IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Online Dynamic SLAM with Incremental Smoothing and Mapping

Jesse Morris, Yiduo Wang and Viorela Ila

***Abstract*— Dynamic SLAM methods jointly estimate for the static and dynamic scene components. However, existing approaches, while accurate, are computationally expensive and unsuitable for online applications. In this work, we present a novel factor-graph formulation and system architecture for Dynamic SLAM that inherently supports incremental optimisation and online estimation. This represents the first formulation explicitly designed to leverage incremental inference methods in the dynamic setting. On multiple datasets, we demonstrate that our method achieves camera pose and object motion accuracy equal to or better than state-of-the-art. We further analyse the structural properties of our approach to demonstrate its scalability and provide insight regarding the challenges of solving Dynamic SLAM incrementally. Finally, we show that our formulation leads to problem structure well-suited to incremental solvers, and our system architecture further enhances performance, achieving a $5\times$ speed-up over existing methods. Code is open-sourced at https://github.com/ACFR-RPG/DynOSAM.**

I. Introduction

To build a complete mental map of complex dynamic environments, robotic systems must comprehensively understand *where* an object is and *how* it is moving. Dynamic objects are inherently unpredictable, and therefore *online* estimation of their motion is fundamental for safety critical operation. Despite recent progress in object and ego-motion estimation [1], [2], existing Dynamic Simultaneous Localisation and Mapping (Dynamic SLAM) frameworks remain computationally prohibitive for real-world use due to their reliance on intensive batch optimisation. Rather than relying on batch methods, online estimation can be achieved using incremental optimisation algorithms [3], [4] which exploit sparsity in the underlying optimisation problem to enable efficient incremental inference [5]. However, the Dynamic SLAM problem is inherently more complex than the static case; additional connections among object points, their motions and the observing camera poses result in a denser optimisation problem that is challenging to solve efficiently. Thus, formulating a factor graph for Dynamic SLAM which enhances problem sparsity, while maintaining accuracy, is vital for accurate online performance.

Motivated by these needs, we propose: i) a novel *Hybrid* formulation which produces a sparse problem formulation while enabling accurate estimation and, ii) an associated *Parallel-Hybrid* architecture that provides a practical and efficient back-end for Dynamic SLAM systems. The Hybrid formulation merges the benefits of the *object-centric* and *world-centric* representations [2], [6], [7]: retaining the accuracy of world-centric methods, which explicitly model the rigid-body motion of each object, while using the object-centric representation to reduce overall graph connectivity and minimise the number of state variables [7]. Our approach jointly estimates the camera pose, static structure, per-object motion and dynamic map, while also accumulating object structure over time and enabling direct recovery of each object's pose and velocity per frame.

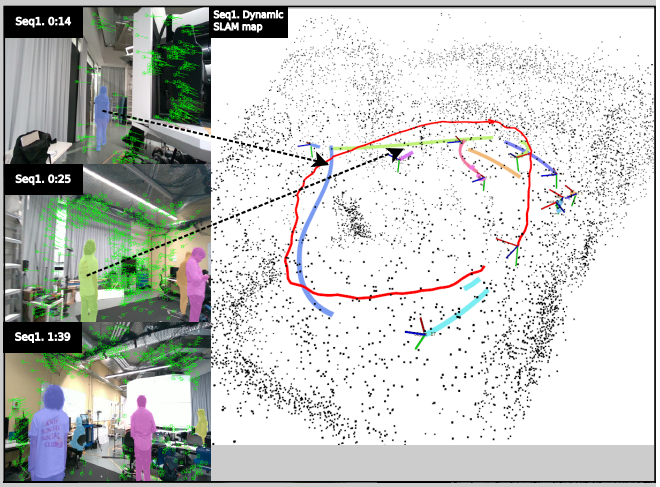

Fig. 1: Output of our proposed Dynamic SLAM system recorded in an indoor environment with multiple moving objects. Our method facilitates online estimation of the camera pose (red trajectory) and static scene (black point cloud) as well as the motion, structure and pose of dynamic objects (uniquely coloured) via incremental estimation using iSAM2 [4].

However, while the proposed Hybrid formulation enhances overall sparsity, the connectivity of the estimation problem scales with the number of visible objects, limiting long-term performance. To address this, we additionally propose the Parallel-Hybrid architecture as a practical solution to the Dynamic SLAM problem. This approach explicitly reduces the connectivity between camera and object variables which significantly improves efficiency and scalability, achieving at least a $5\times$ speed-up with minimal to no accuracy loss compared to a state-of-the-art baseline. Qualitative results on real-world sequences with multiple dynamic (non-rigid) objects additionally demonstrate robustness under realistic conditions, as shown in Fig. 1.

Thus, our contributions are:

- A novel *Hybrid* formulation for Dynamic SLAM that combines the benefits of the existing object and world-centric approaches.
- A novel *Parallel-Hybrid* architecture for Dynamic SLAM that facilitates online and incremental estimation. To the best of our knowledge, this is the first

Jesse Morris, Yiduo Wang and Viorela Ila are with the Australian Centre For Robotics (ACFR), School of Aerospace, Mechanical and Mechatronic Engineering (AMME), University of Sydney, 2006 Sydney, Australia. {jesse.morris,yiduo.wang,viorela.ila}@sydney.edu.au

work to apply incremental optimisation methods to the Dynamic SLAM problem.
- An analysis of the proposed architecture in the context of incremental inference, which highlights the practical trade-off's between accuracy and computation, as well as key challenges specific to Dynamic SLAM.
- The proposed incremental formulation significantly improves scalability and achieves a $5\times$ speed-up compared to existing methods, enhancing support for online estimation in dynamic environments.

II. Related Works

A. Graph-based Dynamic SLAM

Dynamic SLAM systems incorporate dynamic objects into the estimation problem. Typical approaches apply rigid-body motion models to tracked objects and jointly estimate object and camera poses [1], [2], [8], [9], [10]. This contrasts with SLAM methods that improve robustness by removing dynamic objects [11], [12], [13]. Our prior work [7] categorises Dynamic SLAM formulations as object or world-centric and demonstrates that this choice significantly impacts optimisation convergence and estimation accuracy. Therefore, reviewing these representations is essential to better understand ways to solve Dynamic SLAM incrementally.

Object-Centric Representations [1], [6], [14], [15], [16] represent dynamic points in the object's body frame where each point is static is under a rigid-body assumption. DynaSLAM II [6] outlines the benefits of this representation in reducing the number of state variables, but does not compare or analyse its computational benefits. However, the accuracy of object-centric methods is often limited due to the absence of explicit rigid-body constraints, as most approaches rely on a constant velocity model to constrain the optimisation and implicit rigidity through the object-centric representation [7]. Of these methods, MVO [1] attains higher accuracy by explicitly enforcing rigidity via point-pair distance constraints while maintaining physically plausibility motions via a constant velocity prior.

World-Centric Representations [2], [8], [17] represent object motions and points in a common global frame [18]. This parametrisation explicitly enforces rigid-body kinematics without needing additional constraints. Our recent work DynoSAM [2] uses this approach to achieve highly accurate object motion and pose estimation. However, both motions and points must be explicitly modelled at every time-step, resulting in a large factor-graph that is densely connected and computationally intensive to solve [7].

B. Incremental Dynamic SLAM

In this work we consider incremental SLAM methods to be those that perform state estimation using *incremental optimisation techniques*, e.g. [3], [4], [19], which enables efficient inference, minimising per-frame computation time while maintaining optimality. Many state-of-the-art (static) SLAM systems [20], [21], [22] employ iSAM2 [4] to facilitate online localisation and mapping, thereby better supporting robotic exploration. However, to the best of the authors' knowledge, no work has applied incremental optimisation to the Dynamic SLAM problem.

III. Preliminaries

A. Notation

Consider a dynamic scene comprised of camera poses $\mathcal{X}$ and object poses $\mathcal{L}$:

$$\mathcal{X} = \{{}^{W}\mathbf{X}_k \in \mathrm{SE}(3)\}_{k\in\mathcal{K}}, \quad \mathcal{L} = \{{}^{W}\mathbf{L}_k^j \in \mathrm{SE}(3)\}_{k\in\mathcal{K}_j}^{j\in\mathcal{J}_k},$$

where $\{W\}$ is the world frame, $\mathcal{K}$ is the set of all time-steps and $\mathcal{J}$ is the set of all object indices. $\mathcal{J}_k \subseteq \mathcal{J}$ are object indices observed at k, and $\mathcal{K}_j \subseteq \mathcal{K}$ is the subset of time-steps over which object j is observed.

Each pose ${}^{W}\mathbf{X}_k$ and ${}^{W}\mathbf{L}_k^j$ defines the orientation and position of the frames $\{X\}_k$ and $\{L\}_k^j$ with respect to $\{W\}$. $\{L\}_k^j$ defines the temporal instance of the moving object frame $\{L\}^j$ which is fixed to object j. At each k, the camera provides static $\mathcal{S}_k$ and dynamic $\mathcal{D}_k$ point measurements, e.g. from a RGBD camera. The homogenous coordinate of a point $\tilde{\mathbf{m}}^i \in \mathbb{R}^3$ is $\mathbf{m}^i = \left[\tilde{\mathbf{m}}^i, 1\right]^\top$, where i indicates correspondences across frames. A point measured in the sensor frame $\{X\}$ is denoted as $\mathbf{z}_{3\mathrm{D}} \in \mathbb{R}^3$ such that:

$$\left[\mathbf{z}_{3\mathrm{D}}, 1\right]^\top = {}^{X_k}\mathbf{m}_k^i = {}^{W}\mathbf{X}_k^{-1}\ {}^{W}\mathbf{m}_k^i.$$

Indices i and j are omitted when there is no ambiguity.

The motion of the frame $\{L\}^j$ between time-steps $e, k \in \mathcal{K}_j$ is defined with a *pose change* [18] and is denoted ${}^{W}_{e}\mathbf{H}_k \in \mathrm{SE}(3)$ such that:

$${}^{W}_{e}\mathbf{H}_k = {}^{W}\mathbf{L}_k\ {}^{W}\mathbf{L}_e^{-1}. \tag{1}$$

The three-indexed notation [18] is adopted to unambiguously show that ${}^{W}_{e}\mathbf{H}_k$ represents a motion *as observed from the static frame* $\{W\}$ and therefore represents the pose change of $\{L\}_e$ to $\{L\}_k$[1]. Vitally, this motion is numerically and conceptually distinct from both the object's pose and its relative motion. The relative motion, which is defined as:

$${}^{L_e}_{e}\mathbf{H}_k = {}^{W}\mathbf{L}_e^{-1}\ {}^{W}\mathbf{L}_k, \tag{2}$$

describes the motion of the object frame from e to k as observed by the frame $\{L\}_e$, while the frame's pose ${}^{W}\mathbf{L}_k$ describes the coordinates of the frame $\{L\}_k$ in $\{W\}$[2]. Consequently, a point represented locally on the object ${}^{L}\mathbf{m}$ may be transformed into the world frame using the pose:

$${}^{W}\mathbf{m}_k = {}^{W}\mathbf{L}_k\ {}^{L}\mathbf{m}, \tag{3}$$

while the object *motion* will transport any and all points on a rigid-body represented in $\{W\}$ from e to k:

$${}^{W}\mathbf{m}_k = {}^{W}_{e}\mathbf{H}_k\ {}^{W}\mathbf{m}_e. \tag{4}$$

[1]Expressing the $\mathrm{SE}(3)$ in an external reference frame makes it equivalent in function to the Pose Change Group ($\mathrm{PCG}(3)$) introduced in [18]. While $\mathrm{SE}(3)$ and $\mathrm{PCG}(3)$ differ in their group structure - $\mathrm{SE}(3)$ is a semi-direct product while $\mathrm{PCG}(3)$ is a direct product of rotation and translation - both may be used to represent the pose change of a frame.

[2]The matrix representation for a pose change and a pose may be equal in value when the observing frame coincidences with the moving frame, i.e. the left sub- and super-scripts are the same ${}^{W}\mathbf{L}_e = {}^{W}_{W}\mathbf{H}_e$, otherwise this transform represents a rigid-body pose change [18].

These equations highlight the distinction between ${}^{W}\mathbf{L}_k$ and ${}^{W}_{e}\mathbf{H}_k$ on how they apply to different point representations.

As a direct consequence of (4), the object's rigid-body constraints, i.e. the body does not deform, are explicitly captured by the motion ${}^{W}_{e}\mathbf{H}_k$ [2], [18]. Prior works [7], [8], [17] have shown that estimating for the *motion* of an object, rather than its pose, is beneficial in achieving accurate estimation. However, these works exclusively consider an object's consecutive (per-frame) motion ${}^{W}_{k-1}\mathbf{H}_k$, while we expand the utility of this representation to *non-consecutive* frames[3] ${}^{W}_{e}\mathbf{H}_k$ and demonstrate that this allows for a more efficient formulation of the Dynamic SLAM problem.

B. Incremental Inference and iSAM2 in Dynamic SLAM

ISAM2 [4] is a non-linear, graph-based optimisation algorithm that facilitates efficient online updates by using the Bayes Tree [23] to exploit the sparsity structure of the underlying system. A Bayes Tree $\mathcal{B}$ is built from a chordal Bayes Net via variable elimination on a factor graph $\mathcal{F}$, with each node representing a clique of conditionally dependent variables. Inference is performed via variable elimination of each clique in bottom-up order; this is equivalent to forming the Cholesky factor $\mathbf{R}$ for the linearised $\mathcal{F}$ [5]. The sparsity structure of $\mathbf{R}$ is directly captured in the topology of $\mathcal{B}$ where large cliques indicate that $\mathbf{R}$ is dense and solving the system is therefore highly inefficient. Thus, the memory and computation performance depends directly on the sparsity of $\mathbf{R}$, and designing $\mathcal{F}$ to preserve or enhance this sparsity is therefore critical for efficient, incremental estimation. Instead of performing inference over the entire Bayes Tree, iSAM2 achieves fast incremental updates by only recomputing parts of $\mathcal{B}$ affected by new measurements. By ordering recently affected variables near the root, computation is confined to the top of the tree.

The structure of Static SLAM typically results in small cliques [5] that can be solved and updated efficiently. In contrast, our results show that the structure of Dynamic SLAM differs fundamentally from the static case, as large cliques arise more naturally, leading to increased computational and memory demands during inference. Similar observations have also been reported in multi-robot SLAM [24], likewise exhibiting a problem structure distinct from static SLAM. To mitigate this, in the following section, we propose a formulation that preserves sparsity during incremental updates.

IV. Method

Our method incrementally tracks and estimates moving object motion's while jointly solving for scene structure and ego-motion. The full system is based on the *frontend* implemented in DynoSAM [2] to track 3D points and obtain initial estimates of the camera pose and object motion, which are then refined through the *incremental backend*.

[3]The frame-to-frame motion ${}^{W}_{k-1}\mathbf{H}_k$ represents the actual motion experienced by the object (only expressed in a different frame) which is not captured when the motion represents a pose change between non-consecutive time-steps, as in ${}^{W}_{e}\mathbf{H}_k$. We therefore adopt the term 'motion' only for consistency with prior work.

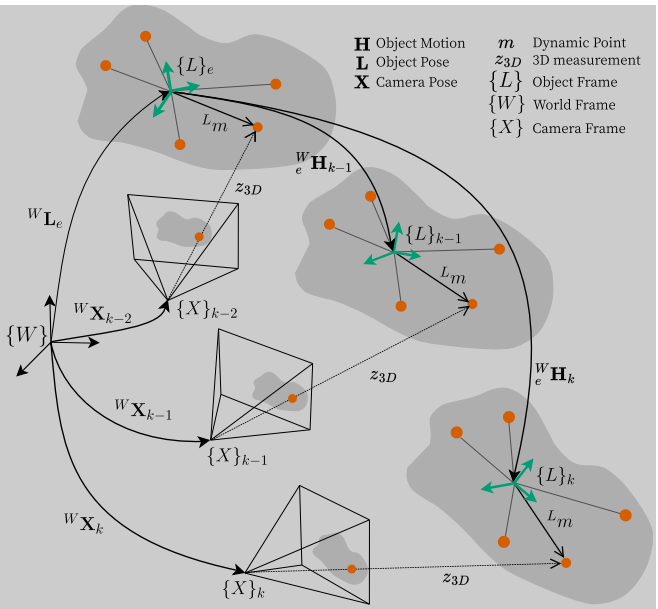

Fig. 2: The *Hybrid* Dynamic SLAM representation expresses object points locally and object motions as a rigid-body pose change expressed in $\{W\}$. A point ${}^{L}\mathbf{m}$ is first observed at time-step e and the fixed frame $\{L\}_e$ serves as a common reference frame for all future motions. For simplicity only one point is shown. Note: unlike a pose, a pose change (expressed in $\{W\}$) lacks a *direct* geometric interpretation [2]; thus the arrows associated with the pose change ${}^{W}_{e}\mathbf{H}_k$ are conceptual, indicating how all the points on the body move from e to k.

A. Hybrid Representation For Dynamic SLAM

We propose a *Hybrid* Dynamic SLAM representation, expressing object points ${}^{L}\mathbf{m}$ in a local reference frame $\{L\}$ attached to each object. Object motions ${}^{W}_{e}\mathbf{H}_k$ express the rigid-body pose change of that same frame, $\{L\}$, from e, the time-step the object was first observed, to the current time-step k. This gives us a consistent reference for all object points and enables direct estimation of rigid-body motion.

Our approach assigns a known pose ${}^{W}\mathbf{L}_e$ to the initial frame $\{L\}^{j}_{e}$ for each object. While any pose initialisation method might be acceptable [2], in our implementation we use the centroid of the first set of measurements with an identity rotation. This construction ensures that $\{L\}_e$ is placed on the object and is defined unambiguously with respect to the world frame. The pose of all subsequent frames $\{L\}_k$ may then be expressed purely in terms of the rigid-body motion ${}^{W}_{e}\mathbf{H}_k$ following (1):

$$ {}^{W}\mathbf{L}_k = {}^{W}_{e}\mathbf{H}_k \; {}^{W}\mathbf{L}_e \quad \forall k \in \mathcal{K}_j. \tag{5}$$

This enables direct computation of the body-frame motion using (2), from which the object's linear and angular velocities can be derived.

While defining dynamic points locally greatly reduces the number of variables required to define its structure, understanding the object map's global position is additionally important; its overall structure and pose with respect to the camera may then also be recovered. Using (4), each point can be expressed in $\{W\}$:

$$ {}^{W}\mathbf{m}_k = {}^{W}_{e}\mathbf{H}_k \; {}^{W}\mathbf{L}_e \; {}^{L}\mathbf{m} \quad \forall k \in \mathcal{K}_j. \tag{6}$$

Furthermore, as new object points are represented in the same

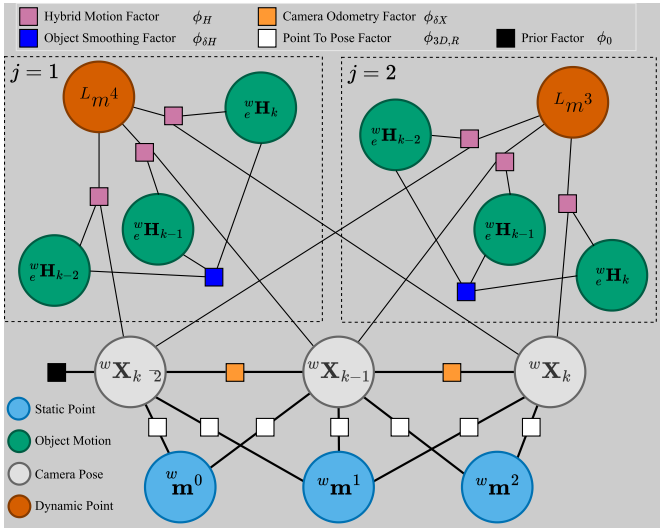

Fig. 3: Full Hybrid Dynamic SLAM factor-graph showing static points (blue) and camera poses (grey) at three consecutive frames. Two dynamic objects are shown, each with one point (orange), observed in all frames and connected by the hybrid motion factors (ϕ_H). The ternary object smoothing factors ($\phi_{\delta H}$) constrains the change in object motion (green).

object frame as existing ones, the object map naturally grows over time, yielding a more complete structure as additional fragments become visible. Consequently, for each object, applying (6) to all its points reconstructs the full object map, regardless of when individual points are first observed.

The novelty of our approach lies in creating a *fixed* coordinate system within which object points are locally defined. Via clearly defining the fixed pose ${}^{W}\mathbf{L}_e$ and its attached reference frame $\{L\}_e$, we create an invariant link between local object points and $\{W\}$. This enables the application of motion ${}^{W}_{e}\mathbf{H}_k$ to the points in $\{W\}$ rather than in $\{L\}$ following (4), thus ensuring the object's rigidity.

In contrast, other object-centric methods [1], [6] attach the object frame to the dynamic object pose ${}^{W}\mathbf{L}_k$, which acts on the local object points following (3). ${}^{W}\mathbf{L}_k$ is then constantly updated along with object points during optimisation. This approach does not inherently enforce rigid-body constraints [7], instead relying on the object-centric representation to implicitly enforce rigidity, unless additional constraints are added, e.g. [1]. By comparison, our approach requires no additional constraints, as the motion parametrisation inherently encodes them. Following the work of [7] this feature is considered essential for accurate estimation.

B. Factor Graph Construction

Using the Hybrid formulation, we pose our Dynamic SLAM estimation as a maximum-a-posteriori (MAP) problem using factors in the form of:

$$\phi(\cdot) \propto \exp\Big\{-\frac{1}{2}\|\mathbf{r}\|^2_{\Sigma}\Big\}, \tag{7}$$

where the residual function $\mathbf{r}$ has a corresponding covariance matrix Σ. This section outlines our novel residual functions associated with dynamic measurements and states. The complete factor-graph is shown in Fig. 3 where the static SLAM component uses the factors and method outlined in [2].

To enforce the rigid-body motion of each dynamic point we introduce the *hybrid motion factor*:

$$\mathbf{r}_H = \mathbf{z}_{3\mathrm{D}} - {}^{W}\mathbf{X}_k^{-1}\ {}^{W}_{e}\mathbf{H}_k\ {}^{W}\mathbf{L}_e\ {}^{L}\mathbf{m}, \tag{8}$$

where $\mathbf{z}_{3\mathrm{D}}$ is a measurement of the tracked point ${}^{L}\mathbf{m}$, ${}^{W}_{e}\mathbf{H}_k$ is the rigid-body motion of object j and ${}^{W}\mathbf{X}_k$ is the observing camera pose. New points seen at k are initialised by projecting the measurement into the object frame:

$${}^{L}\mathbf{m} = {}^{W}\mathbf{L}_e^{-1}\ {}^{W}_{k}\mathbf{H}_e\ {}^{W}\mathbf{X}_k\ \mathbf{z}_{3\mathrm{D}}. \tag{9}$$

In the case that $k > e$, ${}^{W}_{k}\mathbf{H}_e$ moves the point backwards from k to e. Note that this 'reversing' motion is not equivalent to inverting ${}^{W}_{e}\mathbf{H}_k$ [18], and is instead calculated as ${}^{W}_{k}\mathbf{H}_e = {}^{W}\mathbf{L}_k \left({}^{W}\mathbf{L}_e^{-1}\ {}^{W}_{e}\mathbf{H}_k\ {}^{W}\mathbf{L}_e\right)^{-1}\ {}^{W}\mathbf{L}_k^{-1}$. We incentivise plausible motions via a constant motion model:

$$\mathbf{I} = {}^{L_{k-2}}_{k-2}\mathbf{H}^{-1}_{k-1}\ {}^{L_{k-1}}_{k-1}\mathbf{H}_k. \tag{10}$$

The residual for this constraint is modelled in the object's body frame using the *object smoothing factor*:

$$\mathbf{r}_{\delta H} = \log\Bigg[\left(\left({}^{W}_{e}\mathbf{H}_{k-2}\ {}^{W}\mathbf{L}_e\right)^{-1}\left({}^{W}_{e}\mathbf{H}_{k-1}\ {}^{W}\mathbf{L}_e\right)\right)^{-1} \left(\left({}^{W}_{e}\mathbf{H}_{k-1}\ {}^{W}\mathbf{L}_e\right)^{-1}\left({}^{W}_{e}\mathbf{H}_{k}\ {}^{W}\mathbf{L}_e\right)\right)\Bigg]^{\vee} \tag{11}$$

Due to the lack of explicit object frame in purely world-centric approaches, the design of similar smoothing factors are restricted to residuals defined in $\{W\}$. This causes the error to scale with the object's distance from the origin and therefore bias the estimation [2]. Our residual error is intentionally expressed in the object-body frame to mitigate this issue; this design is uniquely facilitated by our formulation's use of non-consecutive object motions. Using the body-frame also allows class-specific motion models, such as those utilised in [1], [25], to be naturally expressed.

Our formulation's explicit object-centric design preserves and enhances sparsity. As shown in Fig. 3, each dynamic point is a leaf node, and therefore may be eliminated without introducing large cliques, making the proposed formulation well-suited to incremental solvers. In contrast, world-centric methods form chains of interconnected dynamic points, which leads to large cliques; this is validated by the Bayes Tree analysis in Section VI-C.1.

However, our results (Section VI) show that, when formulated as a joint estimation problem, computation time scales with the number of object variables. From a factor-graph perspective (Fig. 3), all object variables are linked through shared camera poses, which act as the sole separators between the static and dynamic components of the scene. This coupling introduces dependencies among variables and leads to the formation of large cliques containing variables from all recently observed objects.

V. Parallel Hybrid

To support online estimation and better preserve sparsity during incremental inference, we propose a modified formu-

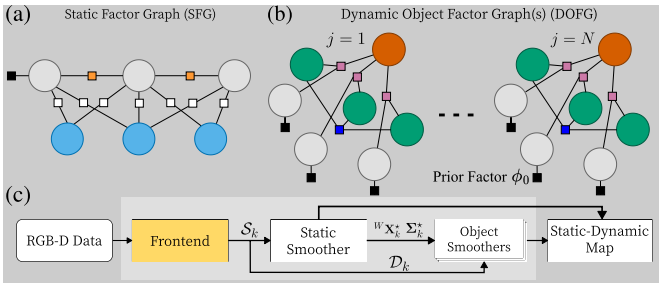

Fig. 4: Parallel-Hybrid architecture decouples the static **(a)** and dynamic **(b)** components from the Hybrid formulation (Fig. 3). **(c)** shows the system architecture where each smoother solves its corresponding factor-graph. Information is passed from the static factor-graph (SFG) to the dynamic object factor-graphs (DOFG) via the camera pose priors ϕ_0.

lation in which the factor-graph (Fig. 3) is partitioned into a static factor-graph (SFG) and $N = |\mathcal{J}|$ dynamic object factor-graphs (DOFGs), as illustrated in Fig. 4 (a–b). To fully decouple the estimation of each dynamic object from both the static scene and other objects, each DOFG is conditioned on the current camera pose estimate. Each DOFG may then be solved independently and in parallel; a structure we refer to as the *Parallel-Hybrid architecture*.

We propose the system architecture shown in Fig. 4 (c) to incrementally solve this decoupled structure where a separate instance of the iSAM2 algorithm is maintained for each factor-graph (one SFG and N DOFGs). The SFG is first updated with factors constructed from new static point measurements $\mathcal{S}_k$; solving yields the optimal camera pose ${}^W\mathbf{X}_k^\star$ and associated marginal covariance $\Sigma_{\mathbf{X}_k}^\star$. Each DOFG is then updated with factors constructed from $\{{}^W\mathbf{X}_k^\star, \Sigma_{\mathbf{X}_k}^\star, \mathcal{D}_k^j\}$. In each DOFG the camera pose is conditioned via a prior factor ϕ_0 constructed from the mean and covariance of the current pose, as highlighted in Fig. 4 (b).

Including the camera pose as a variable within each DOFG is essential to facilitate incremental updates, as updates to any camera pose will naturally propagate through the Bayes Tree and affect the estimates of object points and motions. Since the SFG and the DOFG's are maintained in separate Bayes Trees, we manually update the mean and covariance of each ϕ_0 when the corresponding pose ${}^W\mathbf{X}_k^\star$ undergoes significant relinearisation. Updates are detected using the fluid relinearisation strategy from iSAM2 [4], although this method could be refined. While the proposed architecture's unidirectional information flow enables efficient parallelization, this design choice restricts the propagation of dynamic object information back through the SFG, which in certain situations, e.g. when the object's motion models are correct, could otherwise improve camera pose estimation.

VI. Experiments

A. Setup

Due to the limited availability of open-source Dynamic SLAM systems, we adopt the world-centric formulation [7], [8], [17] as the state-of-the-art baseline. Implemented in DynoSAM [2], this formulation has been thoroughly compared against static and Dynamic SLAM algorithms, showing state-of-the-art performance in both camera pose and object motion/pose evaluations. For additional comparison with a purely static SLAM baseline, we modify the world-centric formulation to reject dynamic points and optimize only over static features. These results are reported as the Static Baseline. Evaluations are performed on KITTI Tracking [26], Outdoor Cluster [14] and OMD [27] datasets which contain relevant ground truth. TartanAir Shibuya [28] and VIODE [29] are used additionally for runtime analysis as these datasets contain highly dynamic sequences but lack object motion/pose ground truth.

Hybrid, Baseline, and Static Baseline formulations are evaluated using both batch (Levenberg–Marquardt) and incremental (iSAM2) solvers; the latter are prefixed with 'i-'. Parallel-Hybrid is always solved incrementally.

Code for both proposed methods is released and integrated with the open-source DynoSAM [2] framework and can therefore be run as part of a full Dynamic SLAM pipeline. As part of our release we include scripts used for all experiments to ensure full reproducibility of results. GTSAM [30] (version 4.2.0) is used for factor-graph optimisation. An Intel Core i9-9900 CPU with 32GB RAM and 2GB swap memory is used for all experiments.

B. Estimation Accuracy

Our method primarily aims to reduce the computational cost of incremental estimation. To verify that accuracy is maintained, we evaluate camera pose and object motion across all methods. To ensure fair comparison with the both baselines and isolate formulation accuracy, we report full-batch results for Hybrid, where, given its similarity to the Baseline we expect comparable accuracy especially in camera pose error. Incremental results (iHybrid and Parallel-Hybrid) demonstrate online performance relative to the batch solution. Parallel-Hybrid, which decouples the estimation, also enables direct comparison with jointly solved methods. These results reveal how solver type (incremental vs batch) and problem structure (joint vs decoupled) affect estimation accuracy. Loop closure is not used in any system.

Table I reports the error of the baseline's for each metric. The relative numerical error difference compared to the (Dynamic SLAM) Baseline is then reported for each proposed method. We report standard camera pose metrics: the RMSE of the absolute trajectory error (ATE) and relative pose error (RPE). For objects we report RMSE of motion error (ME) [2], averaged over all objects in the sequence. ME is a reliable way to report motion error as it is impartial to the object frame definition, which may differ among systems.

In RPE, errors differ by no more than $0.02\,\mathrm{m}$ (translation) and $0.01°$ (rotation), while in ATE the maximum deviation is just $0.19\,\mathrm{m}$ over trajectories spanning hundreds of meters. For object motion, the largest differences are $1.29°$ in rotation and $0.18\,\mathrm{m}$ in translation, although we do observe improvement in rotation error compared to the baseline.

These marginal differences demonstrate that our approach is as performant as the Baseline, which is known to achieve state-of-the-art accuracy in highly dynamic scenes. The (Dynamic SLAM) baseline achieves slightly better ATE than the

TABLE I: Accuracy of Object Motion and Camera Pose. Baseline, Static Baseline and Hybrid methods are solved batch. iHybrid and Parallel-Hybrid are solved incrementally. The metric error is reported for the two baseline's. We report the error relative to the (Dynamic SLAM) Baseline to show equal performance in green, improvement in dark green and worse performance in red. × indicates system failure as discussed in Section VI-C.1

	Metric	Method	KITTI									Outdoor Cluster				OMD
			00	01	02	03	04	05	06	18	20	L1	L2	S1	S2	S4U
Object Error	$ME_r(°)$	Baseline	1.11	1.04	0.97	0.26	1.24	0.85	0.39	0.57	0.52	0.82	0.70	0.69	2.36	0.67
		Hybrid	−0.23	−0.06	0.1	0.01	0.15	0.3	0.19	0.27	−0.3	−0.34	0.19	0.16	0.03	0.08
		iHybrid	0.12	0.08	−0.35	−0.03	0.71	−0.10	0.07	0.24	×	−0.46	−0.4	0.3	0.71	×
		Parallel-Hybrid	−0.06	−0.41	−1.29	−0.12	−0.23	−0.27	−0.09	−0.34	−0.14	−0.25	−0.18	−0.06	0.3	0.07
	ME_t(m)	Baseline	0.15	0.32	0.51	0.11	0.12	0.27	0.09	0.11	0.11	0.08	0.06	0.04	0.15	0.02
		Hybrid	−0.08	−0.05	0.0	0.0	0.0	−0.16	−0.02	0.03	0.0	−0.04	−0.04	0.0	0.11	0.0
		iHybrid	−0.08	−0.08	0.06	−0.01	−0.18	−0.16	−0.11	−0.01	×	−0.04	−0.05	0.0	0.09	×
		Parallel-Hybrid	0.01	−0.06	0.0	0.0	−0.04	−0.09	0.01	−0.04	0.0	0.02	−0.01	−0.02	−0.04	0.0
Camera Error	ATE(m)	Static Baseline	1.57	2.10	0.72	1.67	1.30	1.99	0.70	2.15	2.33	0.60	0.52	0.12	1.15	0.12
		Baseline	1.54	2.10	0.74	1.64	1.28	2.01	0.41	2.30	2.30	0.62	0.52	0.09	1.08	0.10
		Hybrid	0.0	0.0	0.0	−0.02	−0.01	0.0	0.0	0.0	0.0	0.0	0.0	−0.06	0.81	0.0
		iHybrid	−0.01	0.0	0.0	−0.19	−0.02	0.01	−0.01	−0.06	×	−0.02	−0.03	−0.05	0.46	×
		Parallel-Hybrid	−0.01	0.0	0.0	−0.03	−0.04	0.01	−0.02	−0.1	−0.12	0.16	−0.23	−0.02	−0.3	0.0
	$RPE_r(°)$	Static Baseline	0.05	0.03	0.02	0.04	0.08	0.04	0.05	0.02	0.05	0.02	0.03	0.02	0.03	0.71
		Baseline	0.05	0.03	0.02	0.07	0.07	0.07	0.05	0.04	0.03	0.02	0.02	0.01	0.02	0.66
		Hybrid	−0.01	0.0	0.0	0.1	0.0	−0.01	0.0	0.0	0.0	−0.01	−0.01	0.0	0.0	0.0
		iHybrid	0.0	−0.01	0.0	0.2	0.0	−0.01	0.0	0.0	×	0.0	0.0	0.0	0.0	×
		Parallel-Hybrid	0.0	0.0	0.0	0.01	0.02	0.0	0.01	0.0	0.0	0.0	0.01	0.01	0.0	0.0
	RPE_t(m)	Static Baseline	0.04	0.06	0.04	0.1	0.03	0.01	0.01	0.05	0.03	0.02	0.02	0.008	0.05	0.006
		Baseline	0.04	0.06	0.06	0.07	0.06	0.08	0.01	0.05	0.04	0.02	0.01	0.01	0.02	0.01
		Hybrid	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.01	0.0
		iHybrid	0.00	0.01	0.01	−0.01	−0.01	−0.02	0.00	0.00	×	0.00	0.00	0.00	0.00	×
		Parallel-Hybrid	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	−0.01	0.00

Static Baseline; however, the overall difference in camera pose error is marginal, indicating that jointly estimating dynamic objects provides only small improvements to camera pose accuracy on these datasets.

Table I additionally illustrates the impact of solver type and problem structure. Both incremental methods show higher ATE and ME than their batch counterparts—this is a well understood outcome as iSAM2 introduces approximations (e.g. partial state updates) that can deviate the estimation from the batch-optimal solution [4]. While the performance gap between iHybrid and Hybrid can be attributed to the incremental solver, Parallel-Hybrid performs consistently worse than all other methods that solve jointly, regardless of solver used. While prior Dynamic SLAM works [6], [28] have reported improvements in camera pose error due when solved jointly with dynamic objects, our results extend this by showing that the joint formulation also benefits object motion estimation. This strongly affirms that solving the Dynamic SLAM problem jointly leads to improved estimates in both camera pose and object motion. Nevertheless, Parallel-Hybrid delivers substantial efficiency gains (Section VI-C.2), highlighting a practical trade-off between accuracy and computational cost.

C. Incremental Solving

To charaterise the effect of incrementally solving each formulation we analyse the Bayes Tree's topology and assess overall computation time and scalability. Furthermore, since the topology is determined by elimination order, we generate results with iSAM2's `relinearizeSkip` (λ_{rs}) parameter set as 1 and 10. This parameter defines the number of updates skipped between relinearization checks, directly influencing the frequency of Bayes Tree reordering. Setting $\lambda_{\text{rs}} = 1$ enables more frequent relinearization and a more optimal ordering while $\lambda_{\text{rs}} = 10$ reduces recalculation frequency to emphasise performance-computation trade-off, as well as the effect of sub-optimal ordering.

1) Bayes Tree Analysis: Fig. 5 reports the average and maximum clique size, number of re-eliminated variables, and update time for the iHybrid and iBaseline methods. While setting $\lambda_{\text{rs}} = 1$ reduces overall clique size for both methods, and therefore improves efficiency, for either λ_{rs} iBaseline forms large cliques and variable elimination eventually exceeds system memory (marked in red), highlighting its dense problem structure. Furthermore, prior to failure, the steadily increasing number of re-eliminated variables indicates that growing portions of the Bayes Tree are recomputed each frame. In contrast, the Hybrid formulation forms smaller cliques with fewer re-eliminations, demonstrating its suitability for incremental solving.

Fig. 6 also reports Parallel-Hybrid results on `omd-s4u`, a unique sequence where all objects remain visible throughout, thereby increasing overall connectivity. The results reveal a clear trend in the variable number and average clique size of objects 1 and 3 compared to 2 and 4, corresponding to their physical trajectories: objects 1 and 3 undergo less rotation relative to the camera, producing longer tracklets as the same faces remain visible, which increases clique size and causes spikes in affected landmarks. This provides a key insight - in Dynamic SLAM, problem structure depends not only on ego-motion, like in Static SLAM, but also on the motion of objects relative to the camera. Continuous observation of the same dynamic landmarks resembles the special case in static

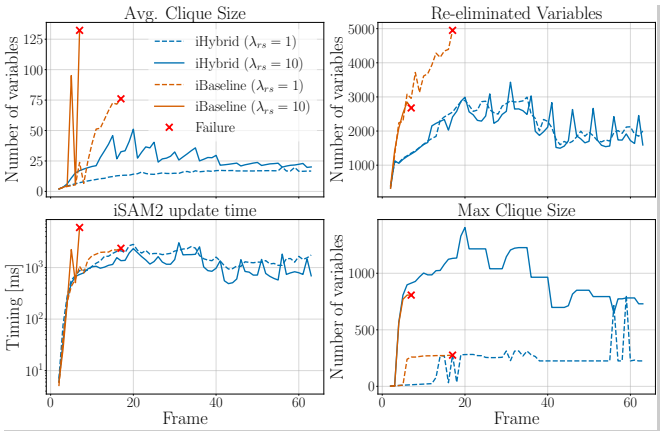

Fig. 5: Bayes Tree evaluation results on a portion of `KITTI 20`. Results are presented for λ_{rs} as 10 and 1 to demonstrate the effect of variable re-ordering and the natural scalability of the Hybrid compared to the Baseline. λ_{rs} limits relinearisation to every n steps, thereby delaying variable reordering and Bayes Tree restructuring.

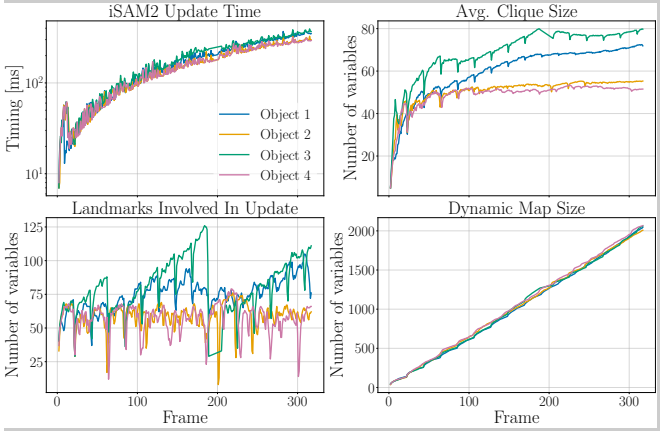

Fig. 6: Evaluation results of Parallel-Hybrid on `omd-s4u`: showing accumulated iSAM2 update time per object, average clique size, number of landmark variables involved in the update and total number of (landmark) variables in the per-object state estimate.

SLAM where the robot operates in a confined environment, in which the optimal solution is obtained by eliminating all landmark variables last via the Schur complement [3]. Since such scenarios are common in Dynamic SLAM, e.g., following a car along a highway, future work will explore similar techniques to improve efficiency.

2) Timing: Table II reports the average per-frame computation time of the iSAM2 update. With $\lambda_{\text{rs}} = 10$ iBaseline fails on almost all sequences excluding Outdoor Cluster, which contains fewer instances of multiple objects per frame, and TartanAir Shibuya VII, where the tracks on each object are noticeably short. In contrast, iHybrid completes four times as many sequences and is more efficient, demonstrating the benefit of its design. Although it still fails on long, object-dense sequences such as KITTI 18 and 20, its performance far exceeds iBaseline, which fails almost immediately once multiple objects appear (Fig. 7). This aligns with the trend in Fig. 5, where Baseline fails due to large clique formation. Reducing λ_{rs} to 1 improves scalability, and allows both methods to complete more sequences, though not all; however the average computation time is increased by over 200%.

TABLE II: iSAM2 update time for all incremental systems, reporting average per-frame time. For VIODE, Mid and High labels are sequence difficulty, associated with the number of dynamic objects. × means failure.

		Timing (ms)		
		Parallel-Hybrid	iHybrid	iBaseline
Dataset	Seq.		$\lambda_{\text{rs}} = 10/1$	
KITTI	00	**247**	601/4771	×/2632
	01	**232**	611/1683	×/2340
	02	**250**	648/2146	×/×
	03	**333**	665/2202	×/1502
	04	**133**	733/2287	×/3252
	05	**332**	550/2931	×/3130
	06	**420**	1013/1527	×/×
	18	**397**	×/8359	×/7166
	20	**1252**	×/×	×/×
OMD	S4U	**494**	×/×	×/×
Outdoor Cluster	L1	**453**	1268/2105	×/3100
	L2	**486**	930/1823	1119/2158
	S1	**124**	322/862	330/868
	S2	**78**	244/589	361/518
VIODE CD=City Day CN=City Night PL=Parking Lot M=Mid H=High	CD_M	**1332**	×/15822	×/×
	CD_H	**881**	2536/16262	×/×
	CN_M	**1164**	×/×	×/×
	CN_H	**1046**	2881/9325	×/×
	PL_M	**791**	×/3648	×/×
	PL_H	**838**	1947/4085	×/×
TartanAir Shibuya	IV	**426**	×/3286	×/×
	V	**276**	760/1673	×/1031
	VI	**149**	360/663	×/835
	VII	**75**	142/476	146/342

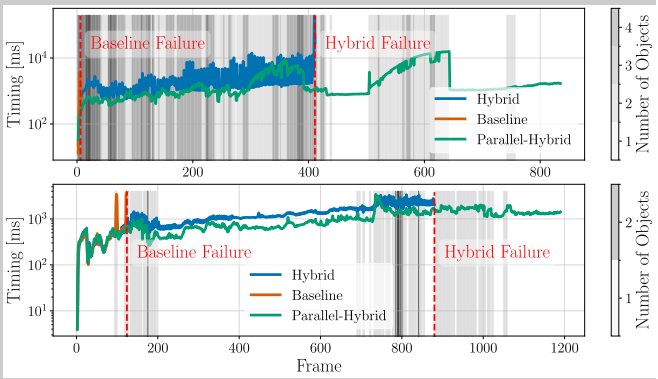

Fig. 7: Per frame iSAM2 update time on KITTI 20 (top) VIODE Parking Lot (bottom). Failure points are highlighted (red) and grey lines indicate the number of objects tracked at each frame.

While the Hybrid structure results in significant improvements over the Baseline, the Parallel-Hybrid method is 2× more efficient compared to iHybrid, 5× faster than iBaseline, completes all sequences and achieves an optimisation frequency of between 1 Hz to 5 Hz. This highlights that the object-camera connections are the key factors limiting computational efficiency. Overall, the Hybrid formulation is crucial for maintaining small cliques, while the Parallel-Hybrid architecture enables long-term, efficient operation.

3) Real World Sequences: To demonstrate real-world applicability, we deployed our system on indoor sequences featuring multiple non-rigid (human) dynamic objects, captured using an Intel RealSense D415, as shown in Fig. 1. With minimal tuning, e.g. limiting the feature tracks per object, our method achieves online performance, as shown

TABLE III: Average timing performance of full pipeline (Parallel-Hybrid back-end) on the real-world sequence shown in Fig. 1.

Module	Frontend (per frame)	DOFG update	SFG update
Timing (ms)	45	112	70

in Table III. Our approach successfully estimates camera and object trajectories even with non-rigid motion, and scales to crowded scenes with 9 dynamic objects, as shown in *Seq1*.

VII. Future Work & Conclusion

The Parallel-Hybrid's decoupled structure prevents informative dynamic observations flowing back from the DOFG's to the SFG. Future work will enable bidirectional information exchange transfer between the dynamic and static components which is relevant when object kinematics are well-known. Furthermore, our results indicate that solving jointly is beneficial to overall estimation accuracy. Therefore, we will continue to explore incremental methods and problem formulations that facilitate online joint estimation.

We propose a novel approach to the Dynamic SLAM problem that enables online estimation by significantly enhancing problem sparsity to facilitate incremental inference. Our results provide insights into the structure of the Dynamic SLAM problem and reveals that decoupling the dynamic object estimation from the static scene can degrade accuracy. Our proposed Hybrid formulation combines world and object-centric representations, which results in a problem structure well-suited for incremental inference. The Parallel-Hybrid is a practical solution to the Dynamic SLAM problem which enables parallel estimation and massively improves the overall efficiency and scalability of online performance.

VIII. Acknowledgments

This work was supported by the ARIAM Research Hub under Australian Research Council Grant IH210100030.

References

[1] K. M. Judd and J. D. Gammell, "Multimotion Visual Odometry (MVO)," *Intl. J. of Robotics Research*, 2024.
[2] J. Morris, Y. Wang, M. Kliniewski, and V. Ila, "Dynosam: Open-source smoothing and mapping framework for dynamic slam," *IEEE Trans. Robotics*, 2025.
[3] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.
[4] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *Intl. J. of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.
[5] F. Dellaert, M. Kaess, *et al.*, "Factor graphs for robot perception," *Foundations and Trends® in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017.
[6] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "Dynaslam ii: Tightly-coupled multi-object tracking and slam," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5191–5198, 2021.
[7] J. Morris, Y. Wang, and V. Ila, "The importance of coordinate frames in dynamic slam," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2024.
[8] J. Zhang, M. Henein, R. Mahony, and V. Ila, "VDO-SLAM: A Visual Dynamic Object-aware SLAM System," *arXiv preprint arXiv:2005.11052*, 2020.
[9] Y. Liu, C. Guo, Y. Luo, and Y. Wang, "Dynameshslam: A mesh-based dynamic visual slammot method," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5791–5798, 2024.
[10] R. Tian, Y. Zhang, L. Yang, J. Zhang, S. Coleman, and D. Kerr, "Dynaquadric: Dynamic quadric slam for quadric initialization, mapping, and tracking," *IEEE Trans. on Intelligent Transportation Systems*, vol. 25, no. 11, pp. 17 234–17 246, 2024.
[11] 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.
[12] C. Campos, R. Elvira, J. J. Gómez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *IEEE Trans. Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
[13] S. Song, H. Lim, A. J. Lee, and H. Myung, "Dynavins++: Robust visual-inertial state estimator in dynamic environments by adaptive truncated least squares and stable state recovery," *IEEE Robotics and Automation Letters*, 2024.
[14] J. Huang, S. Yang, Z. Zhao, Y. Lai, and S. Hu, "Clusterslam: A slam backend for simultaneous rigid body clustering and motion estimation," in *Proc. of the Intl. Conf. on Computer Vision (ICCV)*, 2019, pp. 5874–5883.
[15] J. Huang, S. Yang, T.-J. Mu, and S.-M. Hu, "Clustervo: Clustering moving instances and estimating visual odometry for self and surroundings," in *Proc. of the IEEE/CVF Intl. Conf. Computer Vision and Pattern Recognition*, 2020, pp. 2168–2177.
[16] I. Ballester, A. Fontán, J. Civera, K. H. Strobl, and R. Triebel, "Dot: Dynamic object tracking for visual slam," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2021, pp. 11 705–11 711.
[17] M. Henein, J. Zhang, R. Mahony, and V. Ila, "Dynamic SLAM: The Need for Speed," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020, pp. 2123–2129.
[18] G. S. Chirikjian, R. Mahony, S. Ruan, and J. Trumpf, "Pose changes from a different point of view," in *Proc. of the ASME Intl. Design Engineering Technical Conf. (IDETC)*. ASME, 2017.
[19] V. Ila, L. Polok, M. Šolony, and P. Svoboda, "SLAM++-A highly efficient and temporally scalable incremental SLAM framework," *Intl. J. of Robotics Research*, vol. 36, no. 2, pp. 210–230, 2017.
[20] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.
[21] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robotics*, vol. 33, no. 2, pp. 249–265, 2017.
[22] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020, pp. 5135–5142.
[23] M. Kaess, V. Ila, R. Roberts, and F. Dellaert, "The bayes tree: An algorithmic foundation for probabilistic robot mapping," in *Algorithmic Foundations of Robotics IX: Selected Contributions of the Ninth International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2010, pp. 157–173.
[24] Y. Zhang, M. Hsiao, J. Dong, J. Engel, and F. Dellaert, "Mr-isam2: Incremental smoothing and mapping with multi-root bayes tree for multi-robot slam," in *IROS*. IEEE, 2021, pp. 8671–8678.
[25] M. Gonzalez, E. Marchand, A. Kacete, and J. Royan, "Twistslam: Constrained slam in dynamic environment," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6846–6853, 2022.
[26] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Intl. J. of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.
[27] K. M. Judd and J. D. Gammell, "The Oxford Multimotion Dataset: Multiple SE(3) Motions with Ground Truth," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 800–807, 2019.
[28] Y. Qiu, C. Wang, W. Wang, M. Henein, and S. Scherer, "Airdos: Dynamic slam benefits from articulated objects," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2022, pp. 8047–8053.
[29] K. Minoda, F. Schilling, V. Wüest, D. Floreano, and T. Yairi, "Viode: A simulated dataset to address the challenges of visual-inertial odometry in dynamic environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1343–1350, 2021.
[30] F. Dellaert and GTSAM Contributors, "borglab/gtsam," May 2022. [Online]. Available: https://github.com/borglab/gtsam