

Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-Party Motions

Kevin M. Judd¹ and Jonathan D. Gammell¹ and Paul Newman¹

Abstract—Estimating motion from images is a well-studied problem in computer vision and robotics. Previous work has developed techniques to estimate the motion of a moving camera in a largely static environment (e.g., visual odometry) and to segment or track motions in a dynamic scene using known camera motions (e.g., multiple object tracking).

It is more challenging to estimate the unknown motion of the camera and the dynamic scene simultaneously. Most previous work requires *a priori* object models (e.g., tracking-by-detection), motion constraints (e.g., planar motion), or fails to estimate the full $SE(3)$ motions of the scene (e.g., scene flow). While these approaches work well in specific application domains, they are not generalizable to unconstrained motions.

This paper extends the traditional visual odometry (VO) pipeline to estimate the full $SE(3)$ motion of both a stereo/RGB-D camera and the dynamic scene. This multimotion visual odometry (MVO) pipeline requires no *a priori* knowledge of the environment or the dynamic objects. Its performance is evaluated on a real-world dynamic dataset with ground truth for all motions from a motion capture system.

I. INTRODUCTION

Visual navigation is an important area of research in robotics. Visual odometry (VO) estimates the motion of a camera (i.e., its egomotion) relative to observed static objects within a scene [1]. These static objects must be accurately segmented from any dynamic noise, and this segmentation itself is an area of research focus [2]. Less research in visual navigation has focused on also analyzing the dynamic regions of the scene that these approaches reject.

This motion estimation problem requires both *estimation*, i.e., calculating the motion of a set of points, and *segmentation*, i.e., clustering points according to their movement between observations. The interdependence of these tasks creates a *chicken-and-egg* problem that is addressed in VO systems by using heuristics (e.g., number of features) to select the egomotion and ignore the other motions in the scene. These heuristics are not readily extensible to *multimotion* estimation problems and analyzing multiple independently moving bodies remains a challenging problem for state-of-the-art vision systems. This paper extends traditional VO to multimotion visual odometry (MVO) and applies state-of-the-art techniques to estimate trajectories for *every* motion in a scene.

MVO applies multimodel fitting techniques (e.g., CORAL [3]) to the traditional VO pipeline to simultaneously estimate the trajectories of all motions within a scene. Sparse, 3D visual features are decomposed into independent rigid motions and the trajectories of all of these motions, including the egomotion of the camera, are estimated simultaneously

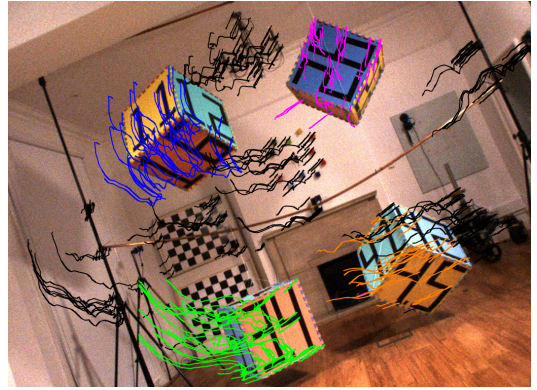


Fig. 1. Motion segmentation produced by our multimotion visual odometry (MVO) system. The egomotion of the camera is estimated from the static points in the scene shown in black. The other colors represent the segmentation of the other motions in the scene.

(Fig. 1). This paper demonstrates MVO on a stereo camera, but the technique is applicable to a variety of other 3D sensors, including RGB-D cameras and lidar. To the best of our knowledge, this is the first approach capable of estimating the full $SE(3)$ trajectory of every rigid motion in a complex, dynamic scene from a stereo/RGB-D camera without relying on simplifying constraints or fragile initialization.

II. BACKGROUND

The constituent aspects of the multimotion estimation problem are often referred to as multiple object tracking (MOT) [4] and multibody structure from motion (MBSfM) [5]. As such, the majority of approaches focus primarily on one part of the problem and either do not fully estimate $SE(3)$ motions (Secs. II-B and II-C), depend on simplifying constraints and assumptions (Secs. II-D and II-E), or require fragile initialization steps (Sec. II-F).

A. Problem Definition

Discrete multimotion estimation is the problem of estimating all the motions, including the camera, in a scene from a set of point observations at each time step. It both estimates the motions as a series of discrete $SE(3)$ transforms and associates the observed tracklets with the estimated motions.

Dynamic environments consist of the static background, a moving observer, i.e., the camera, and one or more independent, third-party motions. The pose of a motion, ℓ , at each discrete time, k , is represented as a coordinate frame, \mathcal{F}_{ℓ_k} , and related to a privileged initial pose through an $SE(3)$ transform, $\mathbf{T}_{\ell_k \ell_1}$ (Fig. 2a). A sequence of these transforms over a set of K frames constitutes the trajectory of the motion,

¹Oxford Robotics Institute, University of Oxford, United Kingdom
{k.judd, gammell, pnewman}@robots.ox.ac.uk.

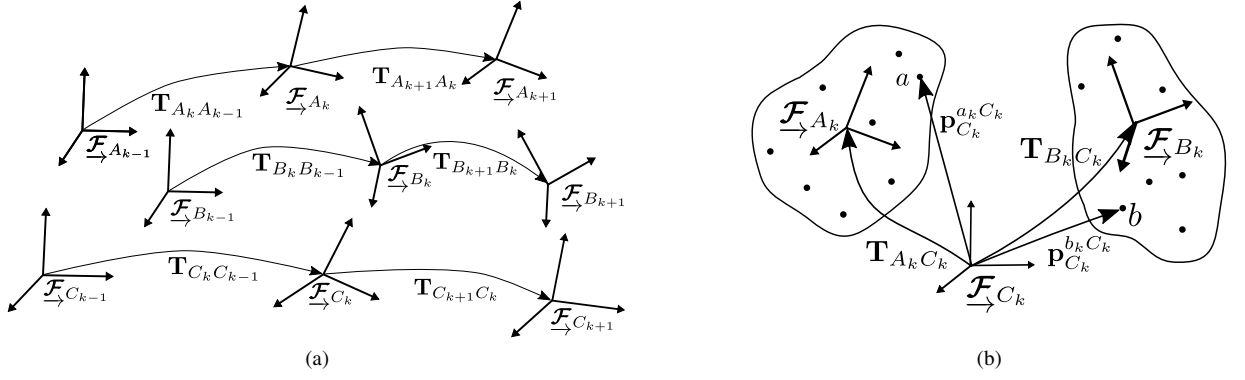


Fig. 2. Illustrations of the multimotion estimation problem showing the motion of frames through time, (a), and the relative point observations, (b). A series of two independent third-party motions, \mathcal{F}_A and \mathcal{F}_B , are observed by a moving camera, \mathcal{F}_C , through feature measurements on the objects, $p_{C_k}^{a_k}$ and $p_{C_k}^{b_k}$. Solving the problem requires simultaneously segmenting and estimating the set of measurements.

$T_\ell := (\mathbf{T}_{\ell_k, \ell_1})_{k=1 \dots K}$. Likewise, a sequence of observations of a point, j , by a moving camera, C , over multiple frames forms a tracklet, $p^j := (\mathbf{p}_{C_k}^{j_k})_{k=1 \dots K}$, where C_k refers to the observing camera frame at time k (Fig. 2b). Tracklets moving with a common trajectory can be grouped into bulk motions as $\mathcal{P}_\ell := \{p^j\}_{1 \dots N}$.

Motions estimated from these measurements are *egocentric*. They can be represented as *geocentric* motions after identifying one motion as the camera, T_C .

B. Flow Techniques

Optical flow [6], scene flow [7], and sparse scene flow [8] are approaches for finding the 2D or 3D velocity vector of pixels or feature points in a scene. These individual velocities are inherently translational and motions involving rotations (i.e., $SE(3)$ transforms) can only be estimated from segmentations of three or more velocities. In the presence of small rotations, these segmentations can be achieved using flow discontinuities [9] or the vector distance between velocities [8].

Larger rotations result in smoothly varying tangential velocities that provide no clear segmentations (Fig. 3a). To correctly estimate these motions, flow techniques must solve an equivalent segmentation and estimation problem posed in the space of velocities. In contrast to these flow techniques, MVO simultaneously segments and estimates full $SE(3)$ transforms of motions in the scene (Fig. 3b).

C. Tracking-by-Detection Methods

Appearance-based tracking techniques detect objects in images and then solve the association and motion estimation problems [4]. Kalman or particle filters are widely used to estimate the motion of the detected objects given a motion model but struggle to handle detection errors or occlusions [10].

Byeon et al. [11] propose an optimization framework for tracking multiple objects and estimating their trajectories from multiple static cameras by incorporating reconstruction and motion dynamics in their cost function. Zhang et al. [12] model tracking as a minimum-flow problem on a graph where nodes represent detections and edges represent transitions between frames.

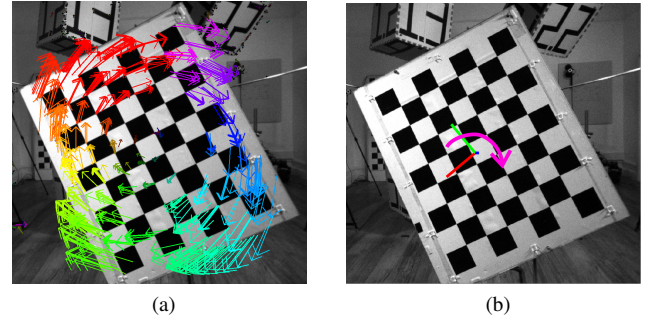


Fig. 3. Scene flow [8], (a), and MVO, (b), on a checkerboard rotating about the optical axis. Flow techniques inherently calculate individual translational velocities. Since each point on a rotating body has a different tangential velocity, flow techniques fail to segment the velocities into bulk motions. MVO estimates motions as $SE(3)$ transforms even in the presence of rotation. To estimate these motions, flow techniques must solve an equivalent segmentation and estimation problem in the space of velocities.

Object detectors are designed either for some specified class of objects or dynamically for some object of interest. These detectors are therefore specialized for a specific set of applications or are fragile to appearance changes and need to be refined over time [13].

Many of these approaches require either static or known camera motion and therefore need to incorporate separate egomotion estimators [14]. The object positions only exist in \mathbb{R}^3 and do not fully encapsulate the $SE(3)$ motions of the objects. Kundu et al. [15] extend egomotion estimation with MBSfM techniques similar to [14] to estimate the $SE(3)$ trajectories of the third-party motions in a scene, but they constrain all the motions to the horizontal plane.

Unlike these appearance-based techniques, MVO relies on low-level feature tracking. This means it can handle large changes in object appearance over time so long as a suitable number of features remain stable between each pair of frames. MVO also estimates the full, unconstrained $SE(3)$ motion within a scene, including the egomotion of the camera.

D. Subspace Methods

Subspace techniques cluster sparse feature points and their $SE(3)$ motions into lower-dimensional subspaces using

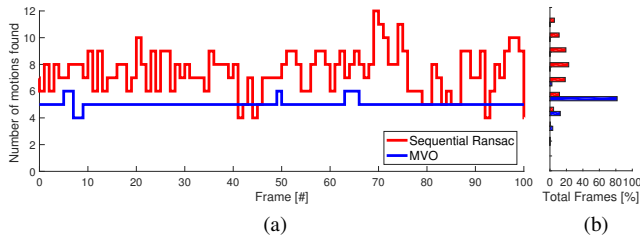


Fig. 4. A comparison of the number of models found by a sequential RANSAC (red) and MVO (blue) over 500 frames of a five-motion experiment (Sec. IV). A 100-frame section of the sequence is shown in (a) and the cumulative totals in (b) for the entire sequence. Without prior knowledge of the number of motions, sequential RANSAC greedily overfits to noise and finds an inconsistent number of models. Where RANSAC only found the correct number of models 24.3% of the time, MVO found the correct number 81.6% of the time.

the affine camera model. The affine model approximates the nonlinear perspective projection with a linear parallel projection. This simplifies the camera model but introduces severe projection errors in scenes with a wide field of view or a large depth of field [16].

Tomasi and Kanade [17] use the affine model and matrix factorization to decompose tracked image points into a motion and a shape matrix. Costeira and Kanade [5] extend this technique to multiple bodies where points may belong to different objects. This approach is inherently fragile as noise propagates through the factorization in complex ways [18]. It also requires points to be tracked for the entirety of the estimation window, which is difficult in complex scenes.

This formulation was extended by using an optimization framework to allow feature point dropouts [19] and by merging motions to mitigate the effect of noise [18, 20]. While these techniques are able to estimate full $SE(3)$ motions, they still depend upon an affine camera model, meaning they fail under any significant perspective effects.

These factorization techniques have also been extended to the perspective camera model by estimating depth in a preprocessing stage [21] or by applying geometric constraints [22] but still remain very sensitive to noise. In comparison, MVO can robustly estimate full $SE(3)$ trajectories using a perspective model while also handling the significant feature tracking failures characteristic of dynamic scenes.

E. Sampling Methods

Sampling methods estimate and fit models (e.g., motion trajectories) to a subset of the data before evaluating them across its entirety. RANSAC [23] is a popular framework to fit a model to data in the presence of noise. Points are sampled from the data to estimate a hypothesis model, which is then used to segment the data into inliers and outliers according to their fit. Hypotheses are repeatedly generated for some number of iterations and the model with the most inliers is selected as the segmentation and estimation.

Torr [24] extends the RANSAC framework to multiple models by finding the dominant model, removing those points that fit the model, and then recursively applying RANSAC to the remaining points. This recursive, sequential RANSAC framework is efficient at finding the dominant models in a scene, but

the ability to sample consistent models decreases as models are removed and the signal-to-noise ratio of the remaining points decreases. Sabzevari and Scaramuzza [25] apply geometric and kinematic constraints to reduce the required number of points to estimate a motion model and then realign point assignments to the best set of motion hypotheses in a separate step. They use the same matrix formulation as [21], meaning points must be successfully tracked through the entirety of the window, and their applications are limited by the constraints.

Other techniques [26, 27] use sampling methods to generate a large number of initial model hypotheses, realizing many of them would be redundant or poorly fit the data. Models are merged if their inlier sets are largely overlapping, and the models with the largest nonoverlapping inlier sets after merging are taken as the constituent scene motions.

These sampling methods are efficient but the probability of sampling inliers all from a single model decreases rapidly with the signal-to-noise ratio. Finding a motion in complex dynamic scenes is challenging because all other motions are outliers that decrease the signal-to-noise ratio and make it harder to find correct models. As a result, many of these sampling-based initializations struggle to find correct models.

Without prior knowledge of the number of models in the data, RANSAC tends to greedily overfit to noise and finds erroneous or incomplete models (Fig. 4). In contrast, MVO estimates all models simultaneously and requires no *a priori* knowledge of the number of models.

F. Energy Minimization Methods

Energy minimization approaches segment data into multiple labels simultaneously by reducing a cost function. In multimotion estimation, this cost is designed to encompass how well the estimated trajectories describe the data, e.g., reprojection or photoconsistency error, as well as encourage piecewise smoothness throughout the scene. Smoothness is enforced over a graph structure, usually either a dense Markov random field [28] or a sparse, feature-based graph [29].

Rother et al. [30] use a minimization framework to find a binary segmentation of the static background from a manually selected dynamic foreground object, but the approach can only segment a single object within a bounding box. PEARL [29] uses α -expansion and model refitting to iteratively estimate both models and point assignments in an expectation-maximization framework. The framework can be applied to motion segmentation by first sampling the data to estimate a large number of motion models (similarly to [26, 27]) and then refining the models and segmentation with PEARL. Roussos et al. [31] use PEARL as part of a dense expectation-maximization pipeline that estimates depth, motion, and segmentation from monocular images in an offline manner. Optical flow is used to initialize the depth maps and the approach is crucially dependent on this initialization.

Rünz and Agapito [32] use a similar optimization framework to segment dense RGB-D camera observations. Using this segmentation they create multiple 3D object models whose $SE(3)$ motions are then tracked, establishing new motions online. This approach requires an initialization

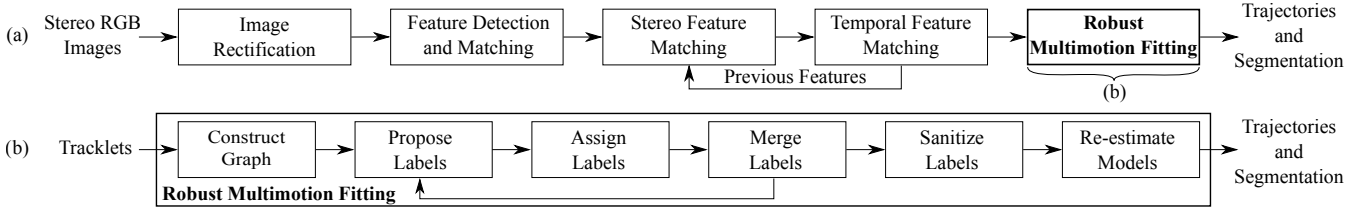


Fig. 5. An illustration of the stereo multimotion visual odometry (MVO) pipeline, (a), including details of the multimotion fitting component, (b). The pipeline operates on RGB stereo image pairs and estimates the segmentation and trajectories of all motions in the scene. It extends the standard stereo visual odometry pipeline by replacing the egomotion estimator with a multimotion estimator that operates on the tracklets produced by the previous stages. The multimotion-fitting pipeline builds a neighborhood graph based on the distances between pairs of points throughout the estimation batch. The pipeline iteratively splits and estimates new labels using the neighborhood graph, performs label-assignment using CORAL [3], and merges labels that can be considered redundant until convergence. Finally, the label set is sanitized and a full-batch estimation step produces the final segmentation and trajectories.

phase that seeds the structure and segmentation of the background of the scene.

These approaches are capable of estimating full $SE(3)$ motions but are dependent upon comprehensive initialization, often using RANSAC, as they are designed around refining existing labels. In comparison, MVO iteratively proposes and refines labels, allowing it to find motions that may be difficult to initially segment with sampling methods alone.

G. Multimotion Visual Odometry (MVO)

To the best of our knowledge, this paper introduces the first online approach capable of directly estimating full $SE(3)$ trajectories for every motion in a complex, dynamic scene from a stereo/RGB-D camera using only a rigid-body assumption. The approach uses multilabeling and estimation techniques to model the motions of tracklet features over multiple frames. The hypothesis trajectories are then applied to individual tracklets by CORAL [3], a convex optimization approach to the multilabeling problem. These hypotheses are iteratively improved through splitting and merging of the models, unlike other labeling approaches that initially sample them from the scene. The full $SE(3)$ trajectory of *each* motion is finally estimated using traditional VO batch estimation techniques. This approach is evaluated on a dataset containing ground-truth trajectories for all motions in the scene.

III. METHODOLOGY

The stereo MVO pipeline (Fig. 5a) extends the traditional stereo VO pipeline to *multimodel* segmentation and estimation. The incoming RGB stereo images are first rectified and undistorted according to known camera extrinsics and intrinsics. Salient image points are detected and matched across left and right images in each stereo pair and across temporally consecutive pairs of stereo frames. These stereo- and temporally-matched feature points are back-projected into the 3D space using the camera intrinsics, forming world- and image-space tracklet histories for each feature point. This set of tracklets, $\mathcal{P} := \{p\}$, becomes the input to the multimotion segmentation and estimation engine (Fig. 5b). These tracklets could alternatively be found by associating observations from other 3D sensors (e.g., RGB-D cameras) over time.

The multimotion engine segments tracklets by their observed motion, which is a combination of camera and object motions. In the absence of *a priori* information about the

scene, each group of tracklets is used to estimate a camera egomotion by assuming those tracklets belong to a static object. These camera egomotion *hypotheses* can later be converted into estimates of the camera and object motions by identifying the static part of the scene (e.g., as in VO).

The segmentation and estimation are posed as a multilabeling problem where a label, ℓ , represents the egomotion hypothesis, ℓT_C , calculated from a group of tracklets, $\mathcal{P}_\ell \subseteq \mathcal{P}$. These labels are assigned by minimizing a cost function over a graph of all observed tracklets (Sec. III-A). New labels are proposed for each disconnected component of a label's subgraph through a multiframe RANSAC procedure (Sec. III-B). Motion labels are assigned to minimize the reprojection residual of the associated trajectory and maximize the label smoothness in the graph (Sec. III-C). An outlier label, \emptyset , is assigned to points whose motions are not well explained by any other label. Redundant and oversegmented labels are then merged (Sec. III-D).

The algorithm iterates this process until label convergence. The final labels are then sanitized and any remaining outliers are rejected (Sec. III-E) before a final, full-batch estimation of each label (Sec. III-F). Egocentric or geocentric trajectories are found by selecting a label to represent the motion of the camera (Sec. III-G).

A. Graph Construction

The rigid-body assumption is approximated through a geometric neighborhood graph, \mathcal{N} . Each vertex of the graph represents an observed tracklet and is connected to its k -nearest-neighbors. The distance between two vertices is defined as the maximum distance in image space between those image tracklets over the entire batch,

$$d(p^i, p^j) := \max_{k=1 \dots K} \left\| \mathbf{s} \left(\mathbf{p}_{C_k}^{i_k C_k} \right) - \mathbf{s} \left(\mathbf{p}_{C_k}^{j_k C_k} \right) \right\|_2,$$

where $\mathbf{s}(\cdot)$ applies the nonlinear perspective camera projection. This allows for edges between features that are consistently close while not connecting features that are ever far apart or that never coexist in a frame. This connectivity forms the basis for label generation and assignment.

B. Label Proposal

The label set, \mathcal{L} , must dynamically grow and adapt to correctly converge in a given scene. To accomplish this, new labels are generated by splitting label support groups

whenever their tracklets' motions could more accurately be explained by multiple trajectories. A potential new label, ℓ' , is generated for each fully-disjoint component of the subgraph defined by the label's support, $\mathcal{N}_{\ell'} \subseteq \mathcal{N}_\ell \subseteq \mathcal{N}$. This ensures a level of spatial smoothness while allowing new labels to be proposed from large label supports comprised of tracklets from spatially or temporally distinct motions in the scene.

The new label proposals are generated by computing both the dominant motion of the given points and the segmentation between inliers and outliers of that motion. This single-motion segmentation and estimation problem is solved by applying RANSAC in a frame-to-frame fashion, similar to standard VO systems.

Three tracklets are sampled from those visible in the current, k , and previous, $k-1$, frames to estimate the $SE(3)$ transform between the two frames ${}^{\ell'}\mathbf{T}_{C_k, C_{k-1}}$. The proposed transform is evaluated according to how many tracklet reprojection residuals,

$$e_k(p^j, {}^{\ell'}T_C) := \left\| \mathbf{s} \left(\mathbf{p}_{C_k}^{j_k C_k} \right) - \mathbf{s} \left({}^{\ell'}\mathbf{T}_{C_k C_{k-1}} \mathbf{p}_{C_{k-1}}^{j_{k-1} C_{k-1}} \right) \right\|_2, \quad (1)$$

are within a given threshold error, e_{th} . This process is repeated many times and the transform with the largest inlier set is appended to the proposed trajectory hypothesis, ${}^{\ell'}T_C$.

Any tracklets found to be outliers of the newly estimated models are appended to the outlier label, \mathcal{O} . New labels are generated from the outlier label last.

C. Label Assignment

Each tracklet, $p \in \mathcal{P}$, is assigned a label, $\ell \in \mathcal{L}$, to minimize the energy functional,

$$E(\mathcal{L}) := \underbrace{\sum_{p \in \mathcal{P}} \rho(p, \ell(p))}_{\text{Residual}} + \underbrace{\lambda \sum_{(p,q) \in \mathcal{N}} \omega_{pq} V(p, q)}_{\text{Smoothness}} + \underbrace{\sum_{\ell \in \mathcal{L}} \gamma_\ell \psi_\ell}_{\text{Complexity}}, \quad (2)$$

where $\ell(p)$ gives the label currently assigned to p . The energy functional combines the residual error, the label smoothness, and the label complexity term, using a user-selected proportionality parameter, λ .

a) Residual: The residual term penalizes labels that poorly describe the observed data. It is defined as the sum of the residual errors of applying the label trajectories to tracklets. The residual for each point-label pair is defined as

$$\rho(p^j, \ell) := \max_{k=1 \dots K} e_k(p^j, {}^{\ell}T_C),$$

where e_k as defined in (1).

b) Smoothness: The smoothness term penalizes neighboring tracklets that do not share the same label by an edge cost, ω_{pq} . This encourages a piecewise-smooth solution. It is a weighted sum of all edges penalized according to

$$V(p, q) := \begin{cases} 1 & \ell(p) \neq \ell(q) \\ 0 & \text{otherwise} \end{cases}.$$

c) Complexity: The complexity term encourages a compact solution by penalizing the use of many labels. It is the sum of the per-label cost, γ_ℓ , of each label with non-empty support set according to the function,

$$\psi_\ell := \begin{cases} 1 & |\mathcal{P}_\ell| > 0 \\ 0 & \text{otherwise} \end{cases}.$$

d) Outliers: The outlier label, \mathcal{O} , is designed to be attractive to all points whose motions are not well explained by existing labels. The residual energy of the outlier label decays exponentially with that of the best-fitting label,

$$\rho(p^j, \mathcal{O}) := \alpha \exp \left(-\frac{\min_{\ell \in \mathcal{L}} \rho(p^j, \ell)}{\beta} \right),$$

where α and β are tuning parameters. Points that are well-explained by an extant label will have high outlier data cost. The label cost, $\gamma_\mathcal{O}$, for the outlier label is zero as outliers are assumed to always exist.

Given the current label set, α -expansion (e.g., PEARL [29]) or convex optimization (e.g., CORAL [3]) assigns a label to each tracklet to minimize the residual and smoothness energies of (2). The minimization can result in an oversegmentation due to outliers and poorly estimated intermediate trajectories. Model merging is therefore used to improve the motion estimation.

D. Label Merging

Two labels, ℓ and ℓ' , may be merged if relabeling all $\mathcal{P}_{\ell'}$ as ℓ would decrease the total energy of (2). This occurs when the increase in residual error due to reduced overfitting is less than the cost of using the label, $\gamma_{\ell'}$, and any change in smoothness.

Only the periods during which the two labels' supports overlap are considered because there is no cost for applying a new label to portions of the batch in which the tracklets do not exist. When more than one merge would reduce the total energy, the one that results in the greatest decrease in cost is chosen. Merging continues until no more merges would reduce (2). The outlier label, \mathcal{O} , is excluded from merging.

The merging stage only considers label pairs with tracklets adjacent in \mathcal{N} , i.e., those where \mathcal{N}_ℓ is connected to $\mathcal{N}_{\ell'}$. If they are disconnected, merging the two supports would be undone by the splitting routine (Sec. III-B). If the two support sets are connected then the new label will persist until the next labeling stage.

The algorithm iterates the label splitting, assignment, and merging (Secs. III-B to III-D) until the labels converge or a maximum number of iterations have been reached. The final label set is then sanitized (Sec. III-E) before being used to estimate the final trajectory hypotheses (Sec. III-F).

E. Label Sanitization

The final labels are sanitized to refine the segmentation output and remove noisy tracklets before the final model estimation. A merging step first combines any redundant labels regardless of graph connectivity as there is no subsequent splitting stage. After merging, any label with fewer than a

minimum number of support tracklets or that exists for fewer than a minimum number of frames is merged with \mathcal{O} . Likewise, tracklets whose residual error is greater than a threshold, e_{th} , are relabeled as outliers. This provides a consistent set of tracklets for the batch estimation of each motion.

F. Final Model Estimation

For each label, an egomotion hypothesis, ${}^\ell T_C$, is estimated to explain the motion of the tracklets, \mathcal{P}_ℓ , using bundle adjustment. This paper follows the single-motion approach described by Barfoot [33] to estimate the trajectory of each label in an egocentric frame.

The system state, \mathbf{x} , of each label is defined to include both the estimated pose transforms, ${}^\ell T := ({}^\ell \mathbf{T}_{C_k C_1})_{k=2 \dots K}$, and the landmark points, $\{\mathbf{p}_{C_1}^{j_1 C_1}\}_{j=1 \dots |\mathcal{P}_\ell|}$. The state $\mathbf{x}_{jk} := \{{}^\ell \mathbf{T}_{C_k C_1}, \mathbf{p}_{C_1}^{j_1 C_1}\}$ is defined for each pair of transforms and points belonging to label ℓ .

Each observation, \mathbf{y}_{jk} , of point \mathbf{p}^j at pose ${}^\ell \mathbf{T}_{C_k C_1}$ is modeled as

$$\begin{aligned} \mathbf{y}_{jk} &:= \mathbf{g}(\mathbf{x}_{jk}) + \mathbf{n}_{jk} = \mathbf{s}(\mathbf{z}(\mathbf{x}_{jk})) + \mathbf{n}_{jk} \\ &= \mathbf{s}({}^\ell \mathbf{T}_{C_k C_1} \mathbf{p}_{C_1}^{j_1 C_1}) + \mathbf{n}_{jk}. \end{aligned}$$

The measurement model, $\mathbf{g}(\cdot)$, encompasses both the motion model, $\mathbf{z}(\cdot)$, which applies $SE(3)$ transforms to observed points, and the sensor model, $\mathbf{s}(\cdot)$, derived from the perspective camera model. The model assumes additive Gaussian noise, \mathbf{n}_{jk} , with zero mean and covariance \mathbf{R}_{jk} . The least-squares cost function is defined as the difference between the measurement model and the observations,

$$J := \frac{1}{2} \sum_{jk} \mathbf{e}_{y,jk}(\mathbf{x})^T \mathbf{R}_{jk}^{-1} \mathbf{e}_{y,jk}(\mathbf{x}),$$

where,

$$\mathbf{e}_{y,jk}(\mathbf{x}) := \mathbf{y}_{jk} - \mathbf{s}({}^\ell \mathbf{T}_{C_k C_1} \mathbf{p}_{C_1}^{j_1 C_1}).$$

This cost is linearized about an operating point, \mathbf{x}_{op} , and then minimized using Gauss-Newton. The operating point is perturbed according to the transform perturbations, $\{\epsilon_k \in \mathbb{R}^6\}$, and landmark perturbations, $\{\zeta_j \in \mathbb{R}^3\}$, which together form the full state perturbation, $\delta \mathbf{x}$. An indicator matrix \mathbf{P}_{jk} is defined such that $\delta \mathbf{x}_{jk} = \mathbf{P}_{jk} \delta \mathbf{x}$. See [33] for more detail.

The error function is linearized using \mathbf{G}_{jk} , the Jacobian of the measurement function, $\mathbf{g}(\cdot)$,

$$\begin{aligned} \mathbf{G}_{jk} &:= \mathbf{S}_{jk} \mathbf{Z}_{jk}, \\ \mathbf{S}_{jk} &= \frac{\partial \mathbf{s}}{\partial \mathbf{z}} \bigg|_{\mathbf{z}(\mathbf{x}_{\text{op}}, jk)}, \\ \mathbf{Z}_{jk} &= \left[({}^\ell \mathbf{T}_{\text{op}, C_k C_1} \mathbf{p}_{\text{op}, C_1}^{j_1 C_1})^\odot \quad {}^\ell \mathbf{T}_{\text{op}, C_k C_1} \mathbf{D} \right], \\ \mathbf{D} &= \begin{bmatrix} \mathbf{1} \\ \mathbf{0}^T \end{bmatrix}, \end{aligned}$$

where the matrix operator $(\cdot)^\odot$ is defined in [33]. The cost function can then be linearized using

$$\begin{aligned} J &\approx J(\mathbf{x}_{\text{op}}) - \mathbf{b}^T \delta \mathbf{x}_{jk} + \frac{1}{2} \delta \mathbf{x}_{jk}^T \mathbf{A} \delta \mathbf{x}_{jk}, \\ \mathbf{b} &= \sum_{jk} \mathbf{P}_{jk}^T \mathbf{G}_{jk}^T \mathbf{R}_{jk}^{-1} \mathbf{e}_{y,jk}(\mathbf{x}_{\text{op}}), \\ \mathbf{A} &= \sum_{jk} \mathbf{P}_{jk}^T \mathbf{G}_{jk}^T \mathbf{R}_{jk}^{-1} \mathbf{G}_{jk} \mathbf{P}_{jk}. \end{aligned}$$

The optimal perturbation, $\delta \mathbf{x}^*$, for minimizing the cost function, J , is the solution to $\mathbf{A} \delta \mathbf{x}^* = \mathbf{b}$. Each element of the state is then updated according to

$$\begin{aligned} {}^\ell \mathbf{T}_{\text{op}, C_k C_1} &\leftarrow \exp(\epsilon_k^{*\wedge}) {}^\ell \mathbf{T}_{\text{op}, C_k C_1} \\ \mathbf{p}_{\text{op}, C_1}^{j_1 C_1} &\leftarrow \mathbf{p}_{\text{op}, C_1}^{j_1 C_1} + \mathbf{D} \zeta_j^*, \end{aligned}$$

where the vector operator $(\cdot)^\wedge$ is defined in [33]. The cost function is then relinearized about the updated operating point and the process iterates until convergence.

G. Egocentric and Geocentric Trajectories

a) *Egocentric*: Egocentric motions are expressed in the moving camera frame, \mathcal{F}_{C_k} . The egocentric motion of the camera is identity by definition and the egocentric motions of the scene are given by

$$\forall \ell \in \mathcal{L}, {}^{\text{ego}} \mathbf{T}_{\ell K \ell_1} := {}^\ell \mathbf{T}_{C_K C_1}^{-1}.$$

One of these motions is the egocentric motion of the static world *caused* by the camera motion.

b) *Geocentric*: Geocentric motions are expressed in some earth-attached frame. The geocentric motion of the camera is given by the hypothesis motion estimated from the static background,

$$\mathbf{T}_{C_K C_1} := \mathbf{T}_C \big|_{\ell=\ell_{\text{static}}},$$

where the static label may be selected by heuristics as in VO (e.g., label support size).

The geocentric motions of the rest of the scene are given by

$$\forall \ell \in \mathcal{L} \setminus \ell_{\text{static}}, \mathbf{T}_{\ell_k \ell_1} = \mathbf{F}_{\ell_k \ell_1} \mathbf{T}_{\ell_1 C_1} {}^\ell \mathbf{T}_{C_K C_1}^{-1} \mathbf{T}_{C_K C_1} \mathbf{T}_{\ell_1 C_1}^{-1},$$

where $\mathbf{F}_{\ell_k \ell_1}$ is the object deformation matrix and is assumed to be identity, (i.e., rigid body). The initial transform,

$$\mathbf{T}_{\ell_1 C_1} = \begin{bmatrix} \mathbf{C}_{\ell_1 C_1} & \mathbf{r}_{\ell_1}^{C_1 \ell_1} \\ \mathbf{0}^T & 1 \end{bmatrix},$$

relates the camera to the center of motion of each object, $\mathbf{r}_{\ell_1}^{C_1 \ell_1}$. The object center is calculated from the centroid of all points, \mathcal{P}_ℓ , projected into the first observed frame,

$$\mathbf{r}_{\ell_1}^{C_1 \ell_1} = -\frac{1}{|\mathcal{P}_\ell|} \mathbf{C}_{\ell_1 C_1} \sum_{j=1}^{|\mathcal{P}_\ell|} {}^\ell \mathbf{T}_{C_{t_j} C_1}^{-1} \mathbf{p}_{C_{t_j}}^{j_{t_j} C_{t_j}},$$

where t_j is the first frame where p^j is observed, and $\mathbf{C}_{\ell_1 C_1}$ is arbitrary and assumed to be identity. This averaging allows the centroid estimate to adjust as new points are observed due to rotation or occlusion.

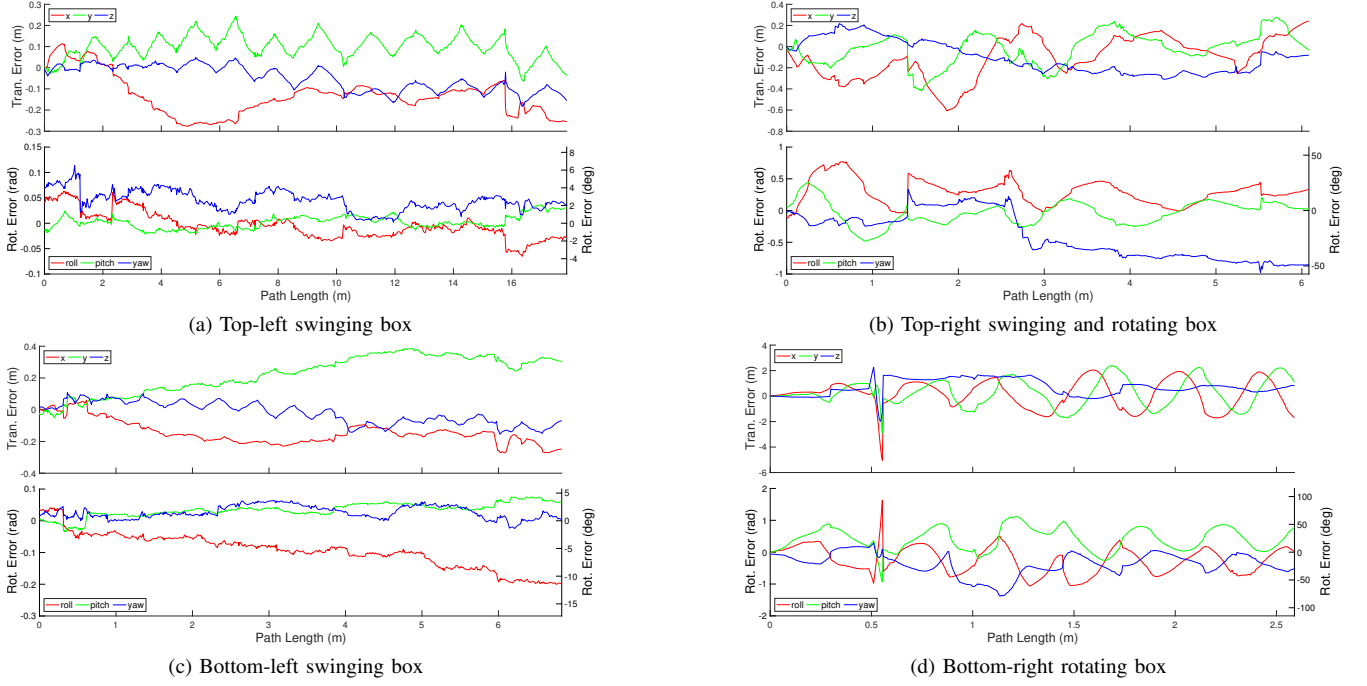


Fig. 6. The translational and rotational errors for the estimated motion of each object over the length of the estimation window as compared to ground-truth trajectory data. Errors are reported in an arbitrary geocentric frame with the z-axis up, and the arbitrary x- and y-axes.

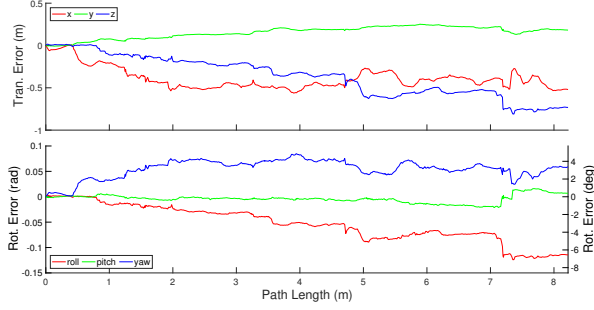


Fig. 7. The translational and rotational errors for the egomotion of the camera over its path compared to ground-truth trajectory data. Errors are reported in the egocentric camera frame with the z-axis along the optical axis, and the y-axis down.

IV. EXPERIMENTS AND RESULTS

The accuracy of the MVO algorithm is evaluated on real-world data collected using a Bumblebee XB3 stereo camera and a Vicon motion capture system. Unlike existing multimotion datasets designed for egomotion or segmentation (e.g., [34, 35]), this new dataset contains ground truth for the *entire* scene which consisted of a moving camera observing four moving blocks (Fig. 1).

The results (Figs. 6 and 7) were produced from a 500-frame image sequence. Estimation was performed as a 48-frame sliding window, with 5 neighbors for each point in the graph, 1000 RANSAC iterations per new label, $e_{th} = 2$, $\alpha = 100$, $\beta = 5$, $\psi_\ell = 1000$, $\lambda = 1$, and a minimum model size and length of 10 points and 3 frames, respectively. Feature detection and matching were performed using LIBVISO2 [36] and the Gauss-Newton minimization was performed with Ceres [37].

The transforms between the Vicon frames and our estimated

frames are arbitrary, so the first 20 frames of the estimates are used to calibrate this transform. All errors are reported for geocentric trajectory estimates.

The camera egomotion (Fig. 7) exhibits a maximum total drift of 0.93 m, 11.7% of total path length, and a maximum rotational error of -6.82° , 0.16° , and 3.13° in roll-pitch-yaw, respectively. This error is reasonable compared to the level of drift in other model-free, camera-only VO systems [34].

The motion estimates of the bodies varied with their motion. Rotating bodies often exhibited jumps in their motion which resulted in larger errors than the swinging bodies and egomotion. The error of each block includes that of the egomotion estimate, and the maximum translational and rotational errors for each block are 0.36 m, 3.09° , 0.16° , and 6.51° for the top-left block; 0.64 m, 25.26° , 1.46° , and -55.83° for the top-right block; 0.45 m, -11.35° , 4.08° , and 0.53° for the bottom-left block; and 5.94 m, 93.56° , -53.75° , and 5.77° for the bottom-right block (Fig. 6).

The jumps in the trajectories occur when the final estimation stage (Sec. III-F) fails to converge. This is often due to poor feature distribution, especially for rotating bodies which are self-occluding. These discontinuities, coupled with the dynamic camera motion, cause errors in the trajectory estimate.

V. DISCUSSION

MVO consistently segmented the motions of the camera and the four independent objects and estimated their motions with varying levels of accuracy. Specifically, highly rotating motions (e.g., the bottom-right box) often resulted in estimates with large, discrete motion jumps despite the correct segmentation. We are currently investigating whether this is

the result of the sparse features used to estimate the motion or our batch estimation pipeline itself.

The distribution of features also influences the estimate of the center of the motion. It is difficult to infer the structure of an object beyond the surfaces that are observed in the batch without *a priori* knowledge of the object's shape. This means the centroid of the observed feature points for a label can often be a bad estimate of the the label's true center of motion.

Feature distribution is an important factor in the performance of any sparse approach [38]. This problem is only exacerbated for dynamic objects that take up a small portion of the scene. Additionally, the appearance of these objects is often more volatile making feature association even more difficult. The algorithm is dependent on the accuracy of the input tracklet set and cannot estimate motions for which there are insufficient features. Feature dropouts and lack of tracklets are therefore significant challenges for this type of pipeline, and the development of more robust feature detection and matching pipelines is an ongoing area of research.

VI. CONCLUSION

This paper extends the classic VO pipeline to address the multimotion estimation problem. The multimotion visual odometry (MVO) pipeline segments and estimates *all* rigid motions in a scene. It does so by using feature-tracking, sparse graph segmentation, and multiframe batch motion estimation such that it avoids many of the limitations of other multimotion estimation approaches.

We evaluated MVO on a multimotion dataset with ground-truth trajectories for all motions in the scene. Its estimation accuracy is comparable to similarly defined egomotion-only VO systems while also exhibiting similar limitations. We are actively exploring the application benefits of continuous-time state estimation and continuous labels, as well as implementing the pipeline such that it can be used in real-time.

ACKNOWLEDGMENT

We would like to thank Paul Amayo for his insightful conversations on convex optimization and multilabeling, and for his implementation of CORAL.

REFERENCES

- [1] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover," Ph.D. dissertation, Stanford, CA, USA, 1980.
- [2] L. Matthies and S. Shafer, "Error modeling in stereo navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 239–248, 1987.
- [3] P. Amayo, P. Piniés, L. M. Paz, and P. Newman, "Geometric Multi-Model Fitting with a Convex Relaxation Algorithm," in *CVPR*, 2018.
- [4] A. Milan, L. Leal-Taixé, I. D. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," 2016, arXiv: 1603.00831 [cs.CV].
- [5] J. P. Costeira and T. Kanade, "A multibody factorization method for independently moving objects," *IJCV*, vol. 29, no. 3, pp. 159–179, 1998.
- [6] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.
- [7] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," in *ICCV*, vol. 2, 1999, pp. 722–729.
- [8] P. Lenz, J. Ziegler, A. Geiger, and M. Roser, "Sparse scene flow segmentation for moving object detection in urban environments," in *IV*, 2011, pp. 926–932.
- [9] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *CVPR*, 2015, pp. 3061–3070.
- [10] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.
- [11] M. Byeon, H. Yoo, K. Kim, S. Oh, and J. Y. Choi, "Unified optimization framework for localization and tracking of multiple targets with multiple cameras," *CVIU*, vol. 166, pp. 51 – 65, 2018.
- [12] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *CVPR*, 2008, pp. 1–8.
- [13] Z. Kalal, K. Mikolajczyk, J. Matas *et al.*, "Tracking-learning-detection," *PAMI*, vol. 34, no. 7, p. 1409, 2012.
- [14] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *IJRR*, vol. 26, no. 9, pp. 889–916, 2007.
- [15] A. Kundu, K. M. Krishna, and C. V. Jawahar, "Realtime multibody visual slam with a smoothly moving monocular camera," in *ICCV*, 2011, pp. 2080–2087.
- [16] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [17] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *IJCV*, vol. 9, no. 2, pp. 137–154, 1992.
- [18] K. Kanatani, "Motion segmentation by subspace separation and model selection," in *ICCV*, vol. 2, 2001, pp. 586–591.
- [19] R. Vidal and R. Hartley, "Motion segmentation with missing data using power factorization and gpca," in *CVPR*, 2004, pp. 310–316.
- [20] Y. Wu, Z. Zhang, T. S. Huang, and J. Y. Lin, "Multibody grouping via orthogonal subspace decomposition," in *CVPR*, vol. 2, 2001, pp. 252–257.
- [21] T. Li, V. Kallem, D. Singaraju, and R. Vidal, "Projective factorization of multiple rigid-body motions," in *CVPR*, 2007, pp. 1–6.
- [22] R. Vidal and S. Sastry, "Optimal segmentation of dynamic scenes from two perspective views," in *CVPR*, vol. 2, 2003, pp. 281–286.
- [23] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [24] P. H. Torr, "Geometric motion segmentation and model selection," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 356, no. 1740, pp. 1321–1340, 1998.
- [25] R. Sabzevari and D. Scaramuzza, "Multi-body motion estimation from monocular vehicle-mounted cameras," *T-RO*, vol. 32, no. 3, pp. 638–651, 2016.
- [26] K. Schindler, U. James, and H. Wang, "Perspective n-view multibody structure-and-motion through model selection," in *ECCV*. Springer, 2006, pp. 606–619.
- [27] K. E. Ozden, K. Schindler, and L. V. Gool, "Multibody structure-from-motion in practice," *PAMI*, vol. 32, no. 6, pp. 1134–1141, 2010.
- [28] C. Nieuwenhuis, E. Töppe, and D. Cremers, "A survey and comparison of discrete and continuous multi-label optimization approaches for the potts model," *IJCV*, vol. 104, no. 3, pp. 223–240, 2013.
- [29] H. Isack and Y. Boykov, "Energy-based geometric multi-model fitting," *IJCV*, vol. 97, no. 2, pp. 123–147, 2012.
- [30] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in *ACM Transactions on Graphics*, vol. 23, no. 3, 2004, pp. 309–314.
- [31] A. Roussos, C. Russell, R. Garg, and L. Agapito, "Dense multibody motion estimation and reconstruction from a handheld camera," in *ISMAR*, 2012, pp. 31–40.
- [32] M. Rünz and L. Agapito, "Co-fusion: Real-time segmentation, tracking and fusion of multiple objects," in *ICRA*, 2017, pp. 4471–4478.
- [33] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.
- [34] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012, pp. 3354–3361.
- [35] R. Tron and R. Vidal, "A benchmark for the comparison of 3-d motion segmentation algorithms," in *CVPR*, 2007, pp. 1–8.
- [36] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *IV*, 2011, pp. 963–968.
- [37] S. Agarwal, K. Mierle, and Others, "Ceres Solver."
- [38] S. Farboud-Sheshdeh, T. D. Barfoot, and R. H. Kwong, "Towards estimating bias in stereo visual odometry," in *CRV*, 2014, pp. 8–15.