2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Robust Ego and Object 6-DoF Motion Estimation and Tracking

Jun Zhang[1] Mina Henein[1] Robert Mahony[1] Viorela Ila[2]

***Abstract*— The problem of tracking self-motion as well as motion of objects in the scene using information from a camera is known as multi-body visual odometry and is a challenging task. This paper proposes a robust solution to achieve accurate estimation and consistent track-ability for dynamic multi-body visual odometry. A compact and effective framework is proposed leveraging recent advances in semantic instance-level segmentation and accurate optical flow estimation. A novel formulation, jointly optimizing SE(3) motion and optical flow is introduced that improves the quality of the tracked points and the motion estimation accuracy. The proposed approach is evaluated on the virtual KITTI Dataset and tested on the real KITTI Dataset, demonstrating its applicability to autonomous driving applications. For the benefit of the community, we make the source code public[†].**

I. Introduction

Visual odometry (VO) has been a popular solution for robot navigation in the past decade due to its low-cost and widely applicable properties. Studies in the literature have illustrated that VO can provide accurate estimation of a camera trajectory in largely static environment, with relative position error ranging from 0.1% to 2% [1]. However, the deployment of robotic systems in our daily lives requires systems to work in significantly more complex, dynamic environments. Visual navigation in non-static environments becomes challenging because the dynamic parts in the scene violate the motion model of camera. If moving parts of a scene dominate the static scene, off-the-shelf visual odometry systems either fail completely or return poor quality trajectory estimation. Earlier solutions proposed to directly remove the dynamic information via robust estimation [2], [3], however, we believe that this information is valuable if it is properly used. In most scenarios, the dynamics corresponds to a finite number of individual objects that are rigid or piecewise rigid, and their motions can be tracked and estimated in the same way as the ego-motion. Accurate object motion estimation and tracking becomes highly relevant in many applications, such as collision avoidance in autonomous driving and robotic systems, visual surveillance and augmented reality.

In this paper, we propose a novel multi-body visual odometry pipeline that address the problem of tracking both ego and object motion in dynamic outdoor scenes. The proposed pipeline leverages instance-level object segmentation algorithms [4] to robustly separate the scene into static background and multiple dynamic objects. Recent advances in optical flow estimation [5], [6] are exploited to maintain enough tracking points on each object to accurately estimate motion. With this data, we propose a new technique that jointly refines the initial optical flow and estimates full 6-DoF motion of both the camera and objects in the scene. We construct a fully-integrated system that is able to robustly estimate and track self and object motions utilizing only visual sensors (stereo/RGB-D). To the best of our knowledge, our work is the first to conduct an extensive evaluation of accuracy and robustness of ego and object 6-DoF motion estimation and tracking, and demonstrates the feasibility on real-world outdoor datasets.

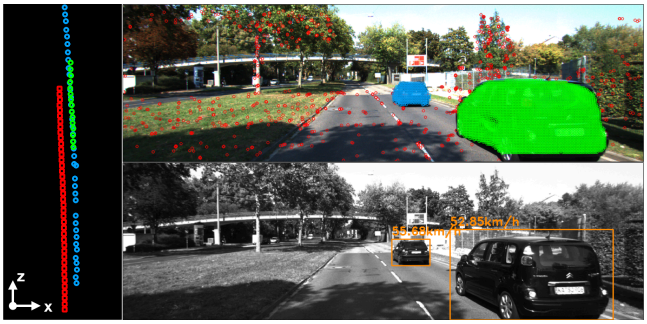

Fig. 1. Results of our proposed system on KITTI sequence 03. Camera and object trajectory (left). Detected points on background and object body (upper-right). Estimated object speed (bottom-right).

In the following, after Sec. II on related work, we introduce the methodology of our proposed algorithm in Sec. III, then describe the implementation of proposed pipeline in Sec. IV. Experimental results are documented in Sec. V.

II. Related Work

Visual odometry/SLAM for dynamic environments has been actively studied in the past few years, as described in a recent survey [7]. Earlier approaches detected non-static object in the scene and removed them from the estimation data. For instance, [3] uses dense scene flow for dynamic objects detection, and obtains improved localization and mapping results by removing erroneous measurements on dynamic objects from estimation. The authors in [2] propose an online keyframe update that reliably detects changed features by projecting them from keyframes to current frame for appearance and structure comparison, and discards them if necessary.

Meanwhile, researchers have started to incorporate dynamic information into camera pose estimation. A multi-camera SLAM system is proposed in [8], that is able to track multiple cameras, as well as to reconstruct the 3D positions of both static background and moving foreground

[1]Jun Zhang, Mina Henein and Robert Mahony are with the Australian National University (ANU), 0020 Canberra, Australia. {jun.zhang2,mina.henein,robert.mahony}@anu.edu.au

[2]Viorela Ila is with the University of Sydney (USyd), 2006 Sydney, Australia. viorela.ila@sydney.edu.au

[†]https://github.com/halajun/multimot_track

points. The idea is that points on moving objects give information about relative poses between different cameras at the same time step. Therefore, static and dynamic points are used together to decide all camera poses simultaneously. Kundu [9] proposed to detect and segment motion using efficient geometric constraints, then reconstruct the motion of dynamic objects with a bearing only tracking. Similarly, a multi-body visual SLAM framework is introduced in [10], which makes use of sparse scene flow to segment moving objects, then estimate the poses of camera as well as moving objects, respectively. Poses are formulated as a factor graph incorporating with constraints to reach a final optimization result.

Lately, the problem of object motion estimation and tracking is receiving increased attention in the robotics and computer vision community. Dewan [11] presents a model-free method for detecting and tracking moving objects in 3D LiDAR scans by a moving sensor. The method sequentially estimates motion models using RANSAC [12], then segments and tracks multiple objects based on the models by a proposed Bayesian approach. Results of sensor/objects speed error are illustrated to prove its effectiveness. In [13], the authors address the problem of simultaneous estimation of ego and third-party motions in complex dynamic scenes using cameras. They apply multi-model fitting techniques into a visual odometry pipeline to estimate all rigid motions within a scene. Promising results of SE(3) motions have been shown on multiple moving cubes dataset for indoor scenes.

III. Methodology

Our setup comprises a depth camera (stereo or RGB-D) moving in a dynamic environment. Let ${}^k\boldsymbol{P} = \{{}^k\boldsymbol{p}^i \in \mathbb{R}^3\}$ be a set of projected points into the image frame k, where ${}^k\boldsymbol{p}^i = [u^i,\ v^i\ , 1]^\top$ represents the point location in homogeneous coordinates. The points are either part of the static background ${}^k\boldsymbol{P}_s \subseteq {}^k\boldsymbol{P}$ or moving object ${}^k\boldsymbol{P}_o \subset {}^k\boldsymbol{P}$.

Assuming that a depth map ${}^k\boldsymbol{D} = \{{}^kd^i \in \mathbb{R}\}$ of frame k is provided, where ${}^kd^i$ is the corresponding depth for each point ${}^k\boldsymbol{p}^i \in {}^k\boldsymbol{P}$, the 3D point ${}^k\boldsymbol{m}^i \in \mathbb{R}^4$ of ${}^k\boldsymbol{p}^i$ can be obtained via back-projection:

$$ {}^k\boldsymbol{m}^i = \begin{bmatrix} m_x^i \\ m_y^i \\ m_z^i \\ 1 \end{bmatrix} = \pi^{-1}({}^k\boldsymbol{p}^i) = \begin{bmatrix} (u^i - c_u)\cdot {}^kd^i/f \\ (v^i - c_v)\cdot {}^kd^i/f \\ {}^kd^i \\ 1 \end{bmatrix} \tag{1} $$

where $\pi^{-1}(\cdot)$ is the inverse of projection function, f the focal length and (c_u, c_v) the principal point of the cameras.

The motion of the camera between frames $k-1$ and k and/or the motion of objects in the scene produce an optical flow ${}^k\boldsymbol{\Phi} = \{{}^k\boldsymbol{\phi}^i \in \mathbb{R}^2\}$, where ${}^k\boldsymbol{\phi}^i$ is the corresponding optical flow for each point ${}^k\boldsymbol{p}^i$ and its correspondence ${}^{k-1}\boldsymbol{p}^i$ in frame $k-1$ and is given by:

$$ {}^k\bar{\boldsymbol{p}}^i = {}^{k-1}\bar{\boldsymbol{p}}^i + {}^k\boldsymbol{\phi}^i \tag{2} $$

where ${}^k\bar{\boldsymbol{p}}^i$ and ${}^{k-1}\bar{\boldsymbol{p}}^i$ only contain the 2D point coordinates. ${}^k\boldsymbol{\Phi}$ can be obtained using off-the-shelf classic or learning-based methods. The motions of the camera and objects in the scene are represented by pose change transformations. The following subsections will describe our new approach to estimate those.

A. Camera Motion Estimation

The camera motion between frame $k-1$ (lower left index) and k (lower right index) represented in body-fixed frame $k-1$ (upper left index) is denoted ${}^{k-1}_{k-1}\mathbf{T}_k \in SE(3)$. The image plane points, associated with static 3D points ${}^{k-1}\boldsymbol{m}^i_{k-1}$, observed at time $k-1$, by the projection onto the k image plane can now be computed by

$$ {}^k\hat{\boldsymbol{p}}^i := \pi({}^k\boldsymbol{m}^i_{k-1}) = \pi({}^{k-1}_{k-1}\boldsymbol{T}_k^{-1}\ {}^{k-1}\boldsymbol{m}^i_{k-1}). \tag{3} $$

Parameterize the $SE(3)$ camera motion by elements $\boldsymbol{\xi}^\wedge_k \in \mathfrak{se}(3)$ the Lie-algebra of $SE(3)$. That is

$$ {}^{k-1}_{k-1}\mathbf{T}_k = \exp({}^{k-1}_{k-1}\boldsymbol{\xi}^\wedge_k) \tag{4} $$

where ${}^{k-1}_{k-1}\boldsymbol{\xi}_k \in \mathbb{R}^6$ and the wedge operator is the standard lift into $\mathfrak{se}(3)$. Combining (2) and (3), and using the Lie-algebra parameterization of $SE(3)$ the minimizing solution of the least squares cost criteria we consider is given by

$$ {}^{k-1}_{k-1}\boldsymbol{\xi}^*_k = \underset{{}^{k-1}_{k-1}\boldsymbol{\xi}_k}{\operatorname{argmin}} \sum_{i=1}^{n_s} \rho_h(||{}^{k-1}\bar{\boldsymbol{p}}^i + {}^k\boldsymbol{\phi}^i - {}^k\hat{\bar{\boldsymbol{p}}}^i||^2_{\Sigma_1}) \tag{5} $$

for all the visible 3D-2D static point correspondences $i = 1, ..., n_s$. Here ρ_h is the Huber robust cost function, and Σ_1 is covariance matrix associated to the re-projection threshold used in initialization. The estimated camera motion is given by ${}^{k-1}_{k-1}\mathbf{T}^*_k = \exp({}^{k-1}_{k-1}\boldsymbol{\xi}^*_k)$ and is found using the Levenberg-Marquardt algorithm to solve for (5).

B. Moving Points Motion Estimation

In this section we derive the motion model of 3D points on a rigid body in motion. The motion of the rigid body in body-fixed frame is given by ${}^{L_{k-1}}_{k-1}\boldsymbol{H}_k \in SE(3)$. If the pose of the object at time $k-1$ in global reference frame is given by ${}^0\boldsymbol{L}_{k-1} \in SE(3)$, we showed in [14] and [15] that the *rigid body pose transformation in global frame* is given by

$$ {}^0_{k-1}\boldsymbol{H}_k = {}^0\ \boldsymbol{L}_{k-1}\ {}^{L_{k-1}}_{k-1}\boldsymbol{H}_k\ {}^0\boldsymbol{L}^{-1}_{k-1} \in SE(3). \tag{6} $$

Consequently the motion of a point on a rigid body in global frame is given by ${}^0_{k-1}\boldsymbol{H}_k$, with the following relation:

$$ {}^0\boldsymbol{m}^i_k = {}^0_{k-1}\boldsymbol{H}_k\ {}^0\boldsymbol{m}^i_{k-1}\ . \tag{7} $$

When formulating the motion estimation problem considering only two consecutive frames, the motion in the global frame in (6) would be expressed in the image frame $k-1$, and is denoted ${}^{k-1}_{k-1}\boldsymbol{H}_k$.

As shown in Fig. 2 (right), a 3D point ${}^{k-1}\boldsymbol{m}^i_{k-1}$ observed on a moving object at time $k-1$, moves according to (7) to ${}^{k-1}\hat{\boldsymbol{m}}^i_k = {}^{k-1}_{k-1}\boldsymbol{H}_k\ {}^{k-1}\boldsymbol{m}^i_{k-1}$. The projection of the estimated 3D point onto the image frame at time k is given by

$$ \begin{aligned} {}^k\hat{\mathbf{p}}^i :=&\ \pi\left({}^{k-1}_{k-1}\boldsymbol{T}_k^{-1}\ {}^{k-1}_{k-1}\boldsymbol{H}_k\ {}^{k-1}\boldsymbol{m}^i_{k-1}\right) \\ =&\ \pi\left({}^{k-1}_{k-1}\boldsymbol{X}_k\ {}^{k-1}\boldsymbol{m}^i_{k-1}\right) \end{aligned} \tag{8} $$

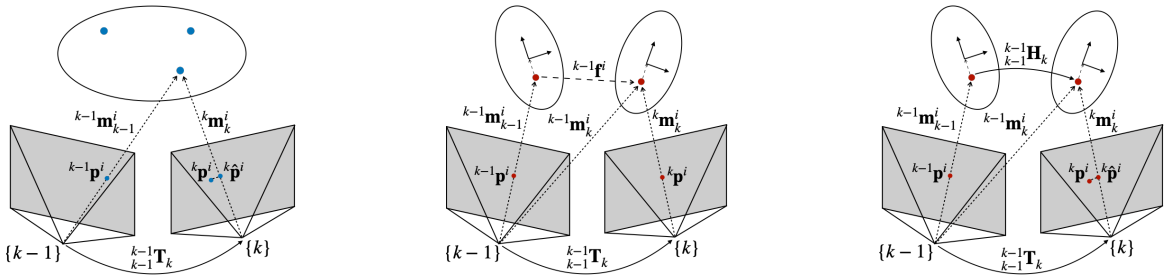

Fig. 2. Sketch maps of ego-motion obtained from static points (left), scene flow of points on moving objects (center) and rigid motion of points on moving object (right). Here blue dots represent static points, and red dots dynamic points.

where ${}^{k-1}_{k-1}\boldsymbol{X}_k \in SE(3)$. Similar to the camera motion estimation, we parameterize ${}^{k-1}_{k-1}\mathbf{X}_k = \exp\left({}^{k-1}_{k-1}\boldsymbol{\zeta}^{\wedge}_k\right)$, with ${}^{k-1}_{k-1}\boldsymbol{\zeta}^{\wedge}_k$ the $\mathfrak{se}(3)$ representation of ${}^{k-1}_{k-1}\boldsymbol{\zeta}_k \in \mathbb{R}^6$, and find the optimal solution via minimizing

$$ {}^{k-1}_{k-1}\boldsymbol{\zeta}^{*}_k = \underset{{}^{k-1}_{k-1}\boldsymbol{\zeta}_k}{\operatorname{argmin}} \sum_{i=1}^{n_o} \rho_h(||{}^{k-1}\bar{\boldsymbol{p}}^i + {}^k\boldsymbol{\phi}^i - {}^k\hat{\bar{\boldsymbol{p}}}^i||^2_{\Sigma_1}) \tag{9} $$

given all the 3D-2D point correspondences on an object $i = 1, ..., n_o$. The motion of the object points, ${}^{k-1}_{k-1}\boldsymbol{H}_k = {}^{k-1}_{k-1}\boldsymbol{T}_k\, {}^{k-1}_{k-1}\boldsymbol{X}_k$ can be recovered afterwards.

C. Refining the estimation of the optical flow

Both, camera motion and object motion estimations rely on good image correspondences. Tracking of points on moving objects can be very challenging due to occlusions, large relative motions and large camera-object distances. In order to insure a robust tracking of points, the technique proposed in this paper aims at refining the estimation of the optical flow jointly with the motion estimation:

$$ \begin{aligned} \{\boldsymbol{\theta}^*, {}^k\boldsymbol{\Phi}^*\} = &\underset{\{\boldsymbol{\theta}, {}^k\boldsymbol{\Phi}\}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho_h(||{}^{k-1}\bar{\boldsymbol{p}}^i + {}^k\boldsymbol{\phi}^i - {}^k\hat{\bar{\boldsymbol{p}}}^i||^2_{\Sigma_1}) \\ &+ \rho_h(||{}^k\boldsymbol{\phi}^i - {}^k\hat{\boldsymbol{\phi}}^i||^2_{\Sigma_2}) \end{aligned} \tag{10} $$

where $\{\boldsymbol{\theta}, {}^k\boldsymbol{\Phi}\}$ can be either $\{{}^{k-1}_{k-1}\boldsymbol{\xi}_k, {}^k\boldsymbol{\Phi}_s\}$ for camera motion estimation, or $\{{}^{k-1}_{k-1}\boldsymbol{\zeta}_k, {}^k\boldsymbol{\Phi}_o\}$ for the object motion estimation, with ${}^k\boldsymbol{\Phi}_s \subseteq {}^k\boldsymbol{\Phi}$ and ${}^k\boldsymbol{\Phi}_o \subset {}^k\boldsymbol{\Phi}$. Here Σ_2 is the covariance matrix associated to initial optic-flow obtained using classic or learning-based methods.

IV. Implementation

In this section, we propose a novel multi-motion visual odometry system that robustly estimates both camera and object motions. Our proposed system takes stereo or RGB-D images as input. If the input data is stereo images, we can apply the method in [16] to generate the depth map $\boldsymbol{D}$. The proposed pipeline is summarised in Fig. 3 and contains three main parts: image preprocessing, ego-motion estimation and object motion tracking.

A. Image Preprocessing

There are two challenging aspects that this pipeline needs to fulfill. One is to separate static background and objects, and the other is to ensure long-term tracking of dynamic objects. For that, we leverage recent advances in computer vision technics for dense optical flow calculation and instance level semantic segmentation in order to ensure good object tracking and efficient object motion segmentation.

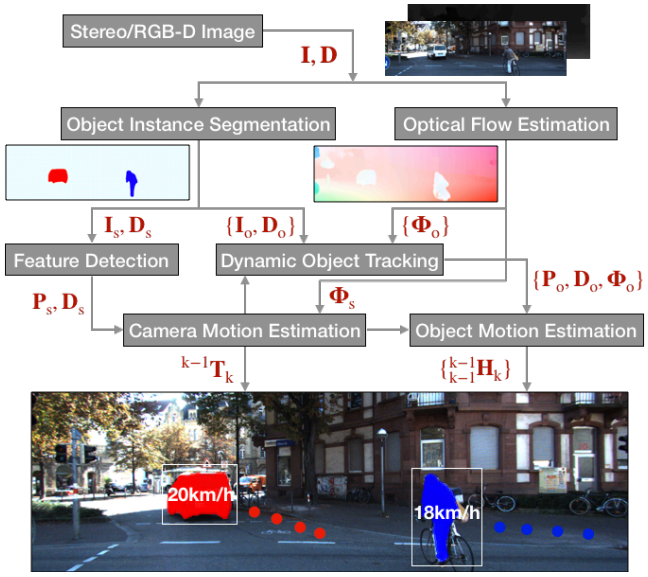

Fig. 3. Overview of our multi-motion visual odometry system. Letters in red colour refer to output for each blocks. $\{\cdot\}$ denotes multiple objects.

The dense optical flow is used to maximize the number of track points on moving objects. Most of the moving objects, they only occupy a small portion of the image. Therefore, using sparse feature matching does not guarantee a robust feature tracking. Our approach makes use of dense optical flow to considerably increase the number of object points. At the same time, our method enhances the matching performance by refining optical flow jointly within the motion estimation process as presented in Section III-C.

Instance-level semantic segmentation is used to segment and identify potentially movable objects in the scene. Semantic information constitutes an important prior in the process of separating static and moving object points, e.g., buildings and roads are always static, but cars can be static or dynamic. Instance segmentation helps to further divide semantic foreground into different instance masks, which makes it easier to track each individual object. Moreover, segmentation mask provides precise boundary of the object body that ensures robust tracking of points on objects.

The image preprocessing part of the pipeline generates the image mask, the depth and the dense flow for the static $\mathbf{I}_s$,

$\mathbf{D}_\mathrm{s}$ and $\mathbf{\Phi}_\mathrm{s}$ and dynamic $\{\mathbf{I}_\mathrm{o}, \mathbf{D}_\mathrm{o}, \mathbf{\Phi}_\mathrm{o}\}$ parts of the scene.

B. Ego-motion Estimation

To achieve fast ego-motion estimation, we construct a sparse feature set $\boldsymbol{P}_s$ in each frame. Since dense optical flow is available, we use optical flow to match those sparse features across frames. Those sparse features are only detected on regions of the image other than labeled objects. To ensure robust estimation, a motion model generation method is applied for initialisation. Specifically, the method generates two models and compares their inlier numbers based on re-projection error. One model is generated by propagating the previous camera motion, while the other by computing a new motion transform using P3P [17] algorithm with RANSAC. The motion model that produces most inliers is then selected for initialisation.

C. Object Motion Tracking

The process of object motion tracking consists of three steps. The first step is to classify all the objects into dynamic and static objects. Then we associate the dynamic objects across the two frames. Finally, individual object motion is estimated.

1) Classifying Dynamic Object: Instance level object segmentation allows us to separate objects from background. Although the algorithm is capable of estimating the motions of all the segmented objects, dynamic object identification helps reduce computational cost of the proposed system. This is done using scene flow estimation as shown in Fig. 2 (center). Specifically, after obtaining camera motion ${}^{k-1}_{k-1}\boldsymbol{T}_k$, the scene flow vector ${}^{k-1}\boldsymbol{f}^i$ describing the motion of a 3D point ${}^{k-1}\boldsymbol{m}^i_{k-1}$ between frame $k-1$ and k, can be calculated as [18]:

$$ {}^{k-1}\boldsymbol{f}^i = {}^{k-1}\boldsymbol{m}^i_{k-1} - ({}^{k-1}_{k-1}\boldsymbol{T}_k \, {}^{k}\boldsymbol{m}^i_k) \tag{11} $$

Unlike optical flow, the scene flow can directly decide whether the scene structure is moving or not. Ideally, the magnitude of the scene flow vector should be zero for static 3D point. However, noise or error in depth and matching complicates the situation in real scenarios.

To robustly tackle this, we compute the scene flow magnitude of all the sampled points on each object, and separate them into two sets (static and dynamic) via thresholding. An object is recognised dynamic if the proportion of "dynamic" points is above a certain level, otherwise static. Table I demonstrates the performance of classifying dynamic and static objects using this strategy. Overall, the proposed approach achieves good accuracy among the tested sequences. Notice that, in sequence 20, we have relatively high false negative cases. That is because most cars throughout sequence 20, move slowly (nearly static) due to traffic jams.

2) Object Tracking: Instance-level object segmentation only provides labels frame by frame, therefore objects need to be tracked between frames and their motion models propagated over time. To manage this, we propose to use optical flow to associate point labels in accross frames. For that, we introduce and maintain a finite tracking label set

TABLE I

PERFORMANCE OF DYNAMIC/STATIC OBJECT CLASSIFICATION OVER VIRTUAL KITTI DATASET.

Sequence	01	02	06	18	20
Total Detection	1383	150	266	970	2091
Dynamic/Static	117/1266	73/77	257/9	970/0	1494/597
False Positive	3	0	9	0	3
False Negative	6	0	0	57	292

$\mathcal{L} \subset \mathbb{N}$ where $l \in \mathcal{L}$ starts from $l = 1$, when the first moving object appears in the scene. The number of elements in $\mathcal{L}$ increases as more objects are being detected. Static objects and background are labeled with $l = 0$.

Ideally, for each detected object in frame k, the labels of all its points should be uniquely aligned with the labels of their correspondences in previous frame $k-1$. However, in practice this is affected by the noise, image boundary and occlusions. To overcome this, we assign all the points with the label that appears most in their correspondences. For a dynamic object, if the most frequent label in the previous frame is 0, it means that the object starts to move, appears in the scene at the boundary, or reappears from occlusion. In this case, the object is assigned with a new tracking label.

3) Object Motion Estimation: As mentioned before, objects normally appear in small sizes in the scene, which makes it hard to get sufficient sparse features to track and estimate their motions robustly. Therefore we densify the object point set $\boldsymbol{P}_o$ via sampling every 3^{rd} pixel within object mask in practice. Similar to the ego-motion estimation, an initial object motion model is generated for initialisation. The model with most inliers is refined using (10) to get the final object motion and the best point matching.

V. EXPERIMENTS

In this section, experimental results on two public datasets are demonstrated. For detailed analysis we use virtual KITTI dataset [19], which provides ground truth of ego/object poses, depth, optical flow and instance level object segmentation. KITTI tracking dataset [20] is used to show the applicability of our algorithm in real life scenarios. We adopt a learning-based method, Mask R-CNN [4], to generate object segmentation in both datasets. The model of this method is trained on COCO dataset [21], and it is directly used without fine-tuning. For dense optical flow, we use a state-of-the-art method, PWC-Net [6]. The model is trained on FlyingChairs dataset [22], and then fine-tuned on Sintel [23] and KITTI training datasets [20]. Feature detection is done using FAST [24].

We use pose change error to evaluate the estimated SE(3) motion, i.e., given ground truth motion $\mathbf{X}$ and estimated $\hat{\boldsymbol{X}}$, where $\mathbf{X} \in SE(3)$ can be either camera or object motion. The pose change error is obtained as: $\mathbf{E} = \hat{\boldsymbol{X}}^{-1}\boldsymbol{X}$. Translation error E_t is computed as the L_2 norm of translational component in $\mathbf{E}$. Rotation error E_R is measured as the angle in axis-angle representation of rotation part of $\mathbf{E}$. We also evaluate object velocity error. According to [25], given an object motion $\boldsymbol{H}$, the object velocity v can be calculated as:

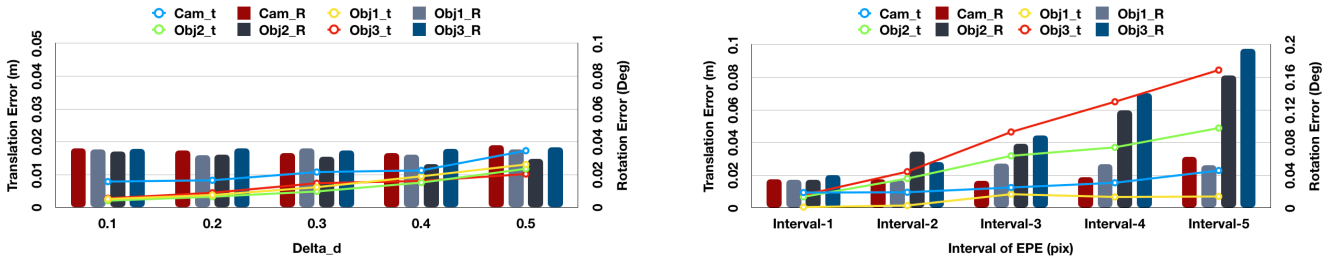

Fig. 4. Average error of rigid motion with regard to noise level of depth (left), and to End-point Error of optical flow (right). Curves represent translation error that are corresponding to left-Y axis, and bars represent rotation error that are corresponding to right-Y axis.

$v = ||\boldsymbol{t} - (\boldsymbol{I} - \boldsymbol{R})\,\boldsymbol{c}||$ where $\boldsymbol{R}$ and $\boldsymbol{t}$ are the rotation and translation part of the motion of points in global reference frame. $\boldsymbol{I}$ is identity matrix and $\boldsymbol{c}$ is centroid of object. Then error of velocity E_v between estimated $\hat{v}$ and ground truth v can be represented as: $E_v = |\hat{v} - v|$. The optical flow is evaluated using end-point error (EPE) [26].

TABLE II

AVERAGE OPTICAL FLOW END-POINT ERROR (EPE) OF STATIC BACKGROUND AND OBJECTS IN S18-F124-134.

	Static	Obj1	Obj2	Obj3
Object Distance (m)	–	7.52	16.52	24.67
Object Area (%)	–	6.29	0.73	0.29
EPE X-axis (pix)	1.34	0.35	0.34	0.15
EPE Y-axis (pix)	0.27	0.24	0.22	0.18

A. Virtual KITTI Dataset

This dataset is used to analyse the influence of the optical flow and depth accuracy on the estimation of the ego and object motion. Moving objects appears scatteredly within a sequence, which makes it hard to perform in-depth tests using the whole sequence. Therefore, we selected a representative set that contains multiple moving objects for analysis. The set is part of the sequence 18 and the frame IDs are between 124-134 (S18-F124-134). It contains 10 frames of the agent car with camera moving forward, and three observed vehicles. Two of them are moving alongside in the left lane, with one closer to the camera and the other farther. The third car is moving upfront and it is furthest from the camera.

TABLE III

AVERAGE ERROR OF OBJECT MOTIONS OF DIFFERENT SETS.

		Motion only		Joint	
		E_t (m)	E_R (deg)	E_t (m)	E_R (deg)
S01-F225-235	Ego	0.0117	0.0354	**0.0043**	**0.0310**
S01-F410-418	Obj	0.0647	0.2811	**0.0470**	**0.2286**
	Ego	0.0367	0.1012	**0.0052**	**0.0315**
S18-F124-134	Obj1	0.0169	0.1016	**0.0132**	**0.0804**
	Obj2	0.1121	0.2720	**0.1008**	**0.1907**

Depth: Ground truth depth is corrupted with zero mean Gaussian noise, with σ following standard depth accuracy of a stereo camera system expressed as: $\sigma = \frac{z^2}{f \cdot b} \cdot \triangle d$ where z is depth, f focal length, b baseline and Δd the disparity accuracy. We set $b = 0.5$ m and control Δd to get the noise level of depth. Normally, Δd varies from 0.1 to 0.2 for a standard industrial stereo camera. Fig. 4 (left) demonstrates the average error of rigid motion over all selected frames. Note that our algorithm is robust to depth noise within reasonable range. The translation error grows gradually with the depth error for both camera and objects, but stays in low range ($E_t < 0.02$m). Rotation error fluctuates slightly but still in low range ($E_R < 0.04$deg).

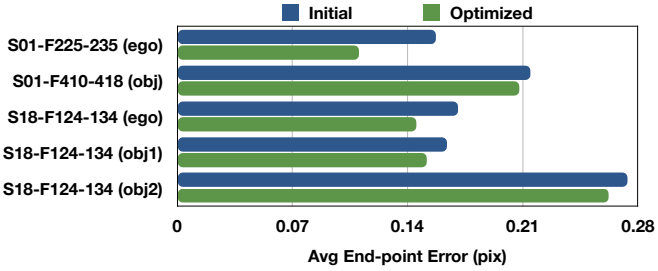

Fig. 5. Average end-point error between initial and optimized optical flow, among different tested sets.

OpticalFlow: The ground truth optical flow is corrupted with zero mean Gaussian noise with σ decided by the end-point error (EPE). Table II demonstrates average EPE of PWC-Net results for the static and object points among this sequence. Since the errors among static background and objects are different, we set five intervals in increasing order and use these average errors as the middle value. For instance, for static points, σ_y is set to [0.09 0.18 0.27 0.36 0.45].

As illustrated in Fig. 4 (right) and Table II, camera and object 1 motion errors are relatively low and stable for different EPEs. However, objects 2 and 3 motion errors increase reaching nearly 0.09 meter in translation and 0.2 degree in rotation and this is because they are far away from the camera and occupy quite small area of the image ($< 1\%$). Consequently, the object motion estimation is sensitive to optical flow error if the objects are not well distributed in the scene. To avoid unreliable estimation, our system addresses only objects within 25m, and 0.5% image presence.

Overall Results: Now overall results without ground truth are demonstrated. Because vKITTI does not provide stereo images, we can not generate depth map. Instead, we use ground truth depth map and add noise with $\Delta d = 0.2$.

As the objects in S18-F124-134 are mainly translating, we introduce two more sets with obvious rotation. One of them (S01-F225-235) contains the agent car (camera) turning left into the main street. The other (S01-F410-418) contains

TABLE IV

AVERAGE VELOCITY ERROR OF SEQUENCES WITH MULTIPLE MOVING OBJECTS.

Sequence	00		01	02	03		04	05	06	18	20
Detected Objects	van	cyclist	5 cars	6 cars	wagon	suv	20 cars	12 cars	10 cars	18 cars	46 cars
Num. of Tracks	44	90	76	39	44	49	109	57	137	431	489
Avg. Velocity (km/h)	18.92	16.06	14.07	34.29	54.44	52.23	30.12	45.22	32.82	20.95	11.73
Avg. Error E_v (km/h)	3.04	2.01	2.02	5.22	2.70	2.63	5.13	5.52	4.26	1.96	2.18

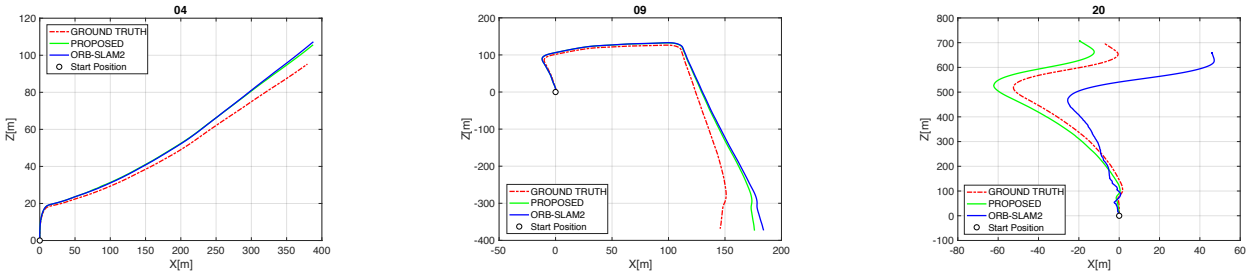

Fig. 6. Top view of camera trajectories of three tested KITTI sequences.

static camera observing one car turning left at the crossroads. To prove the effectiveness of jointly optimising motion and optical flow, we set a baseline method that only optimises for motion (Motion Only) using (5) for camera or (9) for object, and the improved method that optimises for both motion and optical flow with prior constraint (Joint) using (10).

As illustrated in Table III, optimising for the optical flow jointly with the $SE(3)$ motions improve the results, about 300% for the camera motion, and 10-20$\%$ on object motion. Besides, the corresponding optical flow error after optimisation is also reduced, see Fig. 5.

B. Real KITTI Dataset

In KITTI tracking dataset, there are 21 sequences with ground truth camera and object poses. For camera motion, we compute the ego-motion error on all the sequences (12 in total) except the ones that the camera is not moving at all. We also generate results of a state-of-the-art method, ORB-SLAM2 [27] for comparison. Fig. 6 illustrates the camera trajectory results on three sequences. Compared with ORB-SLAM2, our proposed method is able to produce smooth trajectories that are more consistent with the ground truth, given the fact that our method conducts only frame-by-frame tracking, while ORB-SLAM2 integrates more complex modules, such as local map tracking and local bundle adjustment. In particular, the result of Seq. 20 in Fig. 6 (right) shows that ORB-SLAM2 obtains bad estimates in first half of the sequence, mainly because this part contains dynamic scenes of traffic on the highway. Nevertheless, our algorithm is robust against this case. Table V illustrates average motion error over all the 12 tested sequences. The results prove our improved performance over ORB-SLAM2.

For object motion, we demonstrate the results of object velocity error among 9 sequences that contains considerable number of moving objects, since vehicle velocity is important information for autonomous and safety driving applications. As demonstrated in Table IV, the number of tracks refers to how many frames those objects are being tracked. This indicates our pipeline is able to simultaneously and robustly track multiple moving objects for long distances. The average velocity error E_v is computed over all the tracks among one or all objects (see the second row in Table IV). Overall, our method gets around 2-5km/h error, which is considerably accurate for the velocity ranging from 11-55km/h.

TABLE V

AVERAGE EGO-MOTION ERROR OVER 12 TESTED SEQUENCES.

	PROPOSED	ORB-SLAM2
E_t (m)	**0.0642**	0.0730
E_R (deg)	**0.0573**	0.0622

The computational cost of our algorithm is around 6fps when run on an i7 2.6Ghz laptop. The main cost lies in denser points tracking on multiple objects. This can be improved by employing parallel implementation to achieve real-time performance.

VI. CONCLUSION

In this paper we present a novel framework to simultaneously track camera and multiple object motions. The proposed framework detects moving objects via combining instance-level object segmentation and scene flow, and tracks them over frames using optical flow. The SE(3) motions of the objects, as well as the camera are optimised jointly with the optical flow in a unified formulation. We carefully analyse and test our approach on virtual KITTI dataset, and demonstrate its effectiveness. Furthermore, we perform extensively test on the real KITTI dataset. The results show that our method is able to obtain robust and accurate camera trajectories in dynamic scene, and track the velocity of objects with high accuracy. Further work will integrate the proposed motion estimation within a SLAM framework.

ACKNOWLEDGMENT

This research is supported by the Australian Research Council through the Australian Centre of Excellence for Robotic Vision (CE140100016).

REFERENCES

[1] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[2] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust Monocular SLAM in Dynamic Environments," in *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 209–218.

[3] P. F. Alcantarilla, J. J. Yebes, J. Almazán, and L. M. Bergasa, "On Combining Visual SLAM and Dense Scene Flow to Increase the Robustness of Localization and Mapping in Dynamic Environments," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1290–1297.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[5] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017. [Online]. Available: http://lmb.informatik.uni-freiburg.de//Publications/2017/IMKDB17

[6] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-NET: CNNs for Optical Flow using Pyramid, Warping, and Cost Volume," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8934–8943.

[7] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual SLAM and Structure from Motion in Dynamic Environments: A Survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 2, p. 37, 2018.

[8] D. Zou and P. Tan, "CoSLAM: Collaborative Visual SLAM in Dynamic Environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 2, pp. 354–366, 2013.

[9] A. Kundu, K. M. Krishna, and C. Jawahar, "Realtime Multibody Visual SLAM with A Smoothly Moving Monocular Camera," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2080–2087.

[10] N. D. Reddy, I. Abbasnejad, S. Reddy, A. K. Mondal, and V. Devalla, "Incremental Real-time Multibody vSLAM with Trajectory Optimization using Stereo Camera," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 4505–4510.

[11] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based Detection and Tracking in 3D Lidar Scans," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4508–4513.

[12] 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.

[13] K. M. Judd, J. D. Gammell, and P. Newman, "Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-party Motions," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 3949–3956.

[14] J. Zhang and V. Ila, "Multi-frame Motion Segmentation for Dynamic Scene Modelling," in *The 20th Australasian Conference on Robotics and Automation (ACRA)*. Australian Robotics & Automation Association, 2018.

[15] M. Henein, J. Zhang, R. Mahony, and V. Ila, "Dynamic SLAM: The Need For Speed," *IEEE International Conference on Robotics and Automation (ICRA). To appear*, 2020.

[16] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation," in *European Conference on Computer Vision*. Springer, 2014, pp. 756–771.

[17] T. Ke and S. I. Roumeliotis, "An Efficient Algebraic Solution to the Perspective-three-point Problem," in *CVPR*, 2017.

[18] Z. Lv, K. Kim, A. Troccoli, D. Sun, J. M. Rehg, and J. Kautz, "Learning Rigidity in Dynamic Scenes with A Moving Camera for 3D Motion Field Estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 468–484.

[19] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual Worlds as Proxy for Multi-Object Tracking Analysis," in *CVPR*, 2016.

[20] A. Geiger, P. Lenz, and R. Urtasun, "Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[22] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4040–4048.

[23] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A Naturalistic Open Source Movie for Optical Flow Evaluation," in *European Conference on Computer Vision (ECCV)*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.

[24] E. Rosten and T. Drummond, "Machine Learning for High-speed Corner Detection," in *European Conference on Computer Vision*. Springer, 2006, pp. 430–443.

[25] G. S. Chirikjian, R. Mahony, S. Ruan, and J. Trumpf, "Pose Changes from A Different Point of View," in *Proceedings of the ASME International Design Engineering Technical Conferences (IDETC) 2017*. ASME, 2017.

[26] D. Sun, S. Roth, and M. J. Black, "A Quantitative Analysis of Current Practices in Optical Flow Estimation and the Principles behind Them," *International Journal of Computer Vision*, vol. 106, no. 2, pp. 115–137, 2014.

[27] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.