accurate pose recovery, in comparison to a bag-of-words approach.

I. INTRODUCTION

Monocular Keyframe Visual Simultaneous Localization and Mapping (MKVSLAM) is an optimization-based framework used in mobile robotics. This approach computes the robot's pose while simultaneously creating a sparse-reconstruction of the environment incrementally for each keyframe. A keyframe contains compact description of the observed space (no actual image is stored) and is interlinked with others in a directed graph called a pose graph. When incremental tracking is interrupted, the global pose must be recovered in a short period of time to maintain the accuracy of the robot's localization and the map usability. In this paper, this event is defined as the short-term relocalization problem.

An important step in recovering global position, in this framework, is to choose a number of appropriate keyframes from the pose graph. Additionally, information that describes the physical space where incremental tracking was interrupted, is useful for subsequent pose recovery [1]. This step is defined as the Keyframe-based Place Recognition (KPR). In this regard, each keyframe requires a unique description that differentiates it from other keyframes in the pose graph.

Several MKVSLAM frameworks choose keyframes for pose recovery using the bag-of-words (DBoW2) method [2]. In this method, keyframes are described using collections of non-semantic local features. However, the bag-of-words method has shown limited success for short-term relocalization since the events that trigger tracking loss decrease the
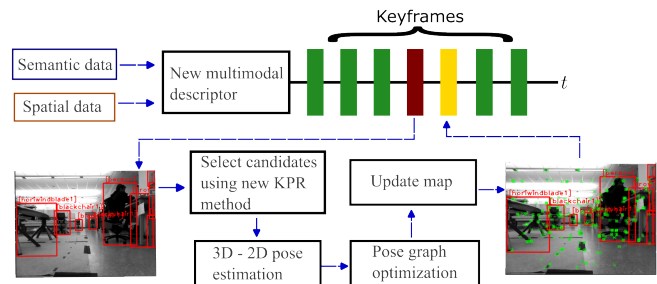

Fig. 1: High level system overview: For each keyframe (colored rectangles) the proposed multimodal descriptor is formed using semantic and spatial data. When tracking is lost in the red keyframe, the proposed KPR method selects a number of keyframes from the pose graph. By solving a series of correspondences between 3D map points (in a candidate keyframe) to 2D keypoints (in query keyframe), an estimation of the global pose is found. Followed by a pose graph optimization step, global pose is recovered in the yellow keyframe and incremental tracking resumes in green keyframes.

number of local features identified. Furthermore, DBoW2 does not contain spatial data [3].

Some MKVSLAM frameworks have integrated human-understandable semantics into their pipeline. In these works, semantic data offers additional information for geometric landmarks such as points, planes, and objects [4] and can provide reliable solutions for people tracking [5].

This paper leverages the advancements in semantic data extraction using deep neural networks and presents a fast and accurate solution to the short-term relocalization problem. The proposed approach introduces a novel multimodal keyframe descriptor consisting of semantics and spatial data from monocular images. This is integrated into a KPR method that chooses appropriate keyframes candidates by passing them through a multi-stage filtering algorithm. The novel KPR is coupled with an off-the-shelf 3D-2D pose estimation technique to create a novel relocalization pipeline that is shown in Fig. 1. By incorporating the novel relocalization method into ORB-SLAM3 [6], an open-source MKVSLAM framework, the robustness and efficiency of the proposed approach is demonstrated for aerial and ground robot datasets operating in GPS-denied environments.

The contributions of this paper are as follows:

- A new keyframe descriptor called the Pose Semantic Descriptor (PSD) is proposed. It utilizes semantic data and camera pose to uniquely characterize keyframe objects in the pose graph. Using this descriptor, a novel

*Corresponding author
[1]The authors are with the Department of Mechanical & Industrial Engineering, at Louisiana State University, Baton Rouge, LA 70803, USA akamal4@lsu.edu, ndadso1@lsu.edu, dgegg1@lsu.edu, cbarbalata@lsu.edu

KPR algorithm called the Pose-Class-Box (PCB) is formulated.

- The integration of the proposed KPR method for improving short-term relocalization within the ORB-SLAM3 MKVSLAM framework.

The remainder of the paper is organized as follows: Section II discusses related keyframe descriptors and global pose recovery methods based on keyframes. Section III presents the proposed PSD descriptor, the KPR method, and how they are integrated within a MKVSLAM system. In Section IV, the experimental results are discussed. Section V summarizes findings and discusses future work.

II. RELATED WORK

Descriptors for keyframes: A variety of MKVSLAM frameworks using local feature [6], feature learning [9], or object detection [10] have adopted the visual bag-of-words model to describe keyframes. This descriptor is created by extracting local features using algorithms such as Oriented FAST and Rotated BRIEF (ORB) [11] or Scale-Invariant Feature Transform (SIFT) [12], and aggregating those features into a collection of visual words based on a pretrained vocabulary [2]. While it is pose-invariant, efficient in traversing the pose graph, and chooses optimal keyframes for loop closures [1], it lacks any spatial or semantic data. This additional information has been shown to improve localization when there is a large viewpoint difference, low texture, or noise [3]. Neural network based approaches aim to learn on how to extract local features [13], update the vocabulary online [14], and compute descriptors as a function of the image [15].

Relocalization approaches: A fast global pose recovery method was introduced in [1] where keyframes were uniquely identified using the bag-of-words model. Here non-semantic local features were aggregated into visual words [16] using a pretrained dictionary. When tracking fails, a KPR method retrieved a number of candidate keyframes based on a normalized L1 score between the query and candidate keyframes. No semantic data was considered in this approach. Bruno and Colombini integrated the Learning Invariant Feature Transform (LIFT) [17] with ORB-SLAM1 to form a hybrid monocular Visual Simultaneous Localization and Mapping (VSLAM) system [13]. A drone and an outdoor ground robot dataset were used in their testing. The approach demonstrated higher positional accuracy when compared with ORB-SLAM1. However, this method focused on the feature extraction and still relied on the bag-of-words approach for relocalization. In [18], class labels of objects were detected using YOLO bounding-box regressor [19] and were used to form an object label array. An *object similarity* score was computed by comparing the class label arrays between query and candidate keyframes to perform place recognition. No spatial information (pose, bounding box) was used, and the size of the object label array is needed to be known apriori.

Semantics for localization: In [20] a metric map was augmented with the 3D poses of objects detected using a neural network. Here, relocalization was demonstrated as a long-term data association problem where global pose was recovered by aligning two semantic maps recorded for the same environment but taken at two different instances. In [21] two approaches for accurate pose estimation in human populated indoor environments were presented. Using YOLO [19] to detect objects and Mask-RCNN [22] to perform semantic segmentation, the approach was integrated with ORB-SLAM2 MKVSLAM framework. Another work building on top of ORB-SLAM2 was proposed in [9] where semantic segmentation data and a geometric constraint were used to develop a new local keypoints extraction method. This method filtered out keypoints belonging to moving objects to improve position estimation accuracy and the semantic data was used create a dense metric-semantic map. Similar to the above, in DynaSLAM [23] and DyOb-SLAM [24], moving objects were semantically segmented to remove the keypoints associated with them. This was shown to jointly improve camera pose estimation, and tracking position of mobile objects in the active map.

III. METHODOLOGY

Figure 2 shows the framework for the short-term relocalization approach. The method uses an Object Detection module to extract semantic data, shown in light purple, based on a Deep Neural Network (DNN) architecture. This data is passed into a MKVSLAM system (shown in light blue). In the following paragraphs, the new keyframe descriptor is presented and the keyframe-based place recognition approach is detailed.

A. Pose Semantic Descriptor in the context of MKVSLAM

The proposed approach starts by extracting meaningful information from the environment and generates semantic data based on identified objects. That is, at each timestep, $n : n \in \mathbb{Z}^+$, an RGB image $\mathbf{I_n}$ is passed through the Object Detection module that contains a DNN object detector as described in [19]. The output is a matrix named here the *semantic matrix* $\mathbf{SM_n} \in \mathbb{R}^{b \times 5}$ where $b \in \mathbb{Z}^+$ is the number of objects detected. Each row of this matrix contains the class label and the 2D bounding box coordinate of an object described by the top-left and bottom-right corners in the image coordinate system. The class labels are defined using an unique integer number.

A copy of the original image, $\mathbf{I_n}$, and the semantic matrix, $\mathbf{SM_n}$, are passed to the VSLAM system (in light blue) were a frame object, $\mathcal{F}_n = \{\mathbf{I_n}, \mathbf{SM_n}, \{x_n^{z_1}\}, \mathbf{T_{wn}}\}$ is created. Here, $\{x_n^{z_1}\}$ are the 2D non-semantic features extracted from the image (keypoints), and $\mathbf{T_{wn}}$ is an initial estimate of the camera pose represented in $SE(3)$. $\mathbf{T_{wn}}$ is refined over time using Bundle Adjustment (BA) [5].

The SLAM module has modes of operations (referred to as states) as depicted with colored arrows in Figure 2. The green path, defined as a tracking state, refers to the continuous simultaneous estimation of the camera pose and environment mapping. The orange path, defined as a lost state, refers to the state where the system attempts to recover from global
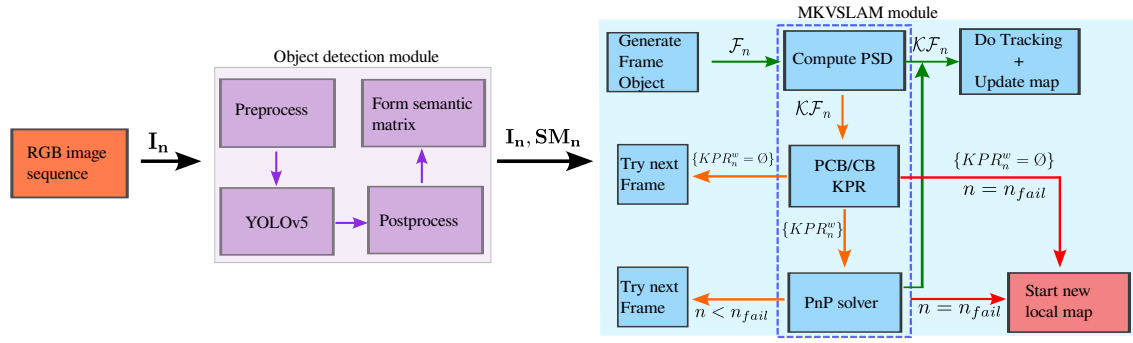
Fig. 2: Overview of the full framework: An image is passed through the purple block where the *preprocess* block is resizing the image, that is next fed to the YOLOv5 object detector [7]. Next the *postprocess* block performs non-maximal suppression [8] on the detection to finalize class predictions for the detected objects. In light blue the MKVSLAM module is presented. The path in green shows the active tracking state. The path in orange shows a relocalization attempt in a lost state. The path in red indicates a full recovery failure. The relocalization pipeline identified by the dashed deep blue rectangle consists of computing the PSD descriptor of the query keyframe, choosing optimal candidates using the proposed KPR method and recovering global camera position by solving a series of Perspective-n-Point (PnP) problems.

position loss. This recovery must happen within a certain number of time steps (n_{fail}) from the moment that tracking loss occurs. If the global pose is not recovered within n_{fail} time steps, the systems follows the red path defined as a failure state. In the failure state, the current map closes and a new map is initialized. Each active map denoted by $\mathcal{M}^*$ is defined as a local map. One session may contain multiple local maps but only one is active at any given time. In the tracking and lost states, a frame object is used to create a keyframe object $\mathcal{KF}_n = \{\mathcal{PSD}_n, \mathbf{X_n}\}$, where $\mathbf{X}_n \in \mathbb{R}^3$ are the initial position estimates of 3D map points associated with $\{x_n^{z_1}\}$ keypoints. If no objects are detected, then $\mathcal{PSD}_n$, is not generated and no keyframe is created from frame $\mathcal{F}_n$.

To increase the knowledge of the system in defining a keyframe object for fast and accurate candidate selection, this paper presents the novel multimodal keyframe descriptor called the Pose Semantic Descriptor (PSD). The proposed descriptor is formulated as the following data structure:

$$\mathcal{PSD}_n = \{n, \{sl_n^k\}, \mathbf{B_n}, \mathbf{T_{wn}}\} \tag{1}$$

where, n is the timestep, $\{sl_n^k\}$ denotes the set of class labels for the k objects detected. $\mathbf{B_n} \in \mathbb{R}^{k \times 4}$ are their 2D bounding box coordinates, and $\mathbf{T_{wn}}$ is the camera pose. The multimodal nature of the descriptor allows the semantic (objects detected and associated class labels) and geometric data (camera pose) to complement each other for place recognition.

B. Pose-Class-Box Keyframe-based Place Recognition (KPR) method

In the following paragraphs a new KPR method is presented, named the *Pose-Class-Box* approach. This is activated during a lost state and is formulated as a three-stage filtering algorithm.

In the first stage, spatial data is used to define a *pose constraint*. Consider the pose of the camera represented by

$\mathbf{T} \in \mathbb{R}^{4 \times 4}$:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & s.\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & s.t_x \\ r_{21} & r_{22} & r_{23} & s.t_y \\ r_{31} & r_{32} & r_{33} & s.t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

where $\mathbf{R} \in \mathbb{R}^{3 \times 3} \in SO(3)$ is the rotation matrix, $s \in \mathbb{R}$ is the scale factor, and $\mathbf{t} = [t_x, t_y, t_z]^T \in \mathbb{R}^3$ is camera's translation in the world coordinate frame w. $\mathbf{T}$ is non-singular matrix, independent of the scale factor s.

For two camera poses $(\mathbf{T_{w1}}, \mathbf{T_{w2}})$ that are similar, the following relation is defined:

$$\begin{aligned} \text{if} \quad & \mathbf{T_{w1}} \approx \mathbf{T_{w2}} \Rightarrow \\ & \delta T_{1,2} = |\ ||\mathbf{T_{w1}}||_F - ||\mathbf{T_{w2}}||_F\ | \in \mathbb{R}^+ \approx 0 \end{aligned} \tag{3}$$

where $||\cdot||_F$ is the Frobenius norm [25]. Based on Equation (3), in this work the geometrical similarity between two keyframe poses is represented by one real number. This is computed pairwise between the query keyframe and all other keyframes in the pose graph. For a given pair of keyframes at times n and l, let this number be denoted $\delta T_{n,l}$ and computed as:

$$\delta T_{n,l} = ||\mathbf{T_{wn}} - \mathbf{T_{wl}}||_F = \left(\sqrt{\sum_{i=1}^{n} \sum_{j=1}^{m} |a_{ij}|^2} \right) \tag{4}$$

Using the above equation, the procedure of selecting keyframe candidates from the pose graph in the active map $\mathcal{M}$ has to satisfy Equation (5). This is defined as the *pose constraint* step.

$$\epsilon \le \delta T_{n,l} \le \delta T_{th}, \tag{5}$$

where δT_{th} is a predefined threshold and $\epsilon = 10^{-4}$ is to account for numeric errors. The list of candidates after the first filtering stage is marked by $\{P_n^j\}$. The *pose constraint* step can be described as selecting candidates within the search sphere centered around the query keyframe $\mathcal{KF}_n$ with

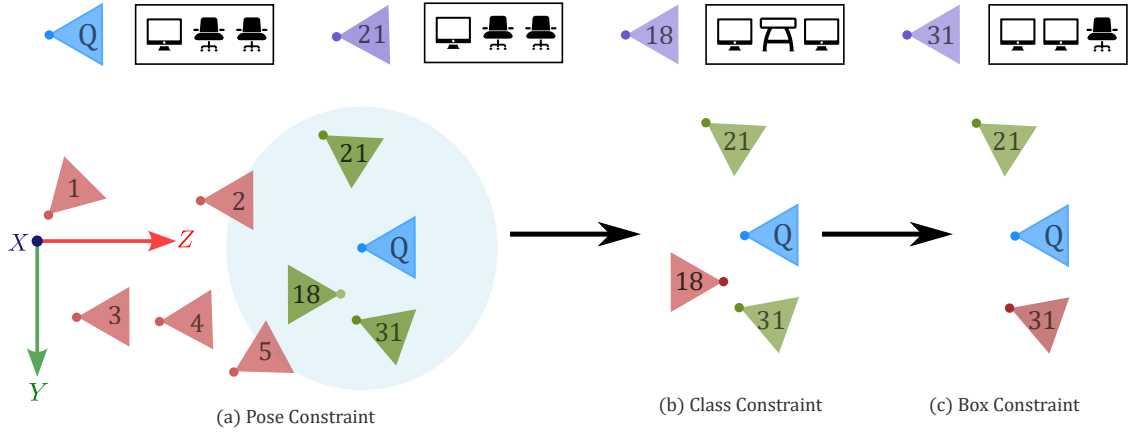(a) Pose Constraint (b) Class Constraint (c) Box Constraint

Fig. 3: Application of proposed KPR method in a 2D case. The top row shows the collection of objects seen by the query keyframe and some of the candidate keyframes. Colored triangles in active map represents keyframes, with red being identified those that aren't chosen by the proposed approach. Keyframes that are selected in each of the three steps are colored in green. The shaded blue circle is the search sphere in the *pose constraint*. In the *class constraint* only keyframes 21 and 31 are selected as they contain the same type of objects as the query keyframe (identified by blue triangle). The *box constraint* retains only keyframe 21 as the location of the observed objects matches the query keyframe.

a radius δT_{th} (shaded-blue circle in Fig. 3). This keyframe is created by computing its PSD and retaining 3D map points $\mathcal{X}_n$ from frame $\mathcal{F}_n$.

Note that, only using the *pose constraint* described above, a candidate is selected from search sphere even if it's orientation is significantly different from $\mathcal{KF}_n$ (blue triangle in Fig. 3). This can lead to choosing keyframes ill-conditioned to recover camera pose using 3D-2D pose estimation techniques. To address this limitation, semantic data is incorporated into the proposed KPR approach in the second stage, *class constraint*. This is detailed below.

Let the list of classes of objects be $\{c_n\}$ in the query keyframe and $\{c_j\}$ in the jth candidate keyframe. As the class labels are represented by unique integer numbers, a numeric score, $\delta C_{n,j} \in \mathbb{R}^+$, describing the similarity of the class labels between $\{c_n\}$ and $\{c_j\}$ is computed as:

$$\delta C_{n,j} = |\|\{c_n\}\|_2 - \|\{c_j\}\|_2| \quad (6)$$

The above numeric score is computed pairwise between the query keyframe and all the l candidates in $\{P_n^j\}$. Each pairwise score is stored in the list $\{C_n^j\}$. A keyframe candidate is considered semantically relevant if the below condition is fulfilled:

$$\delta C_{th} \le \delta C_{n,j} \le \delta C_{th} + 0.1\delta C_{th} \quad (7)$$

where δC_{th} is an adaptive threshold computed as:

$$\delta C_{th} = \min\left(\{C_n^j\}\right) \quad (8)$$

The list of candidates after the second filtering stage is marked by $\{R_n^j\}$. This formulation has two advantages. Firstly, it is order independent, i.e the same score is obtained regardless the order the class labels appear in the semantic matrix. Secondly, the number of objects recorded in the two keyframes is not required to be same.

However, representing classes as integers may lead to some ambiguous cases resulting in wrong keyframe selections. For example, if $\{c_n\} = \{1,1,1,1\}$ and $\{c_j\} = \{4\}$ then both δC_{th} and $\delta C_{n,j}$ are equal leading to a wrong selection. To address this issue and further reduce the pool of candidates, the last stage of the KPR approach is focused on filtering keyframes based on the location of the bounding boxes of the objects in the image coordinate frame, referred as the *box constraint* step.

If the jth candidate satisfies Equation (8), let $\mathbf{B_n} \in \mathbb{R}^{b_n \times 4}$ be the bounding box coordinates of objects recorded in the query keyframe and $\mathbf{B_j} \in \mathbb{R}^{b_j \times 4}$ represent the bounding box coordinates of objects recorded in the jth candidate. The Intersection-over-Union (IoU) scores [26] between the bounding boxes in the query keyframe and all the candidates in $\{R_n^j\}$ is computed. To establish the objects that are visible both in the query and candidate keyframes, a threshold is used:

$$\text{if} \quad \%IoU > \delta IoU \Rightarrow v_j = v_j + 1 \quad (9)$$

where $\%IoU$ is the percentage overlap, δIoU is a predefined threshold, and v_j is a counter for objects that overlap in the scene. If a certain number of objects have been identified as being in both keyframes, the candidate is considered relevant for relocalization. This can be formulated as:

$$\text{if} \quad v_j \ge b_n \Rightarrow j^{th} \text{ keyframe} \in \{KPR_n^w\} \quad (10)$$

where b_n is the number of objects in the query keyframe and $\{KPR_n^w\}$ is the final candidate list.

However, if global pose is not recovered at the exact n^{th} timestep, then frames $\mathcal{F}_{n+1}, \mathcal{F}_{n+2}, ...$ have their camera pose initialized to an identity matrix, $\mathbf{I}_{4 \times 4}$. This is due the fact that in a monocular MKVSLAM system, no other information beyond image data is available to estimate camera pose. In these frames, computing $\delta T_{n,l}$ using Equation (4) will result

in computing the Euclidean distance of $\mathcal{KF}_n$ from origin only, resulting in the *search sphere* pulling candidates that may have no semantic or geometric relevance. In this case, the *pose constraint* step should be skipped, and only the *class constraint* and *box constraint* steps are used to select candidates from the pose graph in the active map $\mathcal{M}$. This is marked as the *Class-Box* KPR method. To select either the PCB or the CB KPR methods, the following conditional check is used:

$$\text{if} \quad \delta T_{L_2} = ||\mathbf{T_{wn}} - \mathbf{I}_4||_2 > 0 \Rightarrow \text{full PCB} \quad (11)$$

C. Proposed short-term relocalization method

The proposed short-term relocalization method is implemented as shown in Algorithm 1. The hyperparameters of the algorithm are ΔT_{th}, n_{fail}. Line 2 extracts all the keyframes $\{K_n^i\}$ in the pose graph at n^{th} timestep where i represents the number of keyframes. Line 5 computes the PSD descriptor as expressed in Equation (1). Line 9 calls for the full PCB method and line 11 calls the reduced version as described by Equation (11). Line 14 recovers the global position in the active map $\mathcal{M}$ by solving a series of 3D to 2D motion estimation problems using the Maximum Likelihood Perspective-n-Point (MLPnP) [27] solver. Here $bFlag$ is set to true if a pre-defined number of map points to keypoints correspondences are found. If $bFlag$ is true, tracking state is resumed (system follows the green path in Figure 2), else Algorithm 1 is repeated as long as recovery is not successful within n_{fail} timesteps. If this is not achieved, a new local map is initialized, as shown in lines 19-21.

Algorithm 1 Relocalization using PCB

Input: Frame $\mathcal{F}_n$, Map $\mathcal{M}$, timestep n, $\delta T_{th}, n_{fail}$

1: $n_r \leftarrow 1$
2: $\{K_n^i\} \leftarrow \mathcal{M}.GetAllKeyFrames()$
3: **for** $n_r < n_{fail}$ **do**
4:     $\mathcal{KF}_n \leftarrow KeyFrame(\mathcal{F}_n)$
5:     $\mathcal{KF}_n.ComputePSD()$
6:     **for** $j \leftarrow 1$ to i **do**
7:         $\delta T_{L_2} \leftarrow ||\mathbf{T_{wj}} - \mathbf{I_4}||$
8:         **if** $\delta T_{L_2} > 0$ **then**
9:             $\{KPR_n^w\} \leftarrow PCB(\mathcal{KF}_n, \{K_n^i\}, \delta T_{th})$
10:        **else**
11:           $\{KPR_n^w\} \leftarrow CB(\mathcal{KF}_n, \{K_n^i\}, \delta T_{th})$
12:        **end if**
13:     **end for**
14:     $bFlag \leftarrow PerformRelocalization(\{KPR_n^w\})$
15:     **if** $bFlag == True$ **then**
16:        **Return** $bFlag, \{KPR_n^w\}, \mathcal{KF}_n$
17:     **else**
18:        $n_r \leftarrow n_r + 1$
19:        **if** $n_r >= n_{fail}$ **then**
20:           **Stop** PCB
21:           **Terminate** $\mathcal{M}$
22:        **end if**
23:     **end if**
24: **end for**

IV. EXPERIMENT & RESULTS

This section details the algorithm implementation, experimental setup, and the obtained results.

A. Implementation details

The proposed approach is implemented and tested on a computer with Ubuntu 22.04, Intel i5-9300H @ 2.4GHz, NVIDIA RTX 2060 GPU, and 16 GB RAM. The PSD keyframe descriptor and PCB KPR method were incorporated in ORB-SLAM3. Multi-mapping, loop closure and four parallel threads were used and the proposed modifications targets the relocalization function in the Tracking, Local Mapping, and Keyframe Database modules in the ORB-SLAM3 framework [6]. For the object detection module, the YOLOv5L has been chosen due to its fast computational capabilities [7]. The complete system was implemented in Robot Operating System (ROS) [28]. The source code and the datasets used for this paper are made publicly available at the link below*. The parameters for the proposed relocalization method were set as follows: n_{fail} was set to 20 timesteps that approximately corresponds to one second, $\delta T_{th} = 0.5$ and $\delta IoU = 90\%$ were set based on trial and error.

B. Dataset

The experiment is performed on 18 indoor image sequences spanning across four unique datasets split into two drone and two ground robots datasets, as shown in Table I. Sequences for the drone dataset are taken from the *Euroc MAV* dataset [29]. The sequences present a drone maneuvering in different indoor environments containing variety of objects. This dataset has ground-truth camera poses. One ground robot dataset is taken from the *FR2 PIONEER SLAM* sequences in the *TUM RGB-D* dataset [30]. This dataset presents motion-blur in the images and discontinuities in the recording, that can induce the lost state. Ground-truth poses are available in this dataset. Another ground vehicle dataset is created for this project and referred as the *LSU iCORE Mono* dataset. This is an indoor dataset representing a robotics laboratory showcasing a cluttered environment, with multiple chairs, people, workbenches, other robots, whiteboards, etc. This dataset has no ground-truth camera poses. The YOLOv5L network is trained for the Object Detection module for each of the four datasets as shown in Table I. An NVIDIA RTX 3080 Ti was used to train all the networks.

C. Experiment

The proposed relocalization method, as introduced in Section III-C, is tested in all the 18 image sequence described before. In the sequences from *EuRoC MAV* and *LSU-iCORE-Mono* datasets, 5 lost states are artificially induced at random times. As the *FR2 PIONEER SLAM* dataset is prone to lost state occurrences, there was no need to artificially induce the lost state. Each sequence is run 10 times and a series of quantitative metrics are computed to evaluate the performance of the system. For comparison, the DBoW2 based relocalization method in ORB-SLAM3 is used as the baseline for analyzing the obtained results.

TABLE I: Description of the experimental dataset showing associated platforms, sequences used for testing relocalization methods, sequences used to train their respective YOLOv5L networks and the number of classes detectable.

| Dataset (abbrev.)	Platform	Sequences used for testing Relocalization methods	Sequences used to train Object detector	Detectable classes
Machine Hall (MH)	Drone	MH01, MH02, MH03, MH04, MH05	MH01, MH04	26
Vicon Room (VicR)		V101,V102, V103, V201, V202, V203	V102, V203	15
FR2 Pioneer SLAM (FR2PS)	Rover	FR2360, FR2PS1, FR2PS2, FR2PS3	FR2PS1, FR2PS2	17
LSU-iCORE-Mono (LiM)		ROBO0, ROBO1, ROBO2	ROBO0, ROBO1, ROBO2	17
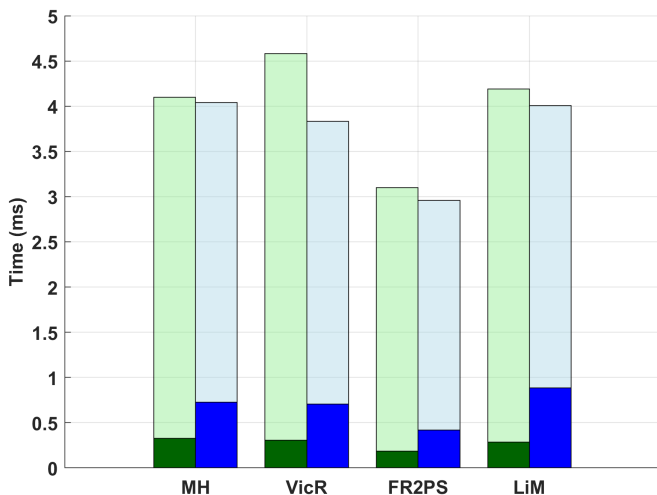
Fig. 4: Execution time for the KPR methods and the full relocalization method. Deep green bar represents the time for proposed KPR, deep blue for DBoW2 KPR, light green for the proposed relocalization method and light blue for the DBoW2 relocalization method. Best viewed in color.

D. Metrics

The Mean Average Precision (mAP) metric [31] is chosen to measure the detection performance of the Object Detection module. For evaluating the KPR method, the average execution time and the average keyframe candidates selected are recorded. The average total execution time, the average number of frames spent in lost state, and the average number of local maps created when pose recovery fails are used to study the relocalization method. To assess the impact of extended tracking loss Absolute Trajectory Error (ATE) [30] score is calculated. ATE scores are reported only for sequences that have ground-truth camera poses.

E. Results and Discussion

Prior to conducting the experiment, it is necessary to assess if the Object Detection Module has high precision. This is important to maximize the effectiveness of semantic information within the proposed relocalization method. The mAP scores are 82.9%, 83.5%, 84.0% and 91.0% for the *MH*, *VicR*, *FR2PS* and *LiM* datasets respectively. The average inference time is 29.55 ms which is lower than the average time needed by ORB-SLAM3 to process one image [6].

An important requirement for the proposed multimodal keyframe descriptor is to facilitate fast candidate retrieval when performing relocalization [1]. As depicted in Figure 4,

the proposed KPR method is faster than the DBoW2 method in all four datasets with an average execution time of 0.297 ms, being two times faster than the DBoW, that has an average execution time of 0.702 ms. The proposed KPR approach extracts on average five, six, four, and four keyframe candidates from the *MH*, *VicR*, *FR2PS*, and *LiM* datasets respectively. In comparison, the DBoW2 KPR selected, on average, only one candidate in all datasets.

These results show that the inclusion of both geometrical and semantic information into the keyframe descriptor improves its capabilities by allowing more keyframe candidates to be selected at a faster rate than the DBoW2 approach. The performance of the proposed relocalization method is shown in Table II. The proposed approach has an average execution time of 3.817 ms, while the DBoW2 relocalization method has an average execution time of 3.011 ms. Although the proposed KPR method demonstrates rapid keyframe selection, in average 0.297 ms, the main computational load during relocalization falls on the PnP solver. This is primarily due to KPR generating four times more keyframe candidates than the baseline DBoW2, leading to increased processing of 3D map points to 2D key point correspondences. Despite this, the proposed method maintains a real-time speed, approximately 4 ms. The proposed relocalization method performs better than the DBoW2 approach by reducing the time spent in the lost state by approximately 50%. This is shown in the the columns *Avg. timesteps in lost state* in Table II. Furthermore, the proposed approach, in average creates less local maps than the DBoW2 approach as shown in columns *Avg. local maps* in Table II. This shows that a richer description for the keyframes aids the overall system, by spending less time in the lost state. If a lost state persists, it is more likely to initialize a new local map that can lead to inaccuracies in pose estimation. In the following lines, the performance of the proposed relocalization method is discussed independently for each of the datasets. In the *MH* dataset, the proposed relocalization method shows good performance, spending only three timesteps in lost state and creating one local map. This leads to no recovery failures. In contrast the DBoW2 relocalization method spent four times as many time steps in the lost state and generated more local maps. The ATE for the proposed approach is 0.337 m, about 39% less in comparison to the DBoW2 approach. This indicates higher positional accuracy. In the *LiM* dataset, a similar performance is observed where the proposed relocalization method on average, spent eight timesteps in lost state and created three Local maps. In comparison, DBoW2

TABLE II: Performance of the proposed PCB relocalization in comparison to DBoW2 relocalization. Only for sequences that had ground-truth data, the ATE score and the number of frames matched to ground-truth are reported. Last row shows the median value calculated from the averages obtained in the 10 experimental runs.

Method	DboW2 Relocalization Module					PCB Relocalization Module				
Sequence	Avg. time (ms)	Avg. timesteps in lost state (int)	Avg. local maps (int)	ATE (m)	Frames retained for pose estimation (int)	Avg. time (ms)	Avg. timesteps in lost state (int)	Avg. local maps (int)	ATE (m)	Frames retained for pose estimation (int)
MH01	3.208	12	3	0.357	3483	**2.774**	**2**	**1**	**0.051**	**3642**
MH02	2.936	10	3	0.201	2843	**2.592**	**2**	**1**	**0.042**	**2901**
MH03	**3.570**	13	4	0.952	**2610**	4.692	**4**	**2**	**0.950**	2609
MH04	**2.957**	11	3	0.553	1854	4.551	**3**	**2**	**0.191**	**1948**
MH05	**3.907**	12	4	0.526	2070	4.239	**3**	**1**	**0.451**	**2169**
V101	**2.868**	7	2	0.146	2784	4.486	**4**	2	0.166	2784
V102	**3.265**	10	3	**0.111**	1505	3.455	**7**	**2**	0.131	**1587**
V103	**3.233**	13	4	**0.242**	**1877**	5.402	**8**	**3**	0.251	1872
V201	**2.779**	7	2	**0.068**	2080	4.737	7	2	0.077	**2084**
V202	**3.545**	10	3	0.078	2111	3.386	**7**	**2**	**0.068**	**2156**
V203	**3.093**	13	6	0.390	**1650**	4.193	**11**	6	**0.229**	1449
FR2360	**1.852**	14	3	0.481	1052	2.631	**12**	3	**0.383**	**1056**
FR2PS1	**2.819**	16	8	**0.346**	1152	2.958	**13**	**7**	0.347	**1295**
FR2PS2	**2.683**	**13**	5	0.154	**634**	2.745	15	5	**0.076**	510
FR2PS3	**2.804**	17	5	–	–	3.315	**15**	5	–	–
ROBO0	**2.885**	13	5	N/A	N/A	4.053	**9**	**4**	N/A	N/A
ROBO1	**3.065**	11	4	N/A	N/A	4.085	**6**	**2**	N/A	N/A
ROBO2	3.422	14	6	N/A	N/A	3.422	**9**	**4**	N/A	N/A
MEDIAN	**3.011**	13	4	0.294	1973	3.817	**7**	**2**	**0.178**	**2016**

relocalization method created five local maps, requiring 13 time steps to recover global pose.

However, in sequences from *VicR* and *FR2PS* datasets, the performance between the two relocalization methods is similar. These datasets feature sequences of a drone and a ground robot surveying two indoor rooms filled with obstacles. For the *VicR* dataset, the average number of timesteps in the lost state is seven, the number of new local maps generated is three, and the ATE score is 0.154 m when the proposed approach is used. Comparatively, for the DBoW2 approach, 10 timesteps are spent in the lost state, three local maps are created, and the ATE score is 0.173 m. For the *FR2PS*, the numbers were 14 timesteps, five Local maps, 0.269 m ATE score for the proposed. For DBoW2 approach they were 15 timesteps, five Local maps and 0.327 m ATE score respectively. In the sequence *FR2PS3*, a prolonged mapping failure towards the end of the sequence prevents recovery of global position by any method. Consequently, ORB-SLAM3 initiates a new *Atlas*, leading to deletion of the entire accumulated map.

This shows that the proposed relocalization method had inherited the limitation of BA based pose-graph optimization method that is known to fail ill-conditioned scenarios. Such cases are seen in the *VicR* and *FR2PS* datasets that show highly dynamic environments, and sometimes uncontrolled movements of the camera. However, even though both methods generated the same average number of local maps, the proposed method had a lower trajectory error in comparison to DBoW2 method.

Fig. 5a shows the performance of the system on the *V102* image sequence from *VicR* dataset. In this case the robotic system visited the same area several times. The pose estimation generated by both the proposed and the DBoW2 approaches show good agreement with the ground-truth. Fig. 5b shows the pose estimation for the *FR2PS1* image

sequence from the *FR2PS* dataset, where both relocalization methods struggled to recover global poses. This can be seen from the discrepancies between the ground-truth trajectory and the estimated ones. This is due to the sudden motion of the agent carrying the camera, an ill-conditioned scenario in which BA has limitations in recovering global pose [5].

The limitation of BA inherited by the proposed relocalization method may be solved using techniques such as Object SLAM [4], [10]. These methods are free from the requirement of the BA method that requires sufficient overlap in keypoints to map points observations between the query and candidate keyframes to optimize the camera's pose. Testing these methods are beyond the scope of this paper.

V. CONCLUSION

This paper proposes a new multimodal keyframe descriptor based on the semantic information of objects detected in the scene and camera poses calculated based on monocular images. A novel KPR method is developed to select appropriate keyframe candidates from the pose graph using a three stage filtering algorithm. These two components, coupled with a PnP solver, create a new short-term relocalization pipeline within the framework of the ORB-SLAM3. Across 18 GPS-denied image sequences collected with drones and ground robots, the proposed approach has exceeded the performance of the bag-of-words method; this being validated through a quantitative analysis. Future work will look at the integration of 3D object pose and the information theory to expand the proposed method into a multi-agent framework.

ACKNOWLEDGMENT
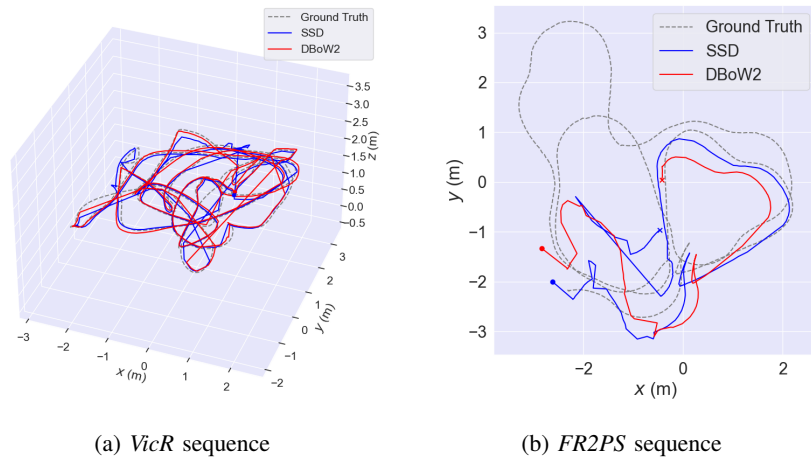
(a) *VicR* sequence

(b) *FR2PS* sequence

Fig. 5: Trajectory estimates with ORB-SLAM3 containing the modified relocalization method (blue) and DBoW2 version (red). Where available, ground-truth are shown in grey.

REFERENCES

[1] R. Mur-Artal and J. D. Tardós, "Fast relocalisation and loop closing in keyframe-based slam," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 846–853.

[2] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[3] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *ieee transactions on robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[4] B. Zhou, M. Gilles, and Y. Meng, "Structure slam with points, planes and objects," *Advanced Robotics*, vol. 36, no. 20, pp. 1060–1075, 2022.

[5] G. Younes, D. Asmar, E. Shammas, and J. Zelek, "Keyframe-based monocular SLAM: design, survey, and future directions," *Robotics and Autonomous Systems*, vol. 98, pp. 67–88, dec 2017. [Online]. Available: https://doi.org/10.1016%2Fj.robot.2017.09.010

[6] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: an accurate open-source library for visual, visual-inertial and multi-map SLAM," *CoRR*, vol. abs/2007.11898, 2020. [Online]. Available: https://arxiv.org/abs/2007.11898

[7] G. Jocher, "Ultralytics yolov5," 2020. [Online]. Available: https://github.com/ultralytics/yolov5

[8] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *18th international conference on pattern recognition (ICPR'06)*, vol. 3. IEEE, 2006, pp. 850–855.

[9] B. Yang, X. Xu, and J. Li, "Keyframe-based camera relocalization method using landmark and keypoint matching," *IEEE Access*, vol. 7, pp. 86 854–86 862, 2019.

[10] S. Yang and S. Scherer, "Cubeslam: Monocular 3-d object slam," *IEEE Transactions on Robotics*, vol. 35, no. 4, pp. 925–938, 2019.

[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[12] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[13] H. M. S. Bruno and E. L. Colombini, "LIFT-SLAM: a deep-learning feature-based monocular visual SLAM method," *CoRR*, vol. abs/2104.00099, 2021. [Online]. Available: https://arxiv.org/abs/2104.00099

[14] E. Garcia-Fidalgo and A. Ortiz, "ibow-lcd: An appearance-based loop-closure detection approach using incremental bags of binary words," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3051–3057, 2018.

[15] M. Kuse and S. Shen, "Learning whole-image descriptors for real-time loop detection and kidnap recovery under large viewpoint difference," *Robotics and Autonomous Systems*, vol. 143, p. 103813, 2021.

[16] Sivic and Zisserman, "Video google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, 2003, pp. 1470–1477 vol.2.

[17] K. Yi, E. Trulls, V. Lepetit, and P. Fua, "Lift: Learned invariant feature transform," vol. 9910, 10 2016, pp. 467–483.

[18] Y.-b. Ai, T. Rui, X.-q. Yang, J.-l. He, L. Fu, J.-b. Li, and M. Lu, "Visual slam in dynamic environments based on object detection," *Defence Technology*, vol. 17, 09 2020.

[19] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2015.

[20] J. Li, D. Meger, and G. Dudek, "Semantic mapping for view-invariant relocalization," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7108–7115.

[21] J. C. V. Soares, M. Gattass, and M. A. Meggiolaro, "Visual slam in human populated environments: Exploring the trade-off between accuracy and speed of yolo and mask r-cnn," in *2019 19th International Conference on Advanced Robotics (ICAR)*, 2019, pp. 135–140.

[22] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017. [Online]. Available: http://arxiv.org/abs/1703.06870

[23] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[24] R. A. Wadud and W. Sun, "Dyob-slam: Dynamic object tracking slam system," *arXiv preprint arXiv:2211.01941*, 2022.

[25] D. S. Watkins, *Fundamentals of Matrix Computations*. USA: John Wiley & Sons, Inc., 1991.

[26] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union," June 2019.

[27] S. Urban, J. Leitloff, and S. Hinz, "Mlpnp - a real-time maximum likelihood solution to the perspective-n-point problem," *CoRR*, vol. abs/1607.08112, 2016. [Online]. Available: http://arxiv.org/abs/1607.08112

[28] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abm6074

[29] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[30] 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)*, Oct. 2012.

[31] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–308, 2009.