DynaNav-SLAM: Efficient continuous dynamic object state estimation aided by pixel flow

Ehsan Hashemi[1], *Member, IEEE*, Xingkang He[2], Karl H. Johansson[3], *Fellow, IEEE*

***Abstract*—A novel robust and slip-aware .**

I. Introduction

Add contribution section

II. Background

A. Lie Group overview

This section discusses the notation used throughout this work to represent the state estimation of the ego-vehicle and objects. Let define $\mathbf{T}_{wc}^{k}$,$\mathbf{O}_{wo}^{k,l} \in SE(3)$ as the ego-vehicle and j^{th} object pose. The pose matrix representation has the blockwise form of:

$$\mathbf{T}_{wc} = \begin{bmatrix} \mathbf{R}_{wc} & \mathbf{t}_{wc} \\ \mathbf{0} & 1 \end{bmatrix} \tag{1}$$

where the rotation $\mathbf{R}_{wc} \in SO(3)$ and translation $\mathbf{t}_{wc}$ are expressed in the camera frame $\{c\}$ w.r.t the global inertial frame $\{w\}$. Likewise, the same coordinate system rule applies to the moving objects. The $SE(3)$ group is considered as a Lie Group and it has a tangent space, known as Lie algebra or $\mathfrak{se}(3)$, which is defined either in the local frame at some pose $\mathbf{T}$ or at the identity $\mathbf{I}$ of the group. Typically, motion updates yielded from optimization are applied in the form of Lie algebra, as it represents an approximation of the Euclidean space in the Lie manifold. The wedge operator $(\cdot)^{\wedge}$ maps the vector $\boldsymbol{\xi} \in \mathbb{R}^6$ to $\mathfrak{se}(3)$.

$$\boldsymbol{\xi}^{\wedge} = \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\phi} \end{bmatrix}^{\wedge} = \begin{bmatrix} \boldsymbol{\phi}^{\wedge} & \boldsymbol{\rho} \\ \mathbf{0}^{\top} & 0 \end{bmatrix} \tag{2}$$

where the $\boldsymbol{\omega}^{\wedge}$ is a symmetric matrix:

$$\boldsymbol{\omega}^{\wedge} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{3}$$

Elements of $\mathfrak{se}(3)$ tangent space are lifted to the $SE(3)$ group by the group exponential mapping $\exp(\cdot)$ such that $\mathbf{T} = \exp(\boldsymbol{\xi}^{\wedge}) \in SE(3)$. Conversely, there exists the vee operator $(\cdot)^{\vee}$ that maps from $\mathfrak{se}(3)$ to $\mathbb{R}^6$ and the $\log(\cdot)$ mapping $\boldsymbol{\xi} = \log(\mathbf{T})^{\vee}$ to go back from the Lie group to its tangent space. Closed forms for the $\exp$ and $\log$ mappings in $SE(3)$ group are found in [1]. Besides, the capitalized $\mathrm{Exp}(\cdot)$ and $\mathrm{Log}(\cdot)$ mappings are convenient bypasses from Euclidean space directly to the Lie Group such that:

$$\begin{aligned} \mathrm{Exp}: \quad & \mathbb{R}^6 \to SE(3) \quad ; \quad \boldsymbol{\xi} \mapsto \mathbf{T} = \mathrm{Exp}(\boldsymbol{\xi}) \\ \mathrm{Log}: \quad & SE(3) \to \mathbb{R}^6 \quad ; \quad \mathbf{T} \mapsto \boldsymbol{\xi} = \mathrm{Log}(\mathbf{T}) . \end{aligned}$$

where

$$\mathbf{T} = \mathrm{Exp}(\boldsymbol{\xi}) = \exp(\boldsymbol{\xi}^{\wedge}) \tag{4}$$

$$\boldsymbol{\xi} = \mathrm{Log}(\mathbf{T}) = \log(\mathbf{T})^{\vee} \tag{5}$$

A quiet useful property is the first order approximation of Exp map:

$$\mathrm{Exp}(\boldsymbol{\xi}) \approx \mathbf{I} + \boldsymbol{\xi}^{\wedge} \tag{6}$$

1) Adjoint: As mentioned beforehand, motion updates in the form of tangent space $\boldsymbol{\xi}^{\wedge}$ are parametrized w.r.t to the group identity or locally. Each one of them is associated with the left $\mathbf{T}_{wc} \leftarrow \mathrm{Exp}(\boldsymbol{\xi}_c)\mathbf{T}_{wc}$ and right increments $\mathbf{T}_{wc} \leftarrow \mathbf{T}_{wc}\,\mathrm{Exp}(\boldsymbol{\xi}_w)$ respectively (Verify the right-left increment). These two updates are related through the adjoint map $\mathrm{Adj}_{\mathbf{T}}(\cdot)$:

$$\boldsymbol{\xi}_w^{\wedge} = \mathrm{Adj}_{\mathbf{T}_{wc}}(\boldsymbol{\xi}_c^{\wedge}) \triangleq \mathbf{T}_{wc}\boldsymbol{\xi}_c^{\wedge}\mathbf{T}_{wc}^{-1} \tag{7}$$

The adjoint also has its own shortcut in a matrix form $\mathbf{Adj}_{(\cdot)} \in \mathbb{R}^{6\times 6}$ (refer to [1] for its closed form):

$$\boldsymbol{\xi}_w = \mathbf{Adj}_{\mathbf{T}_{wc}}\boldsymbol{\xi}_c \tag{8}$$

It is straightforward to prove that $\mathbf{Adj_{AB}} = \mathbf{Adj_A}\mathbf{Adj_B}$. In case that we apply the adjoint map directly over a $\mathfrak{se}(3)$ element, the introduction of a new mapping is required. It is known as the Lie Algebra adjoint and it has the two forms: as operator $\mathrm{adj}_{\boldsymbol{\xi}}(\cdot)$ and as matrix $\mathbf{adj}(\cdot) \in \mathbb{R}^{6\times 6}$:

$$\boldsymbol{\xi}_w^{\wedge} = \mathrm{adj}_{\log(\mathbf{T}_{wc})}\boldsymbol{\xi}_c^{\wedge} \triangleq \log(\mathbf{T}_{wc})\boldsymbol{\xi}_c^{\wedge} - \boldsymbol{\xi}_c^{\wedge}\log(\mathbf{T}_{wc}) \tag{9}$$

$$\boldsymbol{\xi}_w = \mathbf{adj}_{\log(\mathbf{T}_{wc})}\boldsymbol{\xi}_c \tag{10}$$

An interesting property that connects the Lie group adjoint matrix $\mathbf{Adj}(\cdot)$ and the Lie algebra adjoint matrix $\mathbf{adj}(\cdot)$ is the following Taylor expansion:

$$\mathbf{Adj}_{\mathbf{T}} = \mathbf{I} + \mathbf{adj}_{\boldsymbol{\xi}} + \frac{1}{2!}\mathbf{adj}_{\boldsymbol{\xi}}^2 + \text{H.O.T} \tag{11}$$

In later sections, we will exploit this property for specific Jacobian derivations.

2) Jacobian: The differentiation over the Lie Group is the core of several state estimation algorithms and needs a special treatment as a derivative over a manifold. Let introduce the right Jacobian (refer to [1] for its closed form) as:

$$\mathbf{J}_r(\boldsymbol{\xi}) = \left.\frac{\partial\,\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\xi})^{-1}\,\mathrm{Exp}(\boldsymbol{\xi}+\delta\boldsymbol{\xi}))}{\partial\delta\boldsymbol{\xi}}\right|_{\delta\boldsymbol{\xi}=\mathbf{0}} \tag{12}$$

The Jacobian quantifies the variation of the evaluated $\boldsymbol{\xi}$ due to an update $\delta\boldsymbol{\xi}$ expressed at the Lie group identity and

[1] Ehsan Hashemi (*Corresponding author*) is with the Department of Mechanical Engineering, University of Alberta, Edmonton, AB, Canada. This work is supported by the Natural Science and Engineering Research Council of Canada, Discovery Grants YYYYY. ehashemi@ualberta.ca

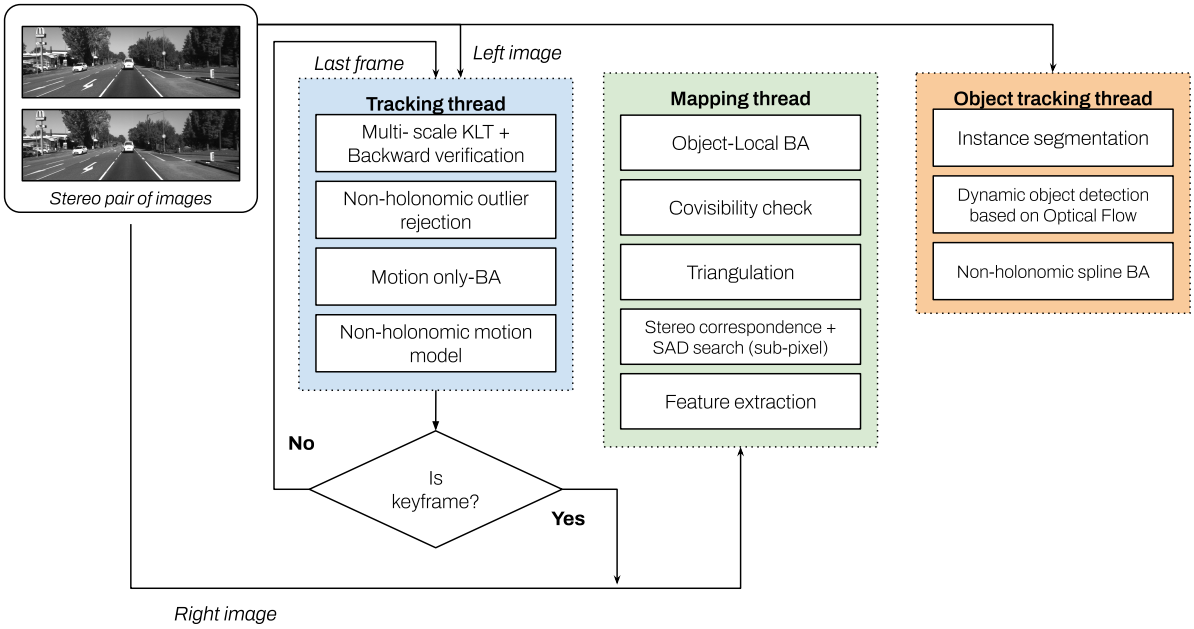

Fig. 1: **System schematic**. Our system receives a stereo pair of images as input and processes them in a tracking thread where 2D features $\mathbf{z}$ are detected, associated between consecutive timestamps t_{k-1} to t_k, and estimates an initial vehicle pose $\mathbf{T}_{wc}^{k}$. The object tracking thread segments the moving instances and detects their feature points to later track them and provide common landmarks for pose inference. Finally, the mapping thread is triggered by an incoming keyframe and performs the covisibility check, triangulation of stereo features for map initialization, and object-level Bundle Adjustment (BA).

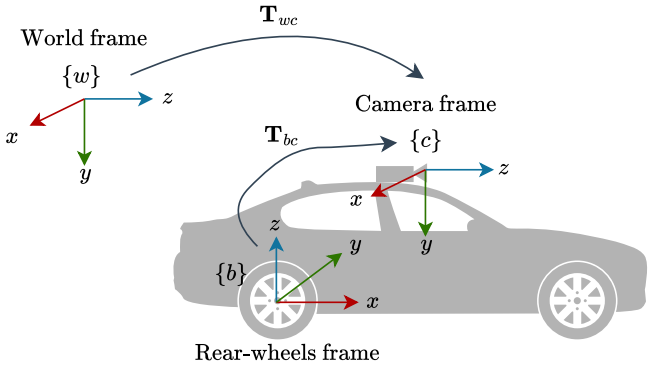

Fig. 2: Coordinate frame systems including world frame $\{w\}$(aligned the first camera pose), camera frame $\{c\}$, and rear wheels frame $\{b\}$. The last two are related by the extrinsics matrix $\mathbf{T}_{bc}$. Add w_b frame

projected to $\mathbb{R}^6$. Two key properties related to the Jacobians for $\boldsymbol{\xi} \to \mathbf{0}$:

$$\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\xi})\,\mathrm{Exp}(\delta\boldsymbol{\xi})) = \boldsymbol{\xi} + \mathbf{J}_r(\boldsymbol{\xi})^{-1}\delta\boldsymbol{\xi} + \mathcal{O}(\boldsymbol{\xi}^2) \tag{13}$$

$$\mathrm{Exp}(\boldsymbol{\xi} + \delta\boldsymbol{\xi}) = \mathrm{Exp}(\boldsymbol{\xi})\,\mathrm{Exp}(\mathbf{J}_r(\boldsymbol{\xi})\delta\boldsymbol{\xi}) + \mathcal{O}(\delta\boldsymbol{\xi}^2) \tag{14}$$

B. Rigid body kinematics

The generalized velocity $\boldsymbol{\varpi}_c^{\wedge} = [\mathbf{v}_c, \boldsymbol{\omega}_c] \in \mathfrak{se}(3)$ comprises the linear velocity $\mathbf{v}_c$ and angular velocity $\boldsymbol{\omega}_c$ of $\{c\}$ w.r.t the instantaneous frame that aligns with $\{c\}$ for a rigid body pose $\mathbf{T} \in SE(3)$ and it follows the equation:

$$\dot{\mathbf{T}}_{wc} = \mathbf{T}_{wc}\boldsymbol{\varpi}_c^{\wedge} \tag{15}$$

It has been defined using the right increment related to the local reference of $\mathbf{T}$. This quantity is uniquely defined for each point contained within the rigid body, thus being frame-dependent. That is inconvenient if we want to infer an object's motion from the 3D points that lie on its surface since their local motions are not compatible, as discussed in [2]. The quickest solution would be to define the velocity in the world frame instead (at the Lie group identity).

$$\dot{\mathbf{T}}_{wc} = \boldsymbol{\varpi}_w^{\wedge}\mathbf{T}_{wc} \tag{16}$$

Nonetheless, most continuous motion solvers utilize local updates [3], [4]. Rewriting their derivations, especially those regarding B-spline basis composition, could be cumbersome. Furthermore, faster computations could be exploited by inter-changing between the two formulations. We can related Eq. 15 and 16 to get a direct mapping for the velocities:

$$\boldsymbol{\varpi}_w^{\wedge} = \mathbf{T}_{wc}\boldsymbol{\varpi}_c^{\wedge}\mathbf{T}_{wc}^{-1} \tag{17}$$

and recalling Eq.7, we utilize the $SE(3)$ adjoint operator.

$$\boldsymbol{\varpi}_w^{\wedge} = \mathrm{Adj}_{\mathbf{T}_{wc}}\boldsymbol{\varpi}_c^{\wedge} \tag{18}$$

The conclusion of this explanation regarding kinematics comes to the hand of motion interpolation, where we can integrate the velocity in discrete form to update the object pose with respect to the group identity $\mathbf{T} \leftarrow \exp(\boldsymbol{\varpi}_w^{\wedge}\Delta t)\,\mathbf{T}$. We will infer the continuous velocity from the B-Spline and embed it into the discrete factors, some of which were originally formulated for local motion. Therefore, we will

need to transform them using the Adj mapping or plug their components directly when the error factors are continuous.

C. Map representation

Let $\mathbf{p}_w^{i,k} = [x, y, z]^\top \in \mathbb{R}^3$ be the i^{th} 3D point seen in the frame $\{w\}$ at t_k. This is a general notation for any landmark. For static landmarks, we omit the time index as it is constant. We transform points to camera frame $\{c\}$ as $\mathbf{p}_c^{i,k} = (\mathbf{T}_{wc}^k)^{-1} \odot \mathbf{p}_w^{i,k}$ where $\odot$ is the group action of $SE(3)$ into $\mathbb{R}^3$ (equivalent to matrix multiplication for homogeneous points). The propagation of global motion also applies to landmarks since they are fixed to the same rigid body.

$$\mathbf{p}_w^{i,k} = {}_{k-1}^{w}\mathbf{H}_k \odot \mathbf{p}_w^{i,k-1} \tag{19}$$

where ${}_{k-1}^{w}\mathbf{H}_k$ has been defined as the camera motion from t_{k-1} to t_k expressed in frame $\{w\}$. Notice the similarities with Eq.15, though one is described in continuous format and the other in discrete time.

The points are measured as image pixel features $\mathbf{z}^{i,k} = [u, v]^\top \in \mathbb{R}^2$ by the pinhole camera model $\pi(\cdot)$:

$$\hat{\mathbf{z}}^{i,k} \triangleq \pi(\mathbf{p}_c^{i,k}) = \begin{bmatrix} \frac{x f_x}{z} + c_x \\ \frac{y f_y}{z} + c_y \end{bmatrix} \tag{20}$$

in which (f_x, f_y, c_x, c_y) are denoted as the camera intrinsic parameters. The inverse projection $\pi(\cdot)^{-1}$ retrieves back the landmarks if the corresponding depth measurement d is available:

$$\mathbf{p}_c^{i,k} \triangleq \pi^{-1}(\mathbf{z}^{i,k}, d) = \begin{bmatrix} \frac{(u-c_x)d}{f_x} \\ \frac{(v-c_y)d}{f_y} \\ d \end{bmatrix} \tag{21}$$

D. B-Splines

B-Splines are one of the plenty forms to represent parametrized continuous trajectories in state estimation. They are represented by the weighted linear combination of the control points (knots) $\mathbf{p}_i$ and the k^{th} degree B-Spline basis functions $N_{i,k}$ as $\mathbf{p}(t) = \sum_{i=0}^{n} B_{i,k}(t)\mathbf{p}_i$. The basis coefficients are computed from the De Boor–Cox recursive form [5]. Generally, the basis is fixed while the control points are adjusted iteratively. A k^{th} B-Splines basis guarantees smoothness C^{k-1}, a desirable property for estimating continuous velocity and acceleration. Locality is also a B-Spline property, and it allows incremental updates for efficient computation [6] as:

$$\mathbf{p}(u) = \mathbf{p}_i + \sum_{i=1}^{k-1} \lambda_j(u)\mathbf{d}_j^i \tag{22}$$

where $\lambda(u) = \mathbf{M}^k\mathbf{u}$, $\mathbf{M}^k$ is the B-spline blending matrix of k^{th} order, $\mathbf{u} = [1, u, u^2, \ldots, u^{k-1}]$ from the normalized time $u = \frac{t-t_i}{t_{i+1}-ti} \in [0, 1]$ between the intervals $[t_i, t_{i+1}]$, and the control points $\mathbf{d}_j^i = \mathbf{p}_{i+j} - \mathbf{p}_{i+j-1}$. In [7], it was shown that Eq.22 could be represented in terms of Lie group elements as:

$$\mathbf{T}(u) = \mathbf{T}_i \prod_{j=1}^{k-1} \mathrm{Exp}(\lambda_j(u)\mathbf{d}_j^i) \tag{23}$$

with the generalized $SE(3)$ displacement vector $\mathbf{d}_j^i$:

$$\mathbf{d}_j^i = \mathrm{Log}(\mathbf{T}_{i+j-1}^{-1}\mathbf{T}_{i+j}) \in \mathbb{R}^6 \tag{24}$$

The displacement between two poses is mapped from the Lie Group to $\mathbb{R}^6$, scaled by $\lambda_j(u)$ and projected back to $SE(3)$ by $\mathrm{Exp}(\cdot)$ to be accumulated and yield the parametrized curve in the manifold. It is essential to note that the displacement vector is defined as either right or left increments and this is where most spline-based SLAM algorithms differ.

E. Non-holonomic Ackermann Steering

Ground-vehicles with Ackermann steering cannot have arbitrary poses as the configuration of wheels prohibits longitudinal slip, and forces to always have a rotational component. As consequence, that phenomena generates an Instantaneous Center of Rotation (ICR) [8]. Most urban vehicles are modeled with those constraints into consideration. Assuming no available information of the steering angle ϕ, the non-linear planar mobile robot kinematics in the state $\boldsymbol{x} = [x, y, \theta]^\top$ at the rear wheels middle axis with the input vector $\boldsymbol{u} = [v, \omega]^\top$ are:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{25}$$

where v is the linear velocity command and ω is the rotational speed. Extracting the first two rows of the model and equating them to solve v:

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0 \tag{26}$$

Since the homogeneous equation cannot be reduced through integration, we denote it as a *non-holonomic constraint*. This constraint can be exploited in state estimators to enhance the robustness as a motion prior. Frequently, Eq. 25 was approximated in discrete time which makes the motion model less valid depending on the magnitude of slip and velocity. For these reason, our proposal infers vehicles motion in a continuous path.

III. Methodology

A. Vision Front-end

The front end is inspired by the light-weight and accurate sensor processing thread from OV^{2}SLAM [9]. During the initialization, the thread detects 500 Shi-Tomasi corners as GFTT features [10] in the left image with a minimum pixel distance of 15 px. The module tracks the features across consecutive images with a coarse-to-fine strategy using the Lucas Kannade (LK) optical flow [11] with five pyramid levels. The initial feature $\mathbf{z}^{i,k}$ location comes from the projection $\pi(\cdot)$ of previously tracked 3D points $\mathbf{p}_c^{i,k}$ with the predicted camera pose $\hat{\mathbf{T}}_{wc}^k$ assuming a constant velocity motion. Their corresponding keypoints are searched in the first two pyramids. If no 3D information is available, the initial position is set as the same as the previous feature observation and a three-level search is used instead. Failed tracks of the first stage are considered in the second search too. All correspondences are verified to ensure accurate tracking by applying backward tracking with an error threshold of 0.5 px.

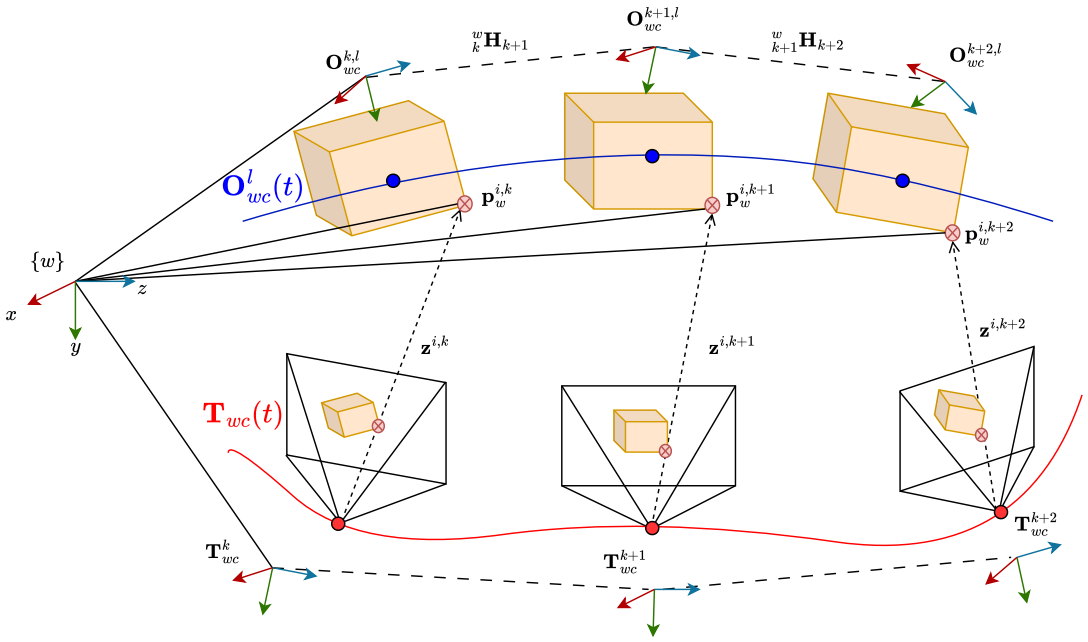

Fig. 3: **Interaction between camera poses and object poses**: Our system regresses the camera and object poses in B-Splines continuous trajectories $\mathbf{T}_{wc}(t)$ and $\mathbf{O}_{wc}^{l}(t)$ through the discrete poses are control points and 2D measurements $\mathbf{z}$ of the 3D point $\mathbf{p}$ at different timestamps.

Estimating the current camera pose $\mathbf{T}_{wc}^{k}$ involves minimizing the 3D reprojection error for a set of static landmarks $\mathcal{I}_k$ at t_k with the covariance matrix $\mathbf{\Sigma}_{i,k}$ and wrapped with the robust Huber loss function $\rho(\cdot)$ to mitigate outliers influence, in a Motion-only Bundle Adjustment (BA) solver where the points are fixed during optimization.

$$\mathbf{T}_{wc}^{k} = \arg\min_{\mathbf{T}_{wc}^{k}} \sum_{i \in \mathcal{I}_k} \rho(||\mathbf{e}_{i,k}||^2_{\mathbf{\Sigma}_{i,k}}) \tag{27}$$

$$\mathbf{e}_{i,k} = \mathbf{z}^{i,k} - \pi\left((\mathbf{T}_{wc}^{k})^{-1} \odot \mathbf{p}_{w}^{i}\right) \tag{28}$$

The optimization runs for ten iterations in four loops with the Levemberg-Marquadt algorithm, where on each loop, outliers are removed according to the χ^2 test of 2 DoF with 95% confidence.

B. Dynamic instance tracker

In dynamic environments, the front-end of a visual SLAM system may retain feature correspondences on moving objects despite applying geometric filters such as the geometric constraints used in tracking thread. This effect is particularly evident in edge cases, e.g., when a vehicle enters the vanishing point region, when camera motion lacks sufficient rotation, or when surrounding vehicles move at a similar velocity to the ego platform as seen in Fig. **??** . In such scenarios, feature points on dynamic objects often remain classified as inliers, contaminating the subsequent motion estimation.

Contrary to existing approaches that generate dense feature groups within regions of interest [2], [12] or optimize a dedicated pose for every detected object instance, both of which significantly increase the computational burden, we avoid redundant feature extraction and per-object optimization.

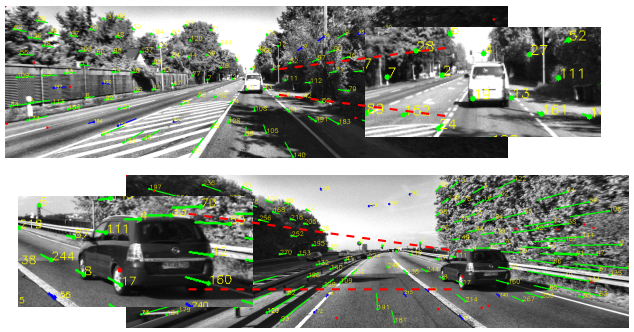

Fig. 4: **Edge Cases with Dynamic Object Inliers**. Common failure scenarios where feature points on dynamic objects are incorrectly classified as inliers after bundle adjustment. This occurs when: (a) vehicles approach the epipole (motion parallel to camera rays), or (b) objects move with velocity similar to the ego vehicle. Green lines represent inlier matches; red points indicate outlier observations. Manuel, please use our own dataset

Instead, we exploit the SLAM map points that are already reprojected and tracked in the image. These map points are intersected with semantic masks provided by YOLOv11 [13], pretrained on the COCO dataset [14], which reliably segments cars and people. To ensure temporal consistency across frames, we employ ByteTrack [15] for multi-object tracking, associating detection masks across consecutive frames. While YOLOv11 and ByteTrack are adopted in this work, our framework remains general and can accommodate alternative segmentation and tracking networks, provided their inference times allow real-time execution.

The primary limitation of this sparse approach is its dependence on feature distribution. It cannot reason about the motion of textureless regions on dynamic objects, unlike dense methods that leverage photometric consistency across entire masks. However, these dense methods incur a substantially higher computational cost, making our sparse strategy a pragmatic trade-off for real-time performance on resource-constrained platforms.

For motion-state inference, we employ a weighted decision rule that evaluates the displacement statistics of the associated keypoints, thereby classifying whether the object is static or dynamic. To mitigate spurious decisions due to noise or short-term inconsistencies, the per-frame classification is subsequently passed through a simple Bayes filter with temporal smoothing. Specifically, the belief over the state $\mathbf{s}^{j,k}$ of object j at time k is recursively updated as

$$\mathbf{p}_{\text{dyn}}^{j,k} = \frac{\mathbf{p}(\mathbf{o}^{j,k}|\mathbf{s}_{\text{dyn}}^{j,k}) \cdot \mathbf{p}_{\text{dyn}}^{j,k-1}}{\mathbf{p}(\mathbf{o}^{j,k}|\mathbf{s}_{\text{dyn}}^{j,k}) \cdot \mathbf{p}_{\text{dyn}}^{j,k-1} + \mathbf{p}(\mathbf{o}^{j,k}|\mathbf{s}_{\text{stat}}^{j,k}) \cdot (1 - \mathbf{p}_{\text{dyn}}^{j,k-1})} \tag{29}$$

where $o^{j,k}$ denotes the current observation derived from the keypoint displacement statistics, $\mathbf{p}(o^{j,k}|s_{\text{dyn}}^{j,k})$ and $\mathbf{p}(o^{j,k}|s_{\text{stat}}^{j,k})$ model the likelihood of the observation under the dynamic and static hypotheses, respectively, and $\mathbf{p}_{\text{dyn}}^{j,k-1}$ encodes the prior belief over the object's motion state, thereby enforcing temporal smoothness. Once an object is labeled as dynamic, its corresponding keypoints are excluded from the Motion-Only and full Bundle Adjustment stages, preventing dynamic inliers from biasing the optimization and ensuring that the trajectory estimation remains uncontaminated even in challenging urban scenarios. Importantly, these keypoints are not discarded from the tracking pipeline but are instead maintained, allowing the system to seamlessly reclassify objects that change motion states, such as vehicles that stop at traffic lights or resume motion in intersections.

To further enhance robustness, we introduce adaptive scaling rules for the likelihood terms based on the spatial distribution and count of static keypoints within each object mask. Specifically, when the number of static keypoints is low, which typically occurs for objects near the vanishing point or at large distances, we down-weight the static likelihood and up-weight the dynamic likelihood to reflect increased uncertainty. Conversely, when the mask contains a sufficient number of static keypoints, the static likelihood is strongly reinforced, indicating high confidence that the object is stationary. These rules leverage the spatial statistics of inlier features, allowing the filter to dynamically adjust its belief as the feature density evolves, and to resolve ambiguities in cases where objects transition between motion states or are observed under challenging geometric conditions.

C. Non-holonomic continuous back-end

The back-end is in charge of initializing the 3D map, creating keyframes when the inserting criteria are met, searching for new points, and refining the joint structure of landmarks and poses with local BA.

Initializing the map in a stereo setup is straightforward since triangulation between the two views can observe the scale factor due to known extrinsic parameters $\mathbf{T}_{lr}$. Stereo correspondences are found in the same manner as with feature tracking with the addition of three outliers filters: i) maximum vertical pixel error less than 2 px, ii) stereo points must be on the frustum with a minimum depth of 0.1 m, and iii) statistical filtering based on distance w.r.t to the median depth. The features in the right image are initialized by the projection of 3D points if available. Otherwise, it uses a sub-pixel Sum of Absolute Differences (SAD) line search at the highest pyramid level (assuming that images are rectified).

New keyframes are accepted into the back-end module if: i) less than 50% of the previous keyframe features were tracked, ii) the cosine of the median parallax angle between the current view and last keyframe is less than 0.9998, and iii) a minimum displacement of 0.1 m w.r.t to last keyframe. When a new keyframe is created, successfully tracked features and points are transferred from the previous frame to maintain long-duration correspondences. The back-end detects a new set of features guided by a mask that restricts locations close to previous features with a distance of 15px such that they tend to be homogeneously distributed across the image, and execution time is reduced significantly.

For features that could not be stereo-triangulated, we search temporal matches within previous keyframes, as high connectivity graph is necessary to maintain robustness and results accuracy. Additionally, matched features with a larger baseline reduce the depth error. We utilize previous feature observations that were transferred in the keyframe creation, i.e., the oldest feature correspondence. Wrong associations are detected by reprojection error of the new triangulated point and parallax angle above a predefined threshold.

Correspondences coming only from tracking are extended with feature descriptor matching. We have to compute the BRIEF descriptor [16] of each feature and match them by the Hamming distance while following the covisibility rules of our previous work [17]. At last, all correspondences are represented in a graph form and optimized by a novel BA.

D. Object-level continuous BA

The optimization problem is formulated as the minimization of a non-linear least square error composed of a combination of factors $\mathbf{e}_i$ which measured the difference between measurements $\boldsymbol{z}_i$ and predictions $\boldsymbol{h_i}(\boldsymbol{x}_i)$. These factors are assumed to follow a Gaussian model with zero-mean and a covariance matrix $\boldsymbol{\Sigma}_i$. This procedure is equivalent to solving a Maximum Posterior (MAP) problem:

$$\phi(\boldsymbol{x}_i) \propto \exp\left(-\frac{1}{2}||\boldsymbol{e}_i||^2_{\boldsymbol{\Sigma}_i}\right) \tag{30}$$

$$\boldsymbol{e}_i = \boldsymbol{z}_i - \boldsymbol{h}_i(\boldsymbol{x}_i) \tag{31}$$

The factors are simplified by taking the negative log and obtaining a non-linear least square problem to compute the optimal solution $\boldsymbol{x}^{\text{MAP}}$:

$$\boldsymbol{x}^{\text{MAP}} = \arg\min_{\boldsymbol{x}} \sum_i ||\boldsymbol{e}_i||^2_{\boldsymbol{\Sigma}_i} \tag{32}$$

In the following subsections we explained the derivation of the required factors and their Jacobians.

1) Reprojection factor: Coming from the Structure-from-Motion field, the reprojection factor quantifies the difference between a keypoint location and the projection of its associate 3D landmark in t_k. This error assumes static scenes; thus, dynamic landmarks must be omitted for this error term. Instead of a discrete camera pose, the factor evaluates the pose spline at t_k thus making it dependent on the control points and adjusting them during the optimization.

$$\mathbf{e}_{\text{reproj}} = \mathbf{z}^{i,k} - \pi((\mathbf{T}_{wc}(t_k))^{-1} \odot \mathbf{p}_w^i) \tag{33}$$

2) Continuous object motion factor: There are several formulations for the object motion factor such as the factor error is associated to the 3D point across consecutive timestamps or if the object motion is continuously parametrized. DynoSAM [2] was a milestone work that provided several insights on why object-centric motion factors fail to model kinematics. In consequence, we must choose the world-centric formulation and hence, applying motion updates using left-side Lie group composition. This formulation has been taken previously by MVO [18] using Gaussian Processes. In terms of Spline-based algorithms, they tend to utilize the object-centric factor, probably due to its compatibility with local (body) measurements for sensor fusion such as from an IMU. Due to those reasons, we introduce a world-centric spline-based object motion factor. As we aim to utilize open-source spline solvers, we consider that the motion is modeled by local updates and we translate them to world frame through the $SE(3)$ Adjoint [19], [20].

$$\begin{aligned}\mathbf{T}_{wc}^{k} &:= \mathbf{T}_{wc}^{k-1}\left({}_{k-1}^{k-1}\mathbf{H}_k\right) \\ &= \text{Exp}(\mathbf{Adj}_{\mathbf{T}_{wc}^{k-1}} \text{Log}\left({}_{k-1}^{k-1}\mathbf{H}_k\right))\left(\mathbf{T}_{wc}^{k-1}\right) \\ &= {}_{k-1}^{w}\mathbf{H}_k(\mathbf{T}_{wc}^{k-1})\end{aligned} \tag{34}$$

Assuming a continuous parametrization, the discrete motion is interpolated from the generalized velocity curve $\boldsymbol{\varpi}_c$ at t_{k-1}.

$$\text{Exp}(\mathbf{Adj}_{\mathbf{T}_{wc}(t_{k-1})}\boldsymbol{\varpi}_c(t_{k-1})\Delta t) = {}_{k-1}^{w}\mathbf{H}_k \tag{35}$$

This derivation is powerful as it means that we can either used left or right increments on the object pose as long as we define the 3D points w.r.t to the world frame $\{w\}$.

Let mark some clear differences with previous works: i) DynoSAM [2] introduced world-centric discrete motion factors with the addition of a smoother factor, and ii) Tirado & Civera [3] model the object motion with splines but with local-centric error.

$$\mathbf{e}_{\text{H}} = \mathbf{p}_w^{i,k} - \text{Exp}(\mathbf{Adj}_{\mathbf{O}_{wc}^{l}(t_{k-1})}\boldsymbol{\varpi}_{c,l}(t_{k-1})\Delta t) \odot \mathbf{p}_w^{i,k-1} \tag{36}$$

An advantage of our spline factor is that we do not smooth the motion with an additional constraint factor.

3) Non-holonomic factor: Our new factor takes the non-holonomic constraint derived for wheeled robots without access to the steering angle (as a general case) from its kinematic model. In contrast to the R-v constraint [21], this factor shows experimentally better robustness against high levels of feature noise and has a lower dimensionality in parameters, critical for problem scaling. The factor evaluates Eq.26 with the evaluated pose curve $\mathbf{T}_{wc}(t_k)$ and evaluated velocity curve $\dot{\mathbf{T}}_{wc}(t_k)$, both coming from the same control points. The necessary motion components are extracted by the generators $\mathbf{G}_x, \mathbf{G}_y, \mathbf{G}_\theta$ over the tangent space $\mathfrak{se}(3)$.

$$\begin{aligned}\mathbf{e}_{\text{non-holo}} &= \cos(\boldsymbol{\theta}_z)\mathbf{G}_x\dot{\mathbf{T}}_{wc}(t_k) - \sin(\boldsymbol{\theta}_z)\mathbf{G}_y\dot{\mathbf{T}}_{wc}(t_k) \\ \boldsymbol{\theta}_z &= \mathbf{G}_\theta \,\text{Log}(\mathbf{T}_{wc}(t_k))\end{aligned}$$

The complete optimization problem (Fig.5) has the final form of:

$$\begin{aligned}\boldsymbol{\xi}_k^* = \arg\min_{\boldsymbol{\theta}_k}(&||\mathbf{e}_0||^2_{\Sigma_0} \\ &+ ||\mathbf{e}_{\text{non-holo}}||^2_{\Sigma_{\text{non-holo}}} \\ &+ \sum_{i\in\mathcal{I}_k} \rho\left(||\mathbf{e}_{\text{reproj}}||^2_{\Sigma_{\text{reproj}}}\right) \\ &+ \sum_{i\in\mathcal{D}_k} \rho\left(||\mathbf{e}_{\text{reproj}}||^2_{\Sigma_{\text{reproj}}}\right) \\ &+ \sum_{l\in\updownarrow\mathcal{L}_k}\sum_{i\in\mathcal{D}_k^j} \rho\left(||\mathbf{e}_{\text{H}}||^2_{\Sigma_{\text{H}}}\right)\end{aligned} \tag{37}$$

with the parameters $\boldsymbol{\xi} = \{\boldsymbol{\theta}_k | k \in \mathcal{K}\}$, $\boldsymbol{\theta}_k = \{\mathcal{P}_w^k, \mathbf{c}^k\}$, where $\mathcal{K}$ represents a set of timestamps in a sliding window fashion, $\mathbf{e}_0$ is the fixed vehicle pose prior, and $\mathcal{L}_k$ is the set of moving objects at time instant k^{th}. We solve BA again with the Levemberg-Marquadt algorithm on the Ceres solver [22] to exploit its feature of automatic differentiation to compute Jacobians of extensive expressions such as the spline-dependent factors.

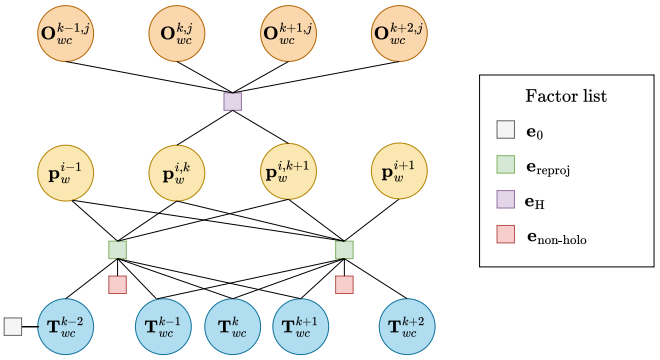

Fig. 5: **Non-holonomic continuous factor graph**. The factor graph embeds the discrete poses as control points for the spline and connects the parameters by factors (□). Reprojection factors are in green, object motion factor is in purple, and *new non-holonomic factor* is in red color.

IV. Jacobian Derivation

In order to solve Eq.37, we compute state updates $\Delta\boldsymbol{\xi}^*$ over the parameters by linearization and then solving the linear problem $\mathbf{H}\Delta\boldsymbol{\xi} = \mathbf{g}$ where $\mathbf{H}$ and $\mathbf{g}$ are the Hessian and gradient of the factor respectively. Linearizing requires to compute the factor Jacobians w.r.t to the control points $\mathbf{d}_j, j \in [0, 1, 2, 3]$ of a cubic B-Spline. Let investigate each factor derivation:

A. Reprojection error

The Jacobian w.r.t $\mathbf{d}_j$ is composed of the rule of chain:

$$\frac{\partial \mathbf{e}_{\text{reproj}}}{\partial \mathbf{d}_j} = -\frac{\partial \pi(\cdot)}{\partial \mathbf{a}} \frac{\partial \mathbf{a}}{\partial \mathbf{T}_{wc}(t_k)} \frac{\partial \mathbf{T}_{wc}(t_k)}{\partial \mathbf{d}_j} \tag{38}$$

where $\mathbf{a} = \pi((\mathbf{T}_{wc}(t_k))^{-1} \odot \mathbf{p}_w^i)$. The leftmost Jacobian is straightforward to compute since it does not act the $\mathbb{SE}(3)$ manifold and it is a well known product of the Structure-from-Motion (SfM) problem.

$$\frac{\partial \pi(\cdot)}{\partial \mathbf{a}} = \frac{1}{z} \begin{bmatrix} 1 & 0 & \frac{-x}{z} \\ 0 & 1 & -\frac{y}{z} \end{bmatrix} \tag{39}$$

For the middle Jacobian $\frac{\partial \mathbf{a}}{\partial \mathbf{T}_{wc}(t_k)}$, we identify it as the Jacobian of the action of an inverse pose into $\mathbb{R}^3$ (see Appendix A-A):

$$\frac{\partial \mathbf{a}}{\partial \mathbf{T}_{wc}(t_k)} = \begin{bmatrix} -\mathbf{I}_{3\times 3} & \left((\mathbf{T}_{wc}(t_k))^{-1} \odot \mathbf{p}_w^i\right)^\wedge \end{bmatrix} \tag{40}$$

The last Jacobian $\frac{\partial \mathbf{T}_{wc}(t_k)}{\partial \mathbf{d}_j}$ is provided by the spline solver in [4] for each control point $j = 0, 1, 2, 3$ through a recursive and computational efficient formula.

B. Continuous object motion factor

Let reintroduce the object motion factor with a slight different notation for better readability:

$$\mathbf{e}_{\text{H}} = \mathbf{p}_w^{i,k} - {}^{w}_{k-1}\mathbf{H}_k \odot \mathbf{p}_w^{i,k-1} \tag{41}$$

$${}^{w}_{k-1}\mathbf{H}^l_k = \text{Exp}(\boldsymbol{\varpi}_{w,l}(t_{k-1})\Delta t) \tag{42}$$

$$\boldsymbol{\varpi}_{w,l}(t_{k-1}) = \mathbf{Adj}_{\mathbf{O}^l_{wc}(t_{k-1})} \boldsymbol{\varpi}_{c,l}(t_{k-1}) \tag{43}$$

The rule of chain of the Jacobian w.r.t $\mathbf{d}_j$ (with omitted indexes) is:

$$\frac{\partial \mathbf{e}_{\text{H}}}{\partial \mathbf{d}_j} = -\frac{\partial \mathbf{H} \odot \mathbf{p}}{\partial \mathbf{H}} \left(\frac{\partial \mathbf{H}}{\partial \mathbf{O}} \frac{\partial \mathbf{O}}{\partial \mathbf{d}_j} + \frac{\partial \mathbf{H}}{\partial \boldsymbol{\varpi}_c} \frac{\partial \boldsymbol{\varpi}_c}{\partial \mathbf{d}_j} \right) \tag{44}$$

The leftmost Jacobian is simply the partial derivative of the $SE(3)$ action of a pose into $\mathbb{R}^3$ w.r.t to the pose (Appendix A-C).

$$\frac{\partial \mathbf{H} \odot \mathbf{p}}{\partial \mathbf{H}} = \begin{bmatrix} \mathbf{H}_{\mathbf{R}} & -\mathbf{H}_{\mathbf{R}} \mathbf{p}^\wedge \end{bmatrix} \tag{45}$$

where the rotation of $\mathbf{H}$ is extracted as $\mathbf{H}_{\mathbf{R}}$

For the Jacobian $\frac{\partial \mathbf{H}}{\partial \mathbf{O}}$, it requires applying the rule of chain again:

$$\frac{\partial \mathbf{H}}{\partial \mathbf{O}} = \frac{\partial \,\text{Exp}(\mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t)}{\partial \mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t} \frac{\partial \mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t}{\partial \mathbf{O}} \tag{46}$$

These two Jacobians have closed forms well known from Lie Group Theory. Be refereed to Appendix A-D for the Jacobian of the adjoint matrix w.r.t the pose $\frac{\partial \mathbf{Adj}_{\mathbf{T}} \boldsymbol{\xi}}{\partial \mathbf{T}}$. For the Jacobian of Exp map, it is equivalent to taking an infinitesimal step over $SE(3)$ group which results in the Left Group Jacobian $\frac{\partial \,\text{Exp}(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \mathbf{J}_l(\boldsymbol{\xi})$ (see [1] for its closed form). Multiplying to the two both gets:

$$\frac{\partial \mathbf{H}}{\partial \mathbf{O}} = -\mathbf{J}_l(\mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t) \mathbf{Adj}_{\mathbf{O}} \mathbf{adj}_{\boldsymbol{\varpi}_c \Delta t} \tag{47}$$

A similar derivation is carry out for jacobian $\frac{\partial \mathbf{H}}{\partial \boldsymbol{\varpi}_c}$ by using the rule of chain once more:

$$\frac{\partial \mathbf{H}}{\partial \boldsymbol{\varpi}_C} = \frac{\partial \,\text{Exp}(\mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t)}{\partial \mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t} \frac{\partial \mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t}{\partial \boldsymbol{\varpi}_{\mathbf{C}}} \tag{48}$$

Notice that the $\mathbf{Adj}_{\mathbf{O}}$ is not dependent on $\boldsymbol{\varpi}_c$ hence can be treated as a constant.

$$\frac{\partial \mathbf{H}}{\partial \boldsymbol{\varpi}_C} = \mathbf{J}_l(\mathbf{Adj}_{\mathbf{O}} \boldsymbol{\varpi}_c \Delta t) \mathbf{Adj}_{\mathbf{O}} \Delta t \tag{49}$$

The remaining Jacobians $\frac{\partial \mathbf{O}}{\partial \mathbf{d}_j}, \frac{\partial \boldsymbol{\varpi}_c}{\partial \mathbf{d}_j}$ are queried from the Spline solver.

C. Non-holonomic factor

The Jacobian w.r.t $\mathbf{d}$ has a couple dependence on the pose $\mathbf{T}$ and its derivative $\dot{\mathbf{T}}$:

$$\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{d}_j} = \frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \dot{\mathbf{T}}} \frac{\partial \dot{\mathbf{T}}}{\partial \mathbf{d}_j} + \frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{T}} \frac{\partial \mathbf{T}}{\partial \mathbf{d}_j} \tag{50}$$

First, we focus on the Jacobians that are dependent on $\dot{\mathbf{T}}$. For $\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \dot{\mathbf{T}}}$, the result is straightforward since all coefficients are decoupled and can be treated as a constant.

$$\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \dot{\mathbf{T}}} = \cos(\boldsymbol{\theta}_z)\mathbf{G}_x - \sin(\boldsymbol{\theta}_z)\mathbf{G}_y \tag{51}$$

Conversely, $\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{T}}$ takes several derivations and abusing the chain of rule slightly:

$$\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{T}} = \mathbf{G}_x \dot{\mathbf{T}} \frac{\partial \cos(\boldsymbol{\theta}_z)}{\partial \mathbf{T}} - \mathbf{G}_y \dot{\mathbf{T}} \frac{\partial \sin(\boldsymbol{\theta}_z)}{\partial \mathbf{T}}$$

where:

$$\frac{\partial \cos(\boldsymbol{\theta}_z)}{\partial \mathbf{T}} = -\sin(\boldsymbol{\theta}_z)\, \mathbf{G}_\theta \frac{\partial \,\text{Log}(\mathbf{T})}{\partial \mathbf{T}}$$

$$\frac{\partial \sin(\boldsymbol{\theta}_z)}{\partial \mathbf{T}} = \cos(\boldsymbol{\theta}_z)\, \mathbf{G}_\theta \frac{\partial \,\text{Log}(\mathbf{T})}{\partial \mathbf{T}}$$

The Jacobian of Log map w.r.t to the $SE(3)$ pose $\frac{\partial \,\text{Log}(\mathbf{T})}{\partial \mathbf{T}}$ is found in Appendix.A-B

As with the previous factor, the remaining Jacobians $\frac{\partial \dot{\mathbf{T}}}{\partial \mathbf{d}_j}, \frac{\partial \mathbf{T}}{\partial \mathbf{d}_j}$ are queried from the Spline solver.

D. Non-holonomic factor with extrinsics

Let redefine the error function that models the non-holonomic constraints with awareness of the extrinsics between the rear-wheels frame, where the constraints are actually defined, and the camera frame.

$$\mathbf{e}_{\text{non-holo}} = \cos(\boldsymbol{\theta}_z)\mathbf{G}_x \dot{\mathbf{T}}_{w_b b}(t_k) - \sin(\boldsymbol{\theta}_z)\mathbf{G}_y \dot{\mathbf{T}}_{w_b b}(t_k) \tag{52}$$

where $\dot{\mathbf{T}}_{w_b b}$ is the rate of change of vehicle pose $\mathbf{T}_{w_b b}$ w.r.t to the world vehicle frame $\{w_b\}$. As with the time derivative, the pose and the generalized velocity also need to be changed of basis with the extrinsics $\mathbf{T}_{bc}$.

$$\boldsymbol{\theta}_z = \mathbf{G}_\theta \,\text{Log}(\mathbf{T}_{w_b b}(t_k)) \tag{53}$$

$$\mathbf{T}_{w_b b} = \mathbf{T}_{bc} \mathbf{T}_{wc} \mathbf{T}_{bc}^{-1} \tag{54}$$

$$\dot{\mathbf{T}}_{w_b b} = \boldsymbol{\varpi}^\wedge_{w_b} \mathbf{T}_{w_b b} \tag{55}$$

$$\boldsymbol{\varpi}_{w_b} = \mathbf{Adj}_{\mathbf{T}_{bc}} \boldsymbol{\varpi}_c \tag{56}$$

The Jacobian derivation follows a similar procedure as without extrinsics, with the addition of longer chain of rule for the Jacobians w.r.t the control points $\mathbf{d}_j$.

$$\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{d}_j} = \frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \dot{\mathbf{T}}_{w_b b}} \frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \mathbf{d}_j} + \frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{T}_{w_b b}} \frac{\partial \mathbf{T}_{w_b b}}{\partial \mathbf{d}_j} \tag{57}$$

The first two Jacobians are:

$$\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \dot{\mathbf{T}}_{w_b b}} = \cos(\boldsymbol{\theta}_z)\mathbf{G}_x - \sin(\boldsymbol{\theta}_z)\mathbf{G}_y \tag{58}$$

$$\frac{\partial \mathbf{e}_{\text{non-holo}}}{\partial \mathbf{T}_{w_b b}} = \mathbf{G}_x \dot{\mathbf{T}}_{w_b b} \frac{\partial \cos(\boldsymbol{\theta}_z)}{\partial \mathbf{T}_{w_b b}} - \mathbf{G}_y \dot{\mathbf{T}}_{w_b b} \frac{\partial \sin(\boldsymbol{\theta}_z)}{\partial \mathbf{T}_{w_b b}} \tag{59}$$

where:

$$\frac{\partial \cos(\boldsymbol{\theta}_z)}{\partial \mathbf{T}_{w_b b}} = -\sin(\boldsymbol{\theta}_z)\mathbf{G}_\theta \frac{\partial \operatorname{Log}(\mathbf{T}_{w_b b})}{\partial \mathbf{T}_{w_b b}} \tag{60}$$

$$\frac{\partial \sin(\boldsymbol{\theta}_z)}{\partial \mathbf{T}_{w_b b}} = \cos(\boldsymbol{\theta}_z)\mathbf{G}_\theta \frac{\partial \operatorname{Log}(\mathbf{T}_{w_b b})}{\partial \mathbf{T}_{w_b b}} \tag{61}$$

The Jacobians w.r.t control points $\mathbf{d}_j$:

$$\frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \mathbf{d}_j} = \frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \boldsymbol{\varpi}_{w_b}} \frac{\partial \boldsymbol{\varpi}_{w_b}}{\partial \mathbf{d}_j} + \frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \mathbf{T}_{w_b b}} \frac{\partial \mathbf{T}_{w_b b}}{\partial \mathbf{d}_j} \tag{62}$$

The Jacobian $\frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \boldsymbol{\varpi}_{w_b}}$ is quite problematic since the $SE(3)$ parametrization does not work since $\dot{\mathbf{T}} \in \mathbf{R}^{4\times4}$. Instead, we vectorized the matrix $\dot{\mathbf{T}}_{w_b b}$ by collapsing all its columns in a row vector $(\dot{\mathbf{T}})_{vec} = (\mathbf{T}^\top \otimes \mathbf{I}_{4\times4})(\boldsymbol{\varpi})_{vec}$ where $\otimes$ is the Kronecker product [23]. The vector $(\boldsymbol{\varpi})_{vec}$ could be mapped linearly as $(\boldsymbol{\varpi})_{vec} = \mathbf{L}\boldsymbol{\varpi}, \mathbf{L} \in \mathbb{R}^{16\times6}$. Therefore, the Jacobian ends as:

$$\frac{\partial (\dot{\mathbf{T}}_{w_b b})_{vec}}{\partial \boldsymbol{\varpi}_{w_b}} = (\mathbf{T}^\top \otimes \mathbf{I}_{4\times4})\mathbf{L} \tag{63}$$

To recover the Jacobian $\frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \boldsymbol{\varpi}_{w_b}}$, we reshape the vectorized Jacobian in a tensor form $\mathbb{R}^{4\times4\times6}$.

The next Jacobian $\frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \mathbf{T}_{w_b b}}$ is effortless computed since $\dot{\mathbf{T}} = \boldsymbol{\varpi}^\wedge \mathbf{T}$.

$$\frac{\partial \dot{\mathbf{T}}_{w_b b}}{\partial \mathbf{T}_{w_b b}} = \boldsymbol{\varpi}^\wedge_{w_b} \tag{64}$$

The remaining Jacobians w.r.t $\mathbf{d}_j$ have a coupling between the extrinsics $\mathbf{T}_{bc}$ and camera-related pose and velocity, thus needing an additional chain of rule:

$$\frac{\partial \boldsymbol{\varpi}_{w_b}}{\partial \mathbf{d}_j} = \frac{\partial \boldsymbol{\varpi}_{w_b}}{\partial \boldsymbol{\varpi}_c} \frac{\partial \boldsymbol{\varpi}_c}{\partial \mathbf{d}_j} \tag{65}$$

$$\frac{\partial \mathbf{T}_{w_b b}}{\partial \mathbf{d}_j} = \frac{\partial \mathbf{T}_{w_b b}}{\partial \mathbf{T}_{wc}} \frac{\partial \mathbf{T}_{wc}}{\partial \mathbf{d}_j} \tag{66}$$

where:

$$\frac{\partial \boldsymbol{\varpi}_{w_b}}{\partial \boldsymbol{\varpi}_c} = \mathbf{Adj}_{\mathbf{T}_{bc}} \tag{67}$$

and (full derivation in Appendix.A-E):

$$\frac{\partial \mathbf{T}_{w_b b}}{\partial \mathbf{T}_{wc}} = \mathbf{J}_l(\mathbf{Adj}_{\mathbf{T}_{bc}} \operatorname{Log}(\mathbf{T}_{wc}))\mathbf{Adj}_{\mathbf{T}_{bc}} \mathbf{J}_r(\operatorname{Log}(\mathbf{T}_{wc}) \tag{68}$$

V. Experimental results and discussion

The computational efficiency and performance of the estimator was analyzed thorough.

VI. Conclusions

A state

VII. Acknowledgement

The authors thank

Appendix A
Jacobians in Lie Group

A. Jacobian of action into $\mathbb{R}^3$ w.r.t $\mathbf{T}^{-1}$

@Marcelo, please re-check the fifth row of the following derivation

$$\begin{aligned}
\frac{\partial \mathbf{T}^{-1} \odot \mathbf{p}}{\partial \mathbf{T}} &= \frac{\partial \left(\mathbf{T} \operatorname{Exp}(\boldsymbol{\xi})\right)^{-1} \odot \mathbf{p}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&= \frac{\partial \operatorname{Exp}(-\boldsymbol{\xi})\mathbf{T}^{-1} \odot \mathbf{p}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\overset{\text{Eq.6}}{\approx} \frac{\partial \left(\mathbf{I} - \boldsymbol{\xi}^\wedge\right)\mathbf{T}^{-1} \odot \mathbf{p}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx -\frac{\partial \boldsymbol{\xi}^\wedge \left(\mathbf{R}^\top\mathbf{p} - \mathbf{R}^\top\mathbf{t}\right)}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx -\frac{\partial \left(\boldsymbol{\phi}^\wedge \left(\mathbf{R}^\top\mathbf{p} - \mathbf{R}^\top\mathbf{t}\right) + \boldsymbol{\rho}\right)}{\partial \begin{bmatrix}\boldsymbol{\rho} & \boldsymbol{\phi}\end{bmatrix}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx -\frac{\partial \left(-\left(\mathbf{R}^\top\mathbf{p} - \mathbf{R}^\top\mathbf{t}\right)^\wedge \boldsymbol{\phi} + \boldsymbol{\rho}\right)}{\partial \begin{bmatrix}\boldsymbol{\rho} & \boldsymbol{\phi}\end{bmatrix}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \begin{bmatrix}-\mathbf{I}_{3\times3} & \left(\mathbf{T}^{-1} \odot \mathbf{p}\right)^\wedge\end{bmatrix}
\end{aligned}$$

where we utilized the property $\boldsymbol{a}^\wedge\boldsymbol{b} = -\boldsymbol{b}^\wedge\boldsymbol{a}$.

B. Jacobian of $\operatorname{Log}(\cdot)$ w.r.t $\mathbf{T}$

$$\begin{aligned}
\frac{\partial \operatorname{Log}(\mathbf{T})}{\partial \mathbf{T}} &= \frac{\partial \operatorname{Log}\left(\mathbf{T} \operatorname{Exp}(\boldsymbol{\xi})\right)}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&= \frac{\partial \operatorname{Log}\left(\operatorname{Exp}\left(\operatorname{Log}(\mathbf{T})\right)\operatorname{Exp}(\boldsymbol{\xi})\right)}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\overset{\text{Eq.13}}{\approx} \frac{\partial \left(\operatorname{Log}(\mathbf{T}) + \mathbf{J}_r^{-1}\left(\operatorname{Log}(\mathbf{T})\right)\boldsymbol{\xi}\right)}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \mathbf{J}_r^{-1}\left(\operatorname{Log}(\mathbf{T})\right)
\end{aligned}$$

C. Jacobian of Group action w.r.t $\mathbf{T}$

$$\begin{aligned}
\frac{\partial \mathbf{T} \odot \mathbf{p}}{\partial \mathbf{T}} &= \frac{\partial \mathbf{T} \operatorname{Exp}(\boldsymbol{\xi}) \odot \mathbf{p}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\overset{\text{Eq.6}}{\approx} \frac{\partial \mathbf{T}\left(\mathbf{I} + \boldsymbol{\xi}^\wedge\right) \odot \mathbf{p}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \frac{\partial \mathbf{T}\boldsymbol{\xi}^\wedge \odot \mathbf{p}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \frac{\partial \mathbf{T}\left(\boldsymbol{\phi}^\wedge\mathbf{p} + \boldsymbol{\rho}\right)}{\partial \begin{bmatrix}\boldsymbol{\rho} & \boldsymbol{\phi}\end{bmatrix}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \frac{\partial \mathbf{R}\left(\boldsymbol{\phi}^\wedge\mathbf{p} + \boldsymbol{\rho}\right) + \mathbf{t}}{\partial \begin{bmatrix}\boldsymbol{\rho} & \boldsymbol{\phi}\end{bmatrix}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \frac{\partial \mathbf{R}\left(-\mathbf{p}^\wedge\boldsymbol{\phi} + \boldsymbol{\rho}\right) + \mathbf{t}}{\partial \begin{bmatrix}\boldsymbol{\rho} & \boldsymbol{\phi}\end{bmatrix}}|_{\boldsymbol{\xi}=\mathbf{0}} \\
&\approx \begin{bmatrix}\mathbf{R} & -\mathbf{R}\mathbf{p}^\wedge\end{bmatrix}
\end{aligned}$$

D. Jacobians of $\mathbf{Adj}$

$$\begin{aligned}\frac{\partial \mathbf{Adj_T}\boldsymbol{\xi}_b}{\partial \mathbf{T}} &= \frac{\partial \mathbf{Adj}_{\mathbf{T}\,\mathrm{Exp}(\boldsymbol{\xi})}\boldsymbol{\xi}_b}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=0} \\ &= \frac{\partial \mathbf{Adj_T}\mathbf{Adj}_{\mathrm{Exp}(\boldsymbol{\xi})}\boldsymbol{\xi}_b}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=0} \\ &\overset{\text{Eq.11}}{\approx} \frac{\partial \mathbf{Adj_T}\left(\mathbf{I}+\mathbf{adj}_{\boldsymbol{\xi}}\right)\boldsymbol{\xi}_b}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=0} \\ &\approx \frac{\partial \mathbf{Adj_T}\mathbf{adj}_{\boldsymbol{\xi}}\boldsymbol{\xi}_b}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=0} \\ &\approx -\frac{\partial \mathbf{Adj_T}\mathbf{adj}_{\boldsymbol{\xi}_b}\boldsymbol{\xi}}{\partial \boldsymbol{\xi}}|_{\boldsymbol{\xi}=0} \\ &\approx -\mathbf{Adj_T}\mathbf{adj}_{\boldsymbol{\xi}_b}\end{aligned}$$

We utilized the commutation trick $\mathbf{adj}_{\boldsymbol{\xi}_1}\boldsymbol{\xi}_2 = -\mathbf{adj}_{\boldsymbol{\xi}_2}\boldsymbol{\xi}_1$ between the arguments of $\mathbf{adj}$.

E. Jacobians of $\mathbf{T}_0\mathbf{T}_1\mathbf{T}_0^{-1}$ w.r.t $\mathbf{T}_1$

$$\begin{aligned}\frac{\partial \mathbf{T}_0\mathbf{T}_1\mathbf{T}_0^{-1}}{\partial \mathbf{T}_1} &= \frac{\partial \,\mathrm{Exp}(\mathbf{Adj}_{\mathbf{T}_0}\,\mathrm{Log}(\mathbf{T}_1))}{\partial \mathbf{T}_1} \\ &= \frac{\partial \,\mathrm{Exp}(E)}{\partial E}\frac{\partial E}{\partial\,\mathrm{Log}(\mathbf{T}_1)}\frac{\partial\,\mathrm{Log}\,\mathbf{T}_1}{\partial \mathbf{T}_1} \\ &= \mathbf{J}_l(\mathbf{Adj}_{\mathbf{T}_0}\,\mathrm{Log}(\mathbf{T}_1))\mathbf{Adj}_{\mathbf{T}_0}\mathbf{J}_r^{-1}(\mathrm{Log}(\mathbf{T}_1))\end{aligned}$$

where $E \triangleq \mathbf{Adj}_{\mathbf{T}_0}\,\mathrm{Log}(\mathbf{T}_1)$

APPENDIX B
JACOBIANS W.R.T CONTROL POINTS IN B-SPLINE

APPENDIX C
JACOBIANS OF CONTINUOUS VELOCITY IN $SE(3)$

APPENDIX D
PSEUDO-GENERATORS OF $SE(3)$

$$\mathbf{G}_x = \begin{bmatrix}1\\0\\0\\0\\0\\0\end{bmatrix}^\top, \mathbf{G}_y = \begin{bmatrix}0\\1\\0\\0\\0\\0\end{bmatrix}^\top, \mathbf{G}_{\theta_z} = \begin{bmatrix}0\\0\\0\\0\\0\\1\end{bmatrix}^\top \tag{69}$$

REFERENCES

[1] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[2] J. Morris, Y. Wang, M. Kliniewski, and V. Ila, "Dynosam: Open-source smoothing and mapping framework for dynamic slam," 2025. [Online]. Available: https://arxiv.org/abs/2501.11893

[3] J. Tirado and J. Civera, "Jacobian computation for cumulative b-splines on se(3) and application to continuous-time object tracking," 2022. [Online]. Available: https://arxiv.org/abs/2201.10602

[4] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative b-splines on lie groups," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] C. de Boor, "On calculating with b-splines," *Journal of Approximation Theory*, vol. 6, no. 1, pp. 50–62, 1972. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021904572900809

[6] K. Qin, "General matrix representations for b-splines," in *Proceedings Pacific Graphics '98. Sixth Pacific Conference on Computer Graphics and Applications (Cat. No.98EX208)*, 1998, pp. 37–43.

[7] E. Mueggler, G. Gallego, H. Rebecq, and D. Scaramuzza, "Continuous-time visual-inertial odometry for event cameras," *Trans. Rob.*, vol. 34, no. 6, p. 1425–1440, Dec. 2018. [Online]. Available: https://doi.org/10.1109/TRO.2018.2858287

[8] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots, second edition*. MIT Press, 2 2011.

[9] M. Ferrera, A. Eudes, J. Moras, M. Sanfourche, and G. Le Besnerais, "Ov^{2}slam: A fully online and versatile visual slam for real-time applications," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1399–1406, 2021.

[10] J. Shi and Tomasi, "Good features to track," in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[11] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, ser. IJCAI'81. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981, p. 674–679.

[12] I. Ballester, A. Fontán, J. Civera, K. H. Strobl, and R. Triebel, "Dot: Dynamic object tracking for visual slam," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 705–11 711.

[13] G. Jocher and J. Qiu, "Ultralytics yolo11," 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[14] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015. [Online]. Available: https://arxiv.org/abs/1405.0312

[15] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," 2022. [Online]. Available: https://arxiv.org/abs/2110.06864

[16] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 778–792.

[17] M. Contreras, N. P. Bhatt, and E. Hashemi, "Dynanav-svo: Dynamic stereo visual odometry with semantic-aware perception for autonomous navigation," *IEEE Transactions on Intelligent Vehicles*, pp. 1–12, 2024.

[18] K. M. Judd and J. D. Gammell, "Multimotion visual odometry," *The International Journal of Robotics Research*, vol. 43, no. 8, p. 1250–1278, Apr. 2024. [Online]. Available: http://dx.doi.org/10.1177/02783649241229095

[19] R. M. Murray, S. S. Sastry, and L. Zexiang, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. USA: CRC Press, Inc., 1994.

[20] G. S. Chirikjian, R. Mahony, S. Ruan, and J. Trumpf, "Pose changes from a different point of view," *Journal of Mechanisms and Robotics*, vol. 10, no. 2, p. 021008, 02 2018. [Online]. Available: https://doi.org/10.1115/1.4039121

[21] K. Huang, Y. Wang, and L. Kneip, "B-splines for purely vision-based localization and mapping on non-holonomic ground vehicles," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 5374–5380.

[22] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," 10 2023. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[23] J. L. Blanco-Claraco, "A tutorial on $\mathbf{SE}(3)$ transformation parameterizations and on-manifold optimization," 2022. [Online]. Available: https://arxiv.org/abs/2103.15980