Visual SLAM and Structure from Motion in Dynamic Environments: A Survey

MUHAMAD RISQI U. SAPUTRA, ANDREW MARKHAM, and NIKI TRIGONI, Department of Computer Science, University of Oxford

In the last few decades, Structure from Motion (SfM) and visual Simultaneous Localization and Mapping (visual SLAM) techniques have gained significant interest from both the computer vision and robotic communities. Many variants of these techniques have started to make an impact in a wide range of applications, including robot navigation and augmented reality. However, despite some remarkable results in these areas, most SfM and visual SLAM techniques operate based on the assumption that the observed environment is static. However, when faced with moving objects, overall system accuracy can be jeopardized. In this article, we present for the first time a survey of visual SLAM and SfM techniques that are targeted toward operation in dynamic environments. We identify three main problems: how to perform reconstruction (robust visual SLAM), how to segment and track dynamic objects, and how to achieve joint motion segmentation and reconstruction. Based on this categorization, we provide a comprehensive taxonomy of existing approaches. Finally, the advantages and disadvantages of each solution class are critically discussed from the perspective of practicality and robustness.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → **Image segmentation**; **Tracking**; **Reconstruction**;

Additional Key Words and Phrases: Structure from motion, visual SLAM, deep learning, 3D reconstruction, visual odometry, motion segmentation, dynamic object segmentation, 3D tracking, dynamic environments

ACM Reference format:
Muhamad Risqi U. Saputra, Andrew Markham, and Niki Trigoni. 2018. Visual SLAM and Structure from Motion in Dynamic Environments: A Survey. *ACM Comput. Surv.* 51, 2, Article 37 (February 2018), 36 pages.
https://doi.org/10.1145/3177853

1 INTRODUCTION

The problems of estimating camera pose and reconstructing the three-dimensional model of the environment has drawn significant attention from many researchers over the past few decades. Techniques for solving this problem come from both computer vision and robotic research communities by means of Structure from Motion (SfM) and visual Simultaneous Localization and Mapping (visual SLAM). Standard SfM and visual SLAM aim to simultaneously estimate the camera pose and 3D structure of the scene through a set of feature correspondences detected from multiple images. By choosing whether to integrate feature measurements from all images by

Authors' addresses: M. R. U. Saputra, A. Markham, and N. Trigoni, Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, Oxford, OX13QD, United Kingdom; email: firstname.lastname@cs.ox.ac.uk.

https://doi.org/10.1145/3177853

estimating the probability distribution or to optimize over selected images, the estimation problem can be solved by filter-based approaches (e.g., Kalman filter) or bundle adjustment (BA) [150].

MonoSLAM [29] can be considered as the first filter-based approach to bring the general SLAM problem from the robotic community into pure vision. It enables the propagation of first-order uncertainty of camera positions and feature measurement through a Bayesian framework under real-time computational constraints for robot navigation. In the computer vision community, Longuet-Higgins's paper [53] was probably the first work that led to the emergence of a flurry of SfM techniques. He discovered that the computation of relative camera pose can be done using 8-point correspondences from two views under epipolar geometry. Subsequently, other different perspectives for solving the problem including factorization [151, 154] and rotation averaging [50, 107] appeared. Some widely adopted systems are publicly available such as Bundler [147, 148] or VisualSfM [174], although they work best in batch mode.

The different goals and characteristics in the early work of SfM (offline) and visual SLAM (online) made the paths traveled by the computer vision and robotics communities different and largely disconnected. However, the work of [109, 110, 117] and PTAM [75] brought the two communities together by introducing incremental SfM that can operate in real time. Furthermore, the results from [150] indicate that incremental SfM based on bundle adjustment is more accurate than visual SLAM based on filtering given the same amount of computation time. Many visual SLAM solutions from the robotic community such as [94] or [113] were then developed based on incremental SfM. On the other hand, due to the growing need for more detailed maps and the availability of affordable depth cameras like Microsoft Kinect, solutions capable of producing a dense or semidense map, e.g., KinectFusion [115] or LSD-SLAM [36], are gaining more popularity.

Despite the remarkable results in SfM and visual SLAM, most approaches work based on the assumption that the observed environments are static. Since the real world contains dynamic objects, current approaches are prone to failure due to false correspondences or occlusion of previously tracked features [152]. Pose estimation might drift or even be lost as there are not sufficiently many features to be matched. There is a clear need to devise localization techniques that are robust under these circumstances. Robust pose estimation or localization in a dynamic environment is paramount for a number of applications such as robot navigation [10, 108, 149], driverless cars [102, 145], or emergency response tasks [23, 127].

Another perspective to look at the SLAM problem in dynamic environments is not only to provide robust localization but also to extend its capability into detecting, tracking, and reconstructing the shape of the dynamic objects. To this end, [169] and [170] employed a laser scanner to track moving objects using a Bayesian approach and created in a system called SLAMMOT (Simultaneous Localization, Mapping, and Moving Object Tracking). The computer vision community also studied the Multibody Structure from Motion (MBSfM) topic, a generalization of SfM for multiple rigid body motions [12, 25]. With the proliferation of mobile and wearable devices, this natural extension of visual SLAM in dynamic environments will benefit many applications, including obstacle avoidance [63], human-robot interaction [51], people following [183], path planning [19], cooperative robotics [46], collaborative mapping [28], driverless cars [102], augmented reality (e.g., mobile phone [76], wearable device [18]), or navigation assistance for the visually impaired [4, 134].

This article reviews visual localization and 3D reconstruction techniques in dynamic environments, which covers three main problems: how to perform robust visual SLAM, how to segment and track dynamic objects in 3D, and how to achieve joint motion segmentation and reconstruction. We provide a taxonomy of the existing approaches and connect the fields of visual SLAM and dynamic object segmentation. Finally, we critically discuss the advantages and disadvantages of existing approaches from a practical perspective.

1.1 Comparison to Other Surveys

There are a number of survey papers related to SfM and visual SLAM. Huang et al. (1994) [64] discussed early development of SfM algorithms that focused on the reconstruction algorithm and its performance depending on the feature correspondence types. Oliensis (2000) [118] provided a critical review of multiple view reconstruction approaches (i.e., optimization, fusing by Kalman filter, projective methods, and invariant-based methods). They suggested that experiments and algorithm design should be based on theoretical analyses of the algorithm behavior. Bonin-Font et al. (2008) [10] discussed visual navigation for mobile robotics and divided the techniques into map-based navigation and mapless-based navigation. Fuentes-Pacheco et al. (2012) [40] reviewed visual SLAM approaches highlighting that visual SLAM techniques are prone to failure if the dynamic elements of the environment are not taken into account. However, the paper did not delve into the problems of dynamic scenes or describe existing techniques in this area.

Recent review papers discussed various flavors of visual SLAM. Yousif et al. (2015) [181] surveyed general visual SLAM approaches covering Visual Odometry (VO) and Visual SLAM, including filter, nonfilter, and RGB-D-based solutions. The fundamental techniques used in both VO and Visual SLAM are presented to assist the community to choose the best techniques for a particular task. Similarly, Younes et al. (2016) [180] also discussed recent techniques in visual SLAM but focused on non-filter-based techniques only. They compared and made a critical assessment of specific strategies used by each technique. On the other hand, Garcia-Fidalgo et al. (2015) [43] focused on topological mapping that models the environment as a graph. They categorized the main solutions from 2000 to 2015 based on the type of image descriptors and discussed the advantages and disadvantages of each solution.

From the existing surveys, it can be seen that no work has addressed the specific problem of dynamic environments. To the best of our knowledge, this article is the first survey article discussing in detail visual localization and 3D reconstruction techniques in dynamic environments.

1.2 Article Organization

This article is organized as follows: Section 2 defines the problem and the general application of visual SLAM in dynamic environments. A taxonomy of existing approaches and the high level pipeline connecting them is also provided. Sections 3, 4, and 5 discuss existing techniques on robust visual SLAM, dynamic object segmentation and 3D tracking, and joint motion segmentation and reconstruction, respectively. Advantages and disadvantages of each approach are critically reviewed in Section 6. Finally, Section 7 concludes the article and discusses directions for future work.

2 TAXONOMY OF EXISTING APPROACHES

The problem of simultaneous localization and reconstruction in dynamic environments can be viewed from two different perspectives: either as a robustness problem or as an extension of standard visual SLAM in dynamic environments. As a robustness problem, pose estimation in visual SLAM should remain accurate despite the presence of multiple moving objects in front of the camera, which might result in false correspondences or occlusion of the previously tracked features. Robustness is achieved by segmenting the static and dynamic features in the image and regarding the dynamic parts as outliers. Pose estimation is then computed based on the static parts only. From the perspective of extending visual SLAM into dynamic environments, the system should be capable of segmenting the tracked features into different clusters, each associated with a different object or body. Then, each object structure (shape) can be reconstructed and its trajectory tracked. If and when the static point cloud is available, the system can even insert the dynamic object into the static map.

Based on this general problem of visual SLAM in dynamic environments, we first divide existing approaches based on the application and its corresponding output. Broadly, the three classes can be viewed as techniques that build static maps by rejecting dynamic features (Robust Visual SLAM); techniques that extract moving objects, ignoring the static background (Dynamic Object Segmentation); and techniques that attempt to simultaneously handle the static and dynamic components of the world (Joint Motion Segmentation and Reconstruction). For each application, we further identify a sequence of actions necessary to produce the output. Existing methods from each action category are then classified based on the similarity of the fundamental technique they use. Finally, the taxonomy of the existing approaches is described as follows, which outlines the structure of the remainder of this review:

A. Robust Visual SLAM
    1. Motion Segmentation
        1. Background/Foreground Initialization
        2. Geometric Constraints
        3. Optical Flow
        4. Ego-Motion Constraints
        5. Deep Learning
    2. Localization and 3D Reconstruction
        1. Feature Based
        2. Deep Learning

B. Dynamic Object Segmentation and 3D Tracking
    1. Dynamic Object Segmentation
        1. Statistical Model Selection
        2. Subspace Clustering
        3. Geometry
        4. Deep Learning
    2. 3D Tracking of Dynamic Objects
        1. Trajectory Triangulation
        2. Particle Filter

C. Joint Motion Segmentation and Reconstruction
    1. Factorization
        1. Multibody Structure from Motion (MBSfM)
        2. Nonrigid Structure from Motion (NRSfM)

Figure 1 depicts how each approach connects to others and forms a full pipeline of visual SLAM in dynamic environments. The pipeline consists of three main applications: (A) robust visual SLAM (input: feature correspondences/image sequences, output: 3D point cloud of static world), (B) Dynamic Object Segmentation and 3D Tracking (input: feature correspondences/image sequences, output: 3D trajectory of each object), and (C) Joint Motion Segmentation and Reconstruction (input: feature correspondences, output: 3D point cloud of static features and dynamic features). Although we only classify the application into three categories, more applications are possible with different configurations; e.g., for the robot-following-people scenario, reconstructing static features and tracking the object in image space or in 2D (instead of 3D) might be enough. Finally, the output from application A and B can be combined to obtain a similar output from application C.

3 ROBUST VISUAL SLAM

Robust visual SLAM in dynamic environments can be achieved if pose estimation is computed based solely on static features. Figure 2 depicts the flow diagram of robust visual SLAM together

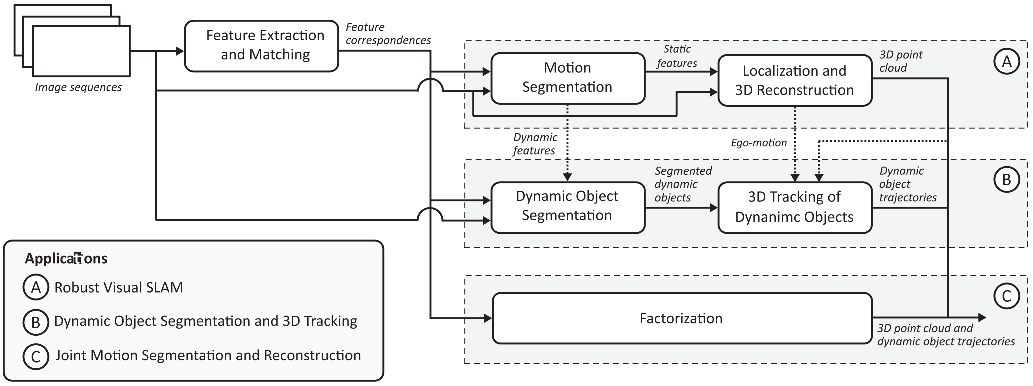
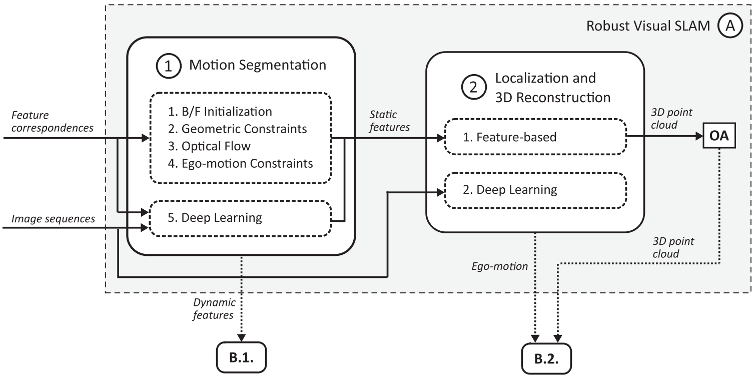


Image sequences
Feature Extraction and Matching
Feature correspondences
Motion Segmentation
Static features
Localization and 3D Reconstruction
3D point cloud
A
Dynamic features
Ego-motion
Dynamic Object Segmentation
Segmented dynamic objects
3D Tracking of Dynanimc Objects
Dynamic object trajectories
B
Factorization
3D point cloud and dynamic object trajectories
C
Applications
A Robust Visual SLAM
B Dynamic Object Segmentation and 3D Tracking
C Joint Motion Segmentation and Reconstruction

Fig. 1. High-level diagram describing the pipeline of visual localization and 3D reconstruction in dynamic environments. Rounded rectangles indicate an action module (approach category), solid arrows denote data transfer, and dashed arrows reflect an optional input. Some actions have input from both feature correspondences and image sequences since the corresponding techniques consist of feature-based and deep-learning-based approaches.



Index	References
A.1.1.	[21], [89], [90], [172], [173], [184]
A.1.2.	[28], [84], [95], [104], [152]
A.1.3.	[4], [30], [31], [74]
A.1.4.	[133], [136]
A.1.5.	[38], [96], [160]
A.2.1.	[75], [94], [109]-[111], [113], [117], [147], [148], [174]
A.2.2.	[32], [77], [78], [85], [103], [105], [112], [168], [171], [187]

Fig. 2. The flow diagram of the first application, robust visual SLAM. Solid rounded rectangles indicate an action and dashed rounded rectangles show existing approaches for a specific action module. A square box shows the output of a particular module. Solid arrows denote data transfer and dashed arrows reflect an optional input. The table on the right side shows the list of relevant literature references for each approach.

with the available approaches and the corresponding references. It can be seen that the input of the application is either the image sequence directly or the extracted feature correspondences depending on whether a deep-learning-based approach is employed or not. The application contains two major modules: (1) motion segmentation and (2) localization and 3D reconstruction. Motion segmentation classifies features into static and dynamic features, but only static features are used for localization and 3D reconstruction of the world. On the other hand, dynamic features and 3D point cloud data (output OA) can be directed to application B through action module B.1. and B.2 for further processing. This section discusses approaches in motion segmentation and standard localization and 3D reconstruction techniques for robust visual SLAM.

3.1 Motion Segmentation

Motion segmentation (also known as *moving object detection/segmentation* [30, 74, 84]) detects moving parts in the image by classifying the features into two different groups, static and dynamic features. Specifically, given a set of feature points $W = \{x_i \in \mathbb{R}^2\}_{i=1}^{n}$ in image space, motion segmentation clusters the feature points into $W_1 = \{x_1, \ldots, x_m\}$ and $W_2 = \{x_{m+1}, \ldots, x_n\}$ for the static and dynamic set, respectively, where $W_1 \cap W_2 = \emptyset$. Standard visual SLAM achieves this by computing geometric models (e.g., fundamental matrix, homography) using a robust statistical approach, such as by Random Sample Consensus (RANSAC) [37], and excludes feature points that do not conform with the model. Specific distance metrics such as the Sampson distance [59] are used to determine the exclusion. This approach will work well if the static features are in the majority. When the dynamic objects in front of the camera are dominant or the captured scene is occluded by a large moving object, these types of approaches may fail. Other approaches leverage external sensors such as an inertial measurement unit (IMU) to solve this problem [67, 92] by estimating the camera ego-motion. Pose estimation from the IMU can be used to initialize the camera pose and segment static and dynamic features robustly. In this section, we discuss alternative approaches to segment static and dynamic features beyond the standard visual SLAM or visual-inertial SLAM techniques (see Table 1 for a summary of existing approaches).

3.1.1 Background-Foreground Initialization. Background-foreground initialization techniques assume that the system has prior knowledge about the environment and leverages that information to segment static and dynamic features. This prior knowledge can be attached to either background (static features) or foreground objects (dynamic features). If the information is about the foreground object, it means that the system has knowledge about the type or the shape of the object that moves in front of the camera.

Most approaches in foreground initialization make use of the *tracking-by-detection* scheme [14, 89]. Wangsiripitak et al. [173] assume a 3D object where the dynamic features lie is known. They used a 3D polyhedral object modeled by a set of control points along the edges and tracked it using Harris's RaPid tracker [56]. If the previously tracked features lie on the tracking object, it will be removed as soon as the object is detected as moving. Any static features that are occluded by the object are removed as well. Similarly, Wang et al. [172] assumed that a set of SURF feature descriptors [8] belonging to the moving object are known and stored in the database. By comparing the descriptors obtained from the feature detection step, the moving object is identified and its displacement and orientation are estimated. Chhaya et al. [21] modeled vehicles in front of the camera using a deformable wireframe object class model. The model is trained on 3D CAD data using Principal Component Analysis (PCA). This model is used to recognize and to segment the car from pose estimation computation. On the other hand, Lee et al. [89, 90] used a pretrained human detector to track pedestrians via the tracking-by-detection scheme. They employed the Constrained Multiple-Kernel (CMK) approach to handle occlusions during tracking by taking into account depth information.

Instead of initializing the foreground object, background initialization sets a background model similarly found in *background subtraction* techniques [7, 126]. Zhang et al. [184] initialized a set of feature points that belong to the background and set it as the background model. They assumed that there is no foreground object when the visual localization is first initialized. Then, when a new frame is processed, 3D motion segmentation is applied using GPCA [165]. Segmented motion with the highest correspondence to the prior background model is used to update the background. Pose estimation is computed using standard epipolar geometry based on the new background model.

3.1.2 Geometric Constraints. Techniques that rely on geometric constraints leverage epipolar geometry properties [59] to segment static and dynamic features. They are based on the fact that dynamic features will violate standard constraints defined in multiple-view geometry for static

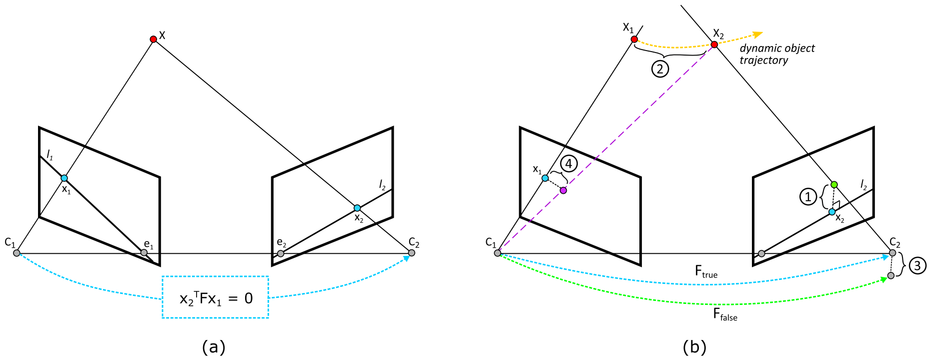


Fig. 3. (a) In static scenes, the transformation of image point from x_1 to x_2 is defined by epipolar constraint $x_2^T F x_1 = 0$. (b) The violation of geometric constraints in a dynamic environment: (1) the tracked feature lies too far from the epipolar line, (2) back-projected rays from the tracked features do not meet, (3) faulty fundamental matrix estimation when dynamic feature is included in pose estimation, (4) high distance between reprojected feature and the observed feature.

scenes (see Figure 3(a)). The constraints can be derived from the equation of epipolar lines, triangulation, fundamental matrix estimation, or reprojection error as seen in Figure 3(b).

Kundu et al. [84] construct the fundamental matrix from robot odometry to define two geometric constraints. The first constraint is derived from the epipolar geometry, which states that a matched point in the subsequent view should lie on the corresponding epipolar line. If the tracked feature resides too far from the epipolar line, then it is most likely a dynamic feature. The second constraint is Flow Vector Bound (FVB), which is aimed to segment degenerate motion that occurs when a 3D point moves along the epipolar line. By setting upper and lower bounds on the flow of the tracked features, a tracked feature that lies outside the bound will be detected as moving. Finally, the decision of classifying features as static or dynamic is determined by a recursive Bayes filter. Instead of using the epipolar line, Migliore et al. [104] segment static and dynamic features by the principle of triangulation. They continuously check the intersection between three projected viewing rays in three different views under a probabilistic filtering framework. If a feature is dynamic, the intersection of the rays is not the same or may not even occur during motion. However, since the sensor measurement is noisy, they employed Uncertain Projective Geometry [61] to check the relationships between viewing rays while taking into account the uncertainty of measurement. The classification of static and dynamic features is then determined via a statistical hypothesis test.

Lin et al. [95] detect moving objects based on an observation that misclassifying a moving object into a static object and incorporating it into the pose estimation would significantly degrade the SLAM performance. They compute the difference of pose estimation under two distinct conditions, one without adding the detected new feature and the other one with including the new feature under assumption that it is stationary. By computing the distance between the two results, setting a threshold value, and integrating it through a binary Bayes filter, they are able to segment stationary and moving features with high accuracy.

Another geometric approach is to leverage the reprojection error. Zou and Tan [28] project features from the previous frame into the current frame and measure the distance from the tracked features. The classification of static and dynamic features is determined by the magnitude of their reprojection distances. Tan et al. [152] also use a similar projection principle to detect dynamic features. However, they also take into account occlusion handling to provide robust visual SLAM. After a feature is projected into the current frame, appearance differences are used to check whether a part of the image has changed. If the appearance changes significantly, it is very likely that the region may be occluded by a dynamic object or by a static object due to viewpoint changes. 3D points occluded by those conditions will be kept and used to robustly estimate the camera pose.

3.1.3 Optical Flow. Optical flow defines the apparent motion of brightness patterns computed from two consecutive images [62]. Generally, it corresponds to the motion field in an image, and thus it can be used to segment a moving object. Klappstein [74] defined a likelihood of a moving object based on a motion metric computed from the optical flow. The motion metric measures to what extent the optical flow is violated if there is a moving object on the scene. The graph-cut algorithm is utilized to segment the moving objects based on the motion metric.

Alcantarilla et al. [4] segment moving objects based on the modulus of the 3D motion vector in scene flow (3D version of optical flow) through residual motion likelihoods. The Mahalanobis distance is used to take into account measurement uncertainty in computing scene flow based on dense optical flow and stereo reconstruction. If the residual is low, the feature point most likely belongs to the static object. By thresholding on the residual motion likelihoods, the feature points that reside on the moving object can be deleted from the SLAM process, making visual odometry estimation more robust. Derome et al. [30, 31] compute optical flow by calculating the residual between the predicted image with the observed image from a stereo camera. By processing backward in time, the predicted image is computed by transforming the current stereo frame into the previous frame using estimated camera ego-motion. Moving objects are then observed by detecting blobs in the residual field.

3.1.4 Ego-Motion Constraints. Standard SfM and visual SLAM compute the motion of the camera by means of the 8-point [53] or the 5-point algorithms [116]. This general ego-motion estimation is calculated without making any assumption on how the camera moves. Another way to estimate the camera pose is by assuming that the camera moves according to particular parameterization given external information (e.g., wheel odometry information). By enforcing this ego-motion constraint, classifying static features can be done by fitting feature points that match with the camera motion constraints.

Scaramuzza [136] proposed to use nonholonomic constraints of wheeled vehicles to compute camera motion. He modeled the ego-motion based on the assumption that the camera motion is planar and circular. By using this constraint, the camera ego motion can be parameterized by one Degree of Freedom (DOF) and can be computed by the 1-point algorithm [137]. Similarly, Sabzevari et al. [133] also employed the wheeled vehicle constraint to estimate camera motion by leveraging Ackermann steering geometry. Feature points satisfying the estimated camera motion are considered as static features, while other points are regarded as dynamic features.

3.1.5 Deep Learning for Motion Segmentation. After winning the ImageNet object recognition competition by reducing classification errors by half compared with state-of-the-art techniques [79], Deep Neural Networks (DNNs) have gained much popularity in the computer vision community. DNNs are a representation learning technique that aim to learn high-level abstractions of the data by using multiple hierarchical layers of neural networks [52, 88]. The main characteristic of DNNs are that they can process raw input data directly without the necessity of hand-engineered feature extraction. This technique has started to make significant changes in many research areas, including ones that were previously considered as not possible to cast them as a learning problem due to the involvement of geometric transformations [55]. While a number of implementations of DNN for visual localization and 3D reconstruction have started to emerge (discussed in Section 3.2.2), DNNs for motion segmentation are still scarce.

Table 1. Summary of Existing Approaches for Motion Segmentation

		Camera[a]			Practical Consideration[b]						
Author(s)	**SLAM**	**T**	**S**	**M**	**CT**	**OT**	**NP**	**TS**	**TU**	**OH**	**DM**
Background/Foreground Initialization (Section 3.1.1)											
Wang et al. [172]	Filter	M	M	P	NT	-	-	✓	✓	-	-
Zhang et al. [184]	SfM	M	M	P	NB	✓	-	✓	✓	-	-
Lee et al. [89]	SfM	M	M	P	NT	✓	-	✓	✓	-	-
Chhaya et al. [21]	SfM	M	M	P	NT	✓	-	✓	-	✓	-
Lee et al. [90]	SfM	M	M	P	NT	✓	-	✓	-	✓	-
Wangsiripitak et al. [173]	Filter	M	M	P	RT	✓	-	✓	✓	✓	-
Geometric Constraints (Section 3.1.2)											
Lin et al. [95]	Filter	S	M	P	RT	-	✓	-	✓	-	-
Migliore et al. [104]	Filter	M	M	P	RT	✓	✓	-	✓	-	-
Zou et al. [28]	SfM	M	M	P	RT	✓	✓	-	✓	-	-
Tan et al. [152]	SfM	M	M	P	RT	✓	✓	-	-	✓	-
Kundu et al. [84]	SfM	M	M	P	RT	✓	✓	-	✓	-	✓
Optical Flow (Section 3.1.3)											
Alcantarilla et al. [4]	SfM	S	M	P	RT	✓	✓	-	-	-	-
Klappstein et al. [74]	SfM	M,S	M	P	NT	-	✓	-	✓	-	-
Derome et al. [30, 31]	SfM	S	M	P	RT	-	✓	-	✓	✓	-
Ego-Motion Constraints (Section 3.1.4)											
Scaramuzza [136]	SfM	M	M	P	RT	✓	-	-	-	-	-
Sabzevari et al. [133]	SfM	M	M	P	RT	✓	-	-	-	-	-
Deep Learning (Section 3.1.5)											
Lin et al. [96]	-	S	M	P	RT	I	✓	-	-	-	-
Fragkiadaki et al. [38]	-	M	S	P	FO	I	✓	-	-	✓	-
Valipour et al. [160]	-	M	S,M	P	NT	I	✓	-	-	-	-

[a] Camera Type (**T**): Monocular (**M**), Stereo (**S**). Camera State (**S**): Static (**S**), Moving (**M**). Camera Model (**M**): Orthography (**O**), Affine (**A**), Perspective (**P**).

[b] **CT**: Computation Time (**RT**: Real time, **NT**: Near real time, **NB**: Need to be batched, **FO**: Fully offline), **OT**: Handle outliers due to false feature correspondences (**I**: irrelevant for the technique), **NP**: No prior knowledge (e.g., background/foreground information, camera motion), **TS**: Supports temporary stopping (ability to keep track of the dynamic objects when they are temporarily stationary), **TU**: Takes into account uncertainty, **OH**: Occlusion handling, **DM**: Supports degenerate motion for the moving objects.

From feature-based motion segmentation, we know that the moving objects can be segmented by leveraging optical flow. Dosovitskiy et al. [33] show that estimating optical flow can be done through supervised learning. They proposed two different architectures of Convolutional Neural Network (CNN) for predicting optical flow. The first architecture (FlowNetS) is designed by stacking two consecutive images as an input of CNN and the other one (FlowNetC) is by introducing a correlation layer to compare two feature maps resulting from two identical CNN streams. Ilg et al. [66] improved this approach into “FlowNet 2.0” by stacking FlowNetS and FlowNetC into a deeper network and adding a new parallel network to handle small displacements. Experimental results show that FlowNet 2.0 can achieve competitive results with the state-of-the-art methods. An extension to scene flow estimation using stereo images is also shown by Mayer et al. [101]. This optical flow can be fed into a deeper network to discover the motion features as shown in [48]. These motion features are shown to be useful for action recognition [47, 146], although it is not clear whether the same network can be used to segment moving objects and provide motion boundaries since it is not explicitly designed for solving the motion segmentation problem.

Lin and Wang [96] construct a network to explicitly segment moving objects in an image space. They employ Reconstruction Independent Component Analysis (RICA) autoencoders [86, 87] to learn spatiotemporal features. However, geometric features are still used to help segment the motion since the spatiotemporal features cannot learn the 3D geometry of the motion. Both geometric and spatiotemporal features are fed into Recursive Neural Networks (RNNs) for final motion segmentation. Using a different approach, Fragkiadaki et al. [38] segments moving objects by regressing the objectness score given RGB image and optical flow. Two parallel CNNs similar to AlexNet [79] are constructed to process RGB images and optical flow before feeding it to the regression network and generating the motion proposal. Recently, Valipour et al. [160] propose Recurrent Fully Convolutional Network (R-FCN) to incorporate temporal data in segmenting foreground motion from online image sequences. Fully Convolutional Network (FCN) [98] is used to learn spatial features and to produce the pixel dense prediction, but Gated Recurrent Unit (GRU) is employed to model temporal features before deconvolution is applied.

3.2 Localization and 3D Reconstruction

Localization and 3D reconstruction refer to the estimation of relative camera pose (translation and rotation) and the 3D structure of the observed environment from multiple images. Standard visual SLAM achieves this by leveraging feature correspondences. Let $\{x_{1j}, x_{2j}\}_{j=1}^{p} \in \mathbb{P}^2$ be a set of feature correspondences in the first and the second image, where p is the total number of points. Visual SLAM estimates the camera pose containing a translation vector $t \in \mathbb{R}^3$ and rotation matrix $R \in SO(3)$ and the 3D structure of all features $\{X_j\}_{j=1}^{p} \in \mathbb{P}^3$ by implementing epipolar geometry [59] on the feature correspondences. In robust visual SLAM, instead of computing the camera pose and 3D structure from all feature correspondences, only static features resulting from techniques described in Section 3.1 are employed. All dynamic features are regarded as outliers and excluded from the computation. On the other hand, deep learning techniques can process the image sequences directly without computing feature correspondences. This section discusses both feature-based and deep-learning-based approaches for solving the localization and 3D reconstruction problem.

3.2.1 Feature-Based Approaches. In feature-based visual SLAM, salient features are extracted to solve the image correspondence problem. The computer vision community has developed a large number of feature extraction techniques. While early work in SfM [157] including the prominent "Visual Odometry" [117] made use of the Harris corner detector [57], most recent work [142, 174] employs robust feature detection techniques such as Scale Invariant Feature Transform (SIFT) [99] or its lightweight variants like Speeded Up Robust Features (SURF) [8]. However, since SIFT and SURF are considered computationally expensive, a faster approach such as Features from Accelerated Segment Test (FAST) [130] is utilized for real-time applications [76, 94].

To find correspondences, extracted features are matched using feature-matching techniques. The techniques can be divided by how far the distance between the optical centers of two cameras (termed *baseline/parallax*) are separated. For short baselines, optical flow-based techniques (e.g., Kanade-Lucas-Tomashi (KLT) tracker [100]) can be used for matching. On the contrary, for long baselines, highly discriminative feature descriptors (e.g., SIFT [99], SURF [8], BRIEF [17], BRISK [91], etc.) are necessary to find correspondences by calculating dissimilarity between those descriptors. Unfortunately, using these feature-matching techniques does not guarantee perfect correspondences, especially when the data contains outliers. Implementation of robust estimators (e.g., RANSAC [37], PROSAC [22], MLESAC [158], etc.) is useful to reject outliers and handle false correspondences.

If the image correspondences are known, the relative pose between two or three images can be recovered up to a scale factor. By enforcing the epipolar constraint, the pose from two views can be computed by the 8-point [53] or the 5-point algorithm [116], while the trifocal tensor [156] can be utilized if three views are available. In case some 3D points of the scene have been reconstructed, camera poses can be obtained with respect to the 3D model by solving the perspective-n-point problems (e.g., P3P algorithm [42]).

When the camera pose is recovered, one can easily reconstruct 3D points of the scene by intersecting two projection ray lines through triangulation. As the rays do not always intersect due to erroneous correspondences, the midpoint method [9] or least-square-based method [60] is proposed to estimate the intersection. Then, to avoid the drifting problem, bundle adjustment (BA) [175] is employed to refine both the camera pose and 3D points by minimizing reprojection errors. A variant of the Gauss-Newton method, namely, Levenberg-Marquardt (LM) optimization, is the prevalent method to jointly optimize the structure of the scene and the motion of the camera.

In practice, there are some variations on how to implement feature-based visual SLAM. Instead of optimizing the camera pose and 3D structure of the environment over all images, Mouragnon et al. [110, 111] propose to optimize the last few images by employing local bundle adjustment (LBA). Klein and Murray [75] introduce "PTAM," which shows that tracking and mapping can run in real time if the pipeline is executed on different threads. Furthermore, PTAM also introduced the idea of choosing key frames, and thus LBA can also be implemented over the selected key frames. On the other hand, Lim et al. [94] used binary descriptors and a metric topological mapping such that large-scale mapping can operate in real time without any parallel computation. Recent state-of-the-art techniques like ORB-SLAM [113] integrate hardware and algorithmic advancement in the past decade by including parallel computing, ORB features [131], statistical model selection [155], loop closures based on bag-of-words place recognition [26, 41], local bundle adjustment [111], and graph optimization [81]. For a more detailed review of ORB-SLAM or other standard feature-based techniques, interested readers can follow [40] or [180].

3.2.2 Deep Learning for Pose Estimation and 3D Reconstruction. Recent developments on deep learning show that pose estimation can be regarded as a learning problem. While many end-to-end architectures for ego-motion computation have emerged [103, 171], there is no end-to-end learning for 3D reconstruction yet. Most recent works only stop the learning process at depth prediction [168, 187], although the resulting depth data can be used to reconstruct the 3D environment using point-based fusion as seen in [85].

There are two common methods for training pose estimation found in the existing literature, namely, supervised learning and unsupervised learning.

1) Supervised Learning. Supervised learning trains CNNs by minimizing errors in predicting the ego-motion compared to the ground-truth pose. As CNN is best known for classification tasks, in early works, pose estimation is considered as a classification problem over the discretized space of translation and rotation of the camera. Konda and Memisevic [78] were probably the first to propose the estimation of visual odometry using this principle. They utilized a stereo camera to predict the velocity and the direction of the camera. The network trains the representation of motion and depth from stereo pairs by using synchrony autoencoders [77]. These motion and depth representations are fed into a CNN to estimate the velocities and orientations through softmax-based classification. Instead of estimating general motion similar to fundamental matrix, DeTone et al. [32] proposed "HomographyNet" to train a CNN for computing homography between two frames using 4-point parameterization of homography. They proposed two different networks: one is a classification network based on cross-entropy loss function and the other one is a regression network based on Euclidean loss function. They showed that the regression network is more accurate than the classification network due to its continuous nature of the prediction.

After realizing that CNNs can be used accurately for the regression problem, all recent techniques for pose estimation employ regression-based CNN. Mohanty et al. [105] utilized a pretrained AlexNet network [79] for the input of the regression network. Two consecutive images are fed into two parallel AlexNet networks and then the outputs are concatenated for regressing the camera odometry through the fully connected layer. Based on the experiments, they observed that the extracted features from AlexNet are not generic for the problem of visual odometry, and thus the odometry only works well in a known environment.

Since pretrained convolutional layers for object detection and classification are not suitable for odometry estimation, researchers turned to optical flow-based networks to generalize the learned parameters in different environments. Muller and Savakis [112] designed "Flowdometry," a network consisting of two sequential CNNs: the first one for predicting optical flow and the latter for estimating camera motion. FlowNetS [33] architecture is used for both networks, although the second network replaces the refinement part by a fully connected layer in order to incorporate interframe odometry computation. Melekhov et al. [103] developed an end-to-end CNN for computing ego-motion between two views. They stacked two parallel CNNs with weight sharing followed by a spatial pyramid pooling (SPP) layer to tackle arbitrary input images while maintaining spatial information in the feature maps. The regression layer consists of two fully connected layers for predicting camera translation and rotation.

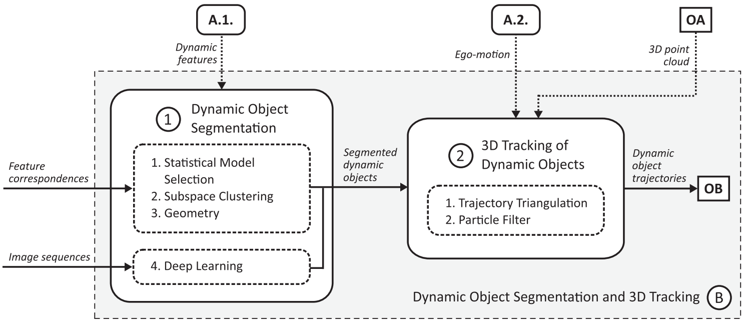
While the previous works only learn geometric feature representation of the scene through CNNs, Wang et al. [171] propose "DeepVO" as an end-to-end learning framework capable of learning sequential motion dynamics from image sequences through a Recurrent Convolutional Neural Network (RCNN), a combination of CNN and Recurrent Neural Network (RNN). RNNs are prominent for learning sequential data such as speech or language since they maintain a history of all elements of the sequence in the network [88]. It turns out that by utilizing both CNN and RNN, the output odometry is much better and has competitive performance over the state-of-the-art methods (compared to VISO2 Monocular and Stereo system [45]). Nonetheless, they stated that the moving objects in front of the camera might reduce the accuracy of pose estimation, but it is unclear how to deal with it under a deep learning framework.

2) Unsupervised Learning. In the unsupervised case, the CNN is trained without the availability of ground-truth data. Instead, the network learns to predict the camera pose by minimizing the photometric error similar to LSD-SLAM [36]. Given I_{ref} as a reference image where $I : \Omega \rightarrow \mathbb{R}$ provides the color intensity, the photometric error minimizes the following objective function:

$$E(\xi) = \sum_{i \in \Omega_{ref}} (I_{ref}(x_i) - I_{new}(\omega(x_i, D_{ref}(x_i), \xi)))^2, \tag{1}$$

where $\omega(x_i, D_{ref}(x_i), \xi)$ is a warp function that projects the image point $x_i \in \Omega_{ref}$ in the reference image I_{ref} to the respective point in the new image I_{new} based on the inverse depth value of the reference image $D_{ref}(x_i)$ and the camera transformation $\xi \in se(3)$.

Zhou et al. [187] developed this unsupervised learning mechanism using the principle of novel view synthesis (the problem of synthesizing a target image with different poses given a source image). They constructed two parallel CNN networks for predicting depth and estimating the camera pose. The predicted depth from the source image is used for synthesizing the target image given the camera transformation matrix and the source image. By minimizing the photometric error as in Equation (1), depth and camera pose can be jointly trained. Instead of generating the target image from depth prediction, Vijayanarasimhan et al. [168] constructed a 3D scene flow based on depth prediction, camera motion, and dynamic object segmentation resulting from the


A.1.
Dynamic features
A.2.
Ego-motion
OA
3D point cloud
Feature correspondences
Image sequences
1 Dynamic Object Segmentation
1. Statistical Model Selection
2. Subspace Clustering
3. Geometry
4. Deep Learning
Segmented dynamic objects
2 3D Tracking of Dynamic Objects
1. Trajectory Triangulation
2. Particle Filter
Dynamic object trajectories
OB
Dynamic Object Segmentation and 3D Tracking B

Index	References
B.1.1.	[122], [132], [133], [138]-[141], [153], [155]
B.1.2.	[20], [34], [35], [49], [72], [97], [128], [161], [164], [165], [177], [179], [185]
B.1.3.	[163], [166], [167]
B.1.4.	[15], [168]
B.2.1.	[5], [6], [69], [70], [121], [124], [125], [144], [186]
B.2.2.	[83]

Fig. 4. The flow diagram of the second application, dynamic object segmentation, and 3D tracking. Solid rounded rectangles indicate an action and dashed rounded rectangles show existing approaches for a specific action module. A square box shows the output of a particular module. Solid arrows denote data transfer, and dashed arrows reflect an optional input. The table on the right side shows the list of relevant literature references for each approach.

convolutional/deconvolutional network. The scene flow is transformed by the camera motion and then back-projected to the current frame for evaluating the photometric error.

4 DYNAMIC OBJECT SEGMENTATION AND 3D TRACKING

Dynamic object segmentation and 3D tracking clusters feature correspondences into different groups based on their motion and tracks their trajectories in 3D. Figure 4 shows the flow diagram of the existing approaches and the corresponding literature references in dynamic object segmentation and 3D tracking. It can be seen that the input of feature-based techniques for dynamic object segmentation consists of either full features or dynamic features only (obtained from action module A.1). On the other hand, the deep-learning-based approach can process the image sequences directly. The segmented dynamic objects are then fed into the 3D tracking module to obtain the object trajectories. Camera ego-motion and a 3D point cloud obtained from action module A.2 can be optionally utilized to help the tracking process. The availability of the 3D point cloud can make the output object trajectories consistent with the static world. This section discusses techniques for segmenting and tracking the dynamic objects in the scene.

4.1 Dynamic Object Segmentation

Dynamic object segmentation (also known as *multibody motion segmentation* [73, 132, 153] or *eorumotion segmentation* [133]) clusters all feature correspondences into n number of different object motions. It is considered a difficult problem due to the chicken-and-egg characteristic of the problem. In order to estimate the motion of the object, the features should be clustered first; on the other hand, the motion models for all moving objects are required to cluster the features. The problem is compounded by the presence of noise, outliers, or missing feature correspondences due to occlusion, motion blur, or losing tracked features. Another challenge is to deal with degenerate motion (e.g., when an object moves on the same plane and the same direction and velocity with the camera motion) or dependent motion (e.g., two people moving together, articulated motion). This section discusses existing approaches to handling this problem (see Table 2 for the summary).

4.1.1 Statistical Model Selection. In a static scene, the transformation of the feature points between consecutive images can be described by one motion model. In contrast, the feature points in dynamic scenes might have arisen from more than one motion model, each associated with

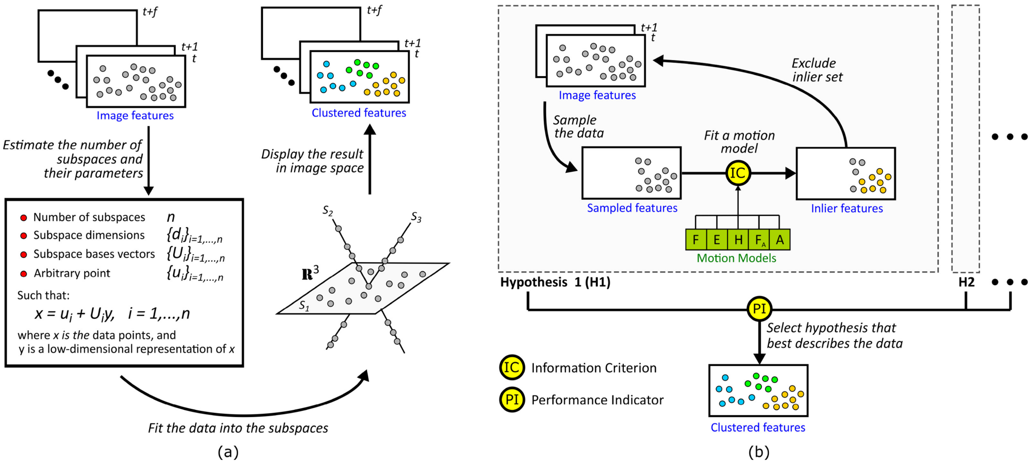

t+f
t+1
t
Image features
Estimate the number of subspaces and their parameters
Number of subspaces n
Subspace dimensions $\{d_i\}_{i=1,\dots,n}$
Subspace bases vectors $\{U_i\}_{i=1,\dots,n}$
Arbitrary point $\{u_i\}_{i=1,\dots,n}$
Such that:
$x = u_i + U_i y, \quad i = 1,\dots,n$
where x is the data points, and
y is a low-dimensional representation of x
Fit the data into the subspaces
(a)
t+f
t+1
t
Clustered features
Display the result in image space
S_2
S_3
$\mathbb{R}^3$
S_1
t+1
t
Image features
Exclude inlier set
Sample the data
Fit a motion model
IC
Sampled features
Inlier features
F E H F_A A
Motion Models
Hypothesis 1 (H1)
H2
PI
Select hypothesis that best describes the data
IC Information Criterion
PI Performance Indicator
Clustered features
(b)

Fig. 5. Illustration of (a) subspace clustering and (b) statistical model selection technique for dynamic object segmentation.

a different body. Motion models can be based on one of the following categories: fundamental matrix (F), affine fundamental matrix (F_A), essential matrix (E), homography/projectivity (H), or affinity (A). The model selection problem tries to fit all possible motion models with the data and select the one that best fits the data. If the data can be described by several models like in a dynamic scene, many hypotheses are required to segment the data based on the motion models.

3D motion segmentation approaches based on statistical techniques sample a subset of the data and fit a motion model into the sampled data under RANSAC [37] or the Monte-Carlo sampling iteration [139]. The motion model is used to build an inlier set and excludes the remaining data as the outliers of the model. Then, sampling is conducted again for the remaining data (outliers of the previous model) to find and fit another model that best describes the remaining data. This process is repeated until all data can be described by n motion models or the remaining outliers are not sufficient to generate more motion models. This motion segmentation process can be repeated again from the beginning to generate many candidate hypotheses (see Figure 5(b)).

The method to determine which model is best to describe the data is based on an information criterion. Several information criteria exist in the literature. Akaike's information criterion (AIC) [2] selects the model that maximizes the likelihood function yet minimizes the number of estimated parameters to generate the model. The penalization in the number of parameters is based on the observation that the maximum likelihood estimation always selects the most general model as the best fit model [155]. An intuitive example is that the errors of any points with respect to a point are higher or equal to the errors with respect to a line; thus, a line is always selected as the best model to describe the data points. AIC tackles this drawback by balancing the tradeoff between the goodness of the fit with the complexity of the model. It has the following form:

$$AIC = (-2)log(L) + 2K, \tag{2}$$

where L is the log likelihood function and K is the number of parameters of the model. The likelihood function is generally estimated to maximize the likelihood of observing correspondences based on a particular distance metric such as reprojection error or Sampson distance approximation [59]. Then, AIC selects the model that has the minimum AIC score under the minimum AIC estimate (MAICE) procedure.

Despite its popularity, AIC does not have asymptotically consistent estimates and is prone to overfitting because it does not take into account the number of observations. Schwarz [143] proposes a revision using the Bayesian theorem termed Bayes Information Criterion (BIC). BIC extends the posterior probability of observing the data by modeling the prior based on its complexity. On the other hand, Rissanen [129] developed Minimum Description Length (MDL) by using a minimum-bit representation to minimize the coding length of the data. Based on the limitation of previous works, Kanatani [71, 72] proposed Geometric Information Criterion (G-AIC, or in some literature called GIC) by taking into account the number of observations and the dimension of the model; it has the following form:

$$GIC = (-2)log(L) + 2(DN + K), \tag{3}$$

where N is the number of data and D is the dimension of the model (e.g., two for a homography, three for a fundamental matrix). Another extension based on BIC is Geometrically Robust Information Criterion (GRIC) devised by Torr [155]. By incorporating robustness to outliers and the capability to deal with different dimensions, GRIC has the following form:

$$GRIC = (-2)log(L) + DNlog(R) + Klog(RN), \tag{4}$$

where R is the dimension of data.

There are different ways to implement statistical model selection for 3D motion segmentation. Torr [155] samples nearby feature correspondences and computes different motion models (F, F_A, H, A) under the RANSAC iteration. GRIC is used to select the best motion model that fits with a particular inlier cluster. However, Expectation-Maximization (EM) is applied when the number of inliers for the selected model is lower than a threshold. In order to avoid the expensive computation of brute-force sampling, Schindler and Suter [138, 139] propose local Monte-Carlo sampling by drawing samples from a defined subregion on the image. They present a method to estimate the noise scale from the data, thus allowing the residual distribution for each motion and its standard deviation to be recovered. Moreover, they derived a new likelihood function that allows the motion models (F, H) to overlap, while the best model is selected by GRIC as shown in Equation (4).

While the previous approaches operate on two image sequences, Schindler et al. [141] extended the technique in [138] to several perspective images under a general motion model (essential matrix E). In order to link several essential matrix candidates from more than two image sequences, temporal coherence is enforced by connecting only essential matrices with similar inlier sets. Finally, an MDL-like approach is utilized to select the best model that describes the motion. This method has been generalized for any camera model (not only perspective camera) and motion model (not only essential matrix E) by Schindler et al. [140]. Practical considerations have also been taken into account by Ozden et al. [122]. They handled how to merge a previously moving object with the background or how to split a cluster into two different motions.

Thakoor et al. [153] formulated the model selection problem as a combinatorial optimization. The branch-and-bound technique is employed to optimize the segmentation of motion using AIC as the cost function, by splitting the optimization problem into smaller subproblems. Local sampling of correspondences is also used to generate the motions, while the null hypothesis is introduced to handle outliers. Recently, Sabzevari and Scaramuzza [132] utilized a statistical model selection technique under factorization of the projective trajectory matrix framework. Epipolar geometry is used to generate the motion models, while reprojection error is employed to reject invalid hypotheses. The hypotheses are evaluated by iteratively refining the structure estimation and motion segmentation. This has been extended in [133] by enforcing ego-motion constraints such that the camera motion and the moving object motions can be computed by using the one-point algorithm [136, 137] and the two-point algorithm [119], respectively.

4.1.2 Subspace Clustering. Subspace clustering is developed based on the observation that many high-dimensional data can be represented by a union of low-dimensional subspaces. A subspace of data points can be represented by basis vectors and low-dimensional representation of the data. The problem of 3D motion segmentation under the subspace clustering framework is basically finding each individual subspace associated with each body motion and fitting the data into the subspaces (see Figure 5(a)). However, since the subspaces and the data segmentation are not known in practice, estimating the subspace parameters and clustering the data into different subspaces should be done simultaneously. This problem was originally pointed out by Costeira-Kanade [25] and Gear [44] based on the observation that independent rigid body motion lies in a linear subspace. By enforcing the rank constraint (see Section 5.1 for more details), each linear subspace can be recovered.

Kanatani [72] coined the term of *subspace separation* as a general method for clustering low-dimensional subspace (not only limited to motion segmentation). The subspace separation is done by borrowing the principle of statistical model selection, but a subspace is fitted instead of a motion model. AIC is used to select the best subspace configuration by balancing the increase of the residual when data points are fitted to a subspace and the decrease of the degree of freedom when merging two subspaces into one group. Least median of squares is employed to fit the data points that contain outliers. Differently, Vidal et al. [164, 165] proposed Generalized Principal Component Analysis (GPCA) as an extension of PCA. While PCA only works for data lying in a linear subspace, GPCA generalizes the problem into data points arising from multiple linear subspaces. In GPCA, the problem of finding subspaces is done by fitting of the homogeneous polynomial of degree n into the data through polynomial embedding (or Veronese map) and finding the normals of each subspace by computing the derivatives of the polynomial at a particular point. Then, the segmentation is obtained by computing the similarity matrix from the angle between the normal vectors and clustering it using spectral clustering. For practical consideration in motion segmentation, GPCA is extended in [165] by projecting the data into a lower-dimensional space before clustering is executed. Then, the number of motions n can be computed by finding the rank of the polynomial embedding.

While the previous works assumed that the motions are rigid, Yan and Pollefeys [177] proposed a general framework called Local Subspace Affinity (LSA) for independent, articulated, rigid, nonrigid, degenerate, and nondegenerate motions. LSA estimates a subspace by sampling a point and its nearest neighbors and fitting a local subspace to the sampled data. The nearest neighbors can be found by computing the angles or the distance between the vectors. Then, an affinity matrix is computed as the principal angles between two local subspaces and the clustering is done by applying spectral clustering to the affinity matrix. Projection into a lower-dimensional subspace is also carried out before the subspace is estimated. Similar to LSA, Goh and Vidal [49] also fit a local subspace to a point and its nearest neighbors. The method, known as Locally Linear Manifold Clustering (LLMC), is developed based on the Locally Linear Embedding (LLE) [135] algorithm. They cluster separated manifolds associated with each motion by transforming the data into low-dimensional representation using LLE and computing the null space of the matrix resulting from LLE. They showed that the segmentation of the data is indicated by the vectors in the null space.

Another point of view is given by Elhamifar and Vidal [34, 35] that leverages a sparse representation to cluster motion. They propose Sparse Subspace Clustering (SSC) based on the observation that a point in a union of linear or affine subspaces can be represented as a linear or affine combination of all data points in the subspaces. However, the sparsest representation is only obtained when the point is written as a linear or affine combination of the data lying in the same subspace. Under noiseless data, the sparsest coefficient can be estimated by solving the L_1 minimization problem. Given the sparsest coefficient, an affinity matrix can be built and the clustering can be done by spectral clustering. An extension of SSC is developed by Rao et al. [128]. They fused sparse representation and data compression to deal with practical issues such as when the data is missing, is incomplete, or contains outliers. Recently, Yang et al. [179] also improved SSC by proposing various matrix completion techniques for data with missing entries. Instead of using sparse representation, Liu et al. [97] and Chen et al. [20] employ Low-Rank Representation (LRR), which can also be used to define the affinity matrix for subspace segmentation using spectral clustering.

It is worth noting that most subspace clustering techniques operate in batch mode. Vidal [161] devised an iterative clustering technique for data lying in multiple moving hyperplanes. He modeled the union of moving hyperplanes by a set of time-varying polynomials. The segmentation is done recursively by estimating the normal vector of the hyperplanes within the normalized gradient descent framework. Another implementation of online subspace clustering was proposed by Zhang et al. [185]. They modified the K-flats algorithm such that it can take the input data incrementally. L_1 is used as the objective function instead of L_2 in order to boost its performance under noise and data containing outliers.

In past decades, subspace clustering has become a widely studied topic, and many approaches have been developed by diverse research communities. There are several survey papers related to subspace clustering, from general techniques to those focusing on the application of motion segmentation and face clustering. For a more detailed review of subspace clustering, interested readers can follow [162].

4.1.3 Geometry. Geometry approaches extend the standard formulation of geometry of multiple views from static scenes to dynamic scenes containing independent moving objects. While there is one fundamental matrix that describes general motion of the camera with respect to the static scene, in a dynamic environment, there will be n fundamental matrices that describe the motion of n bodies, including one for static features. Vidal et al. [166] study a generalization of this problem by proposing *multibody epipolar constraints*. Given x_1 and x_2 as feature correspondence between the first and the second image, respectively, $x_2^T F x_1 = 0$ represents the epipolar constraint for the static scene, where $F \in \mathbb{R}^{3\times 3}$ is the fundamental matrix (see Figure 3(a)). If the scene contains n independent moving objects, there are a set of fundamental matrices $\{F_i\}_{i=1}^n$ associated with each moving object such that the following multibody epipolar constraint is satisfied [167]:

$$\varepsilon(x_1, x_2) \doteq \prod_{i=1}^{n} (x_2^T F_i x_1) = 0. \tag{5}$$

This multibody epipolar constraint transforms the standard epipolar constraint equation from a bilinear to a homogeneous polynomial of degree n (in x_1 and x_2). This homogeneous polynomial equation can be converted into the bilinear problem again by mapping the polynomial equation into a vector containing M_n monomials using the veronese map $\upsilon_n : \mathbb{R}^3 \rightarrow \mathbb{R}^{M_n}$, where $M_n \doteq \binom{n+2}{n}$. Thus, the multibody epipolar constraint in Equation (5) can be transformed into

$$\upsilon_n(x_2)^T \tilde{F} \upsilon_n(x_1) = 0, \tag{6}$$

where $\tilde{F}$ is the *multibody fundamental matrix*, a symmetric tensor product representation of all fundamental matrices [166, 167]. If n is known, by reordering the entries of $\upsilon_n(x_1)$ and $\upsilon_n(x_2)$ using the Kronecker product and stacking the row of $\tilde{F}$ into $f \in \mathbb{R}^{M_n^2}$, Equation (6) can be transformed into a linear equation in f and can be estimated by least squares. Individual fundamental matrices F_i can then be recovered by finding the epipolar line associated with each motion through

Table 2. Summary of Existing Approaches on Dynamic Object Segmentation

	Camera[a]				Motion[b]			Practical Consideration[d]							
Author(s)	**T**	**ST**	**M**	**SQ**	**N**	**T**	**AC**[c]	**RT**	**SO**	**NP**	**ND**	**OT**	**MD**	**DP**	**DG**
Statistical Model Selection (Section 4.1.1)															
Torr [155]	M	M	A,P	S	M	R	-	FO	✓	✓	✓	✓	-	-	✓
Schindler et al. [138, 139]	M	M	P	S	M	R	s:4.6	FO	✓	✓	✓	✓	-	✓	✓
Schindler et al. [141]	M	M	P	M	M	R	s:2.5	FO	✓	✓	✓	✓	✓	-	-
Schindler et al. [140]	M	M	P	M	M	R,A	r:6.1	FO	✓	✓	✓	✓	✓	✓	✓
Thakoor et al. [153]	M	M	P	M	M	R	s:5	NT	✓	✓	-	✓	-	✓	✓
Ozden et al. [122]	M	M	P	M	M	R	-	FO	✓	✓	✓	✓	✓	✓	✓
Sabzevari et al. [132]	M	M	P	M	M	R	h:0.35	NT	✓	✓	✓	✓	-	-	-
Sabzevari et al. [133]	M	M	P	M	M	R	h:0.11	NT	✓	-	✓	✓	-	-	-
Subspace Clustering (Section 4.1.2)															
Kanatani [72]	M	M	A	S	M	R	-	FO	-	✓	✓	-	-	-	-
Vidal et al. [164]	M	M	A	M	M	R	h:19.8	FO	-	✓	✓	✓	-	✓	-
Vidal et al. [165]	M	S,M	A	M	M	R	h:19.83	FO	-	✓	✓	✓	✓	✓	-
Yan et al. [177]	M	S,M	A	M	M	R,N,A	h:25.07	FO	✓	✓	✓	-	✓	✓	✓
Vidal et al. [161]	M	M	A	M	S	R	s:4	NT	✓	✓	-	-	-	-	-
Goh et al. [49]	M	M	A,P	M	M	R	h:5.62	FO	-	✓	✓	✓	-	✓	✓
Zhang et al. [185]	M	M	A	M	M	R	h:12.29	NT	✓	-	✓	✓	-	-	-
Rao et al. [128]	M	M	A	M	M	R,N,A	h:3.37	FO	-	✓	✓	✓	✓	✓	-
Elhamifar et al. [34, 35]	M	M	A	M	M	R	h:0.52	FO	-	✓	✓	✓	✓	✓	-
Liu et al. [97]	M	M	A	M	M	R	h:1.71	FO	-	✓	✓	✓	✓	✓	-
Chen et al. [20]	M	M	A	M	M	R	h:2.69	FO	-	✓	✓	✓	✓	✓	-
Yang et al. [179]	M	M	A	M	M	R	h:0.06	FO	-	✓	✓	✓	✓	✓	-
Geometry (Section 4.1.3)															
Vidal et al. [166, 167]	M	M	P	S	M	R	r:5.88	NB	✓	-	✓	-	-	-	-
Vidal et al. [163]	M	M	P	S	M	R	r:8	NB	✓	-	✓	✓	-	-	-
Deep Learning (Section 4.1.4)															
Vijayanarasimhan et al. [168]	M	M	P	M	M	R	-	NT	✓	-	-	I	-	-	-
Byravan et al. [15]	D	S	P	L	M	R	-	NT	✓	-	✓	I	-	-	-

[a] Camera Type (**T**): Monocular (**M**), Stereo (**S**), Depth (**D**). Camera State (**ST**): Static (**S**), Moving (**M**). Camera Model (**M**): Orthography (**O**), Affine (**A**), Perspective (**P**). Camera Sequences (**SQ**): Short (**S**, $f < 11$), Medium (**M**, $10 < f < 501$), Long (**L**, $f > 500$), where f is the number of images.

[b] Number of Motions (**N**): Single (**S**), Multiple (**M**). Motion Type (**T**): Rigid (**R**), Nonrigid (**N**), Articulated (**A**).

[c] **AC**: Accuracy defined by the percentage of segmentation errors. **s**: Evaluated on synthetic data, **r**: Evaluated on real data, **h**: Evaluated on Hopkins 155 dataset [159]. Only the results from three motion sequences or the overall mean are displayed.

[d] **CT**: Computation Time (**RT**: Real time, **NT**: Near real time, **NB**: Need to be batched, **FO**: Fully offline), **SO**: Support sequential operation, **NP**: No prior knowledge (e.g., number and dimension of the moving objects), **ND**: Handle noise in data, **OT**: Handle outliers due to false feature correspondences (**I**: irrelevant for the technique), **MD**: Handle missing data (e.g., due to occlusion, lost tracks, motion blur), **DP**: Support dependent motion, **DG**: Handle degenerate motion.

polynomial factorization of multibody epipolar line $\tilde{l} \doteq \tilde{F}v_n(x_1) \in \mathbb{R}^{M_n}$. Subsequently, the motion segmentation of dynamic features can be done by assigning each feature correspondence with the correct fundamental matrix [167].

Vidal and Hartley [163] extended the multibody SfM formulation from two views into three views by introducing the multibody trilinear constraint and multibody trifocal tensor. It is the generalization of the trilinear constraint and trifocal tensor [59, 156] from a static scene to a dynamic scene containing multiple objects. The multibody trifocal tensor can be solved linearly by embedding the feature correspondences as in Equation (6) and estimating using least squares. Each trifocal tensor corresponding to each object is recovered from the multibody trilinear constraint by computing its second-order derivative.

4.1.4 Deep Learning for Dynamic Object Segmentation. Current works of DNNs for solving the dynamic object segmentation problem rely on a predefined number of rigid body motions. The network and its associated cost function to produce dense object masks might be derived from 3D point cloud data or optical flow. Byravan and Fox [15] introduce "SE3-Net" as a DNN that is capable of segmenting predefined n dynamic objects represented in $SE(3)$ transforms from a 3D point cloud. A convolutional/deconvolutional encoder-decoder network is constructed to predict object masks and a rigid body transformation for each object. The encoder consists of two parallel convolutional and fully connected networks that produce latent variables from the point cloud and encode the control vector, respectively. The decoder processes the concatenated output from the encoder to produce pointwise object masks and $SE(3)$ transformation through two parallel deconvolutional and fully connected networks. A transform layer is used to fuse the 3D point cloud data, the object masks, and their $SE(3)$s to generate a predicted point cloud for data training.

Vijayanarasimhan et al. [168] have shown the utilization of optical flow for segmenting dynamic objects using DNN. They designed a network termed "SfM-Net," a geometry-aware network capable of predicting depths, camera motion, and dynamic object segmentation. The networks consist of two stream convolutional/deconvolutional subnetworks, acting as structure and motion networks. The structure network learns to predict depth, while the motion network estimates camera and object motion. While the object motion is computed by two fully connected layers on top of the embedding layer produced by CNN, the dynamic object segmentation is predicted by feeding the embedding layer to the deconvolutional network. The outputs from both structure and motion networks are then converted into optical flow by transforming the point cloud from depth prediction according to camera and object motion, followed by reprojecting the transformed point cloud into the image space. By using this technique, the network can be trained by self-supervision through minimizing photometric error as in Equation (1), although full supervised learning is also possible.

4.2 3D Tracking of Dynamic Objects

The problem of tracking dynamic objects in 3D, knowing the position of the moving object in 3D coordinates, including depth information, is substantial. The challenge is that the standard approach in visual SLAM for estimating the 3D structure of the scene, which is triangulation [60], does not work for dynamic objects since the rays back-projected from the corresponding feature points do not meet. Given x_1 and x_2 as the feature correspondences from the first and the second image, respectively, the corresponding 3D point X should be able to be computed by intersecting the back-projected rays of x_1 and x_2 via their associated camera projection matrix P_1 and P_2. Since the object has independent motion (from camera motion), the projection rays from the first to the second frame are also moving, and thus do not intersect (see Figure 3). Alternative techniques are required to solve this problem. This section discusses existing approaches for recovering 3D trajectories of the objects moving in front of camera (see Table 3 for the summary).

4.2.1 Trajectory Triangulation. Standard triangulation [60] cannot be used to reconstruct the 3D structure of the moving objects since the back-projected rays do not intersect. Avidan and Shashua [5, 6] coined the term *trajectory triangulation* as a technique to reconstruct 3D points of the moving object when the object trajectory is known or satisfies a parametric form. They assumed that the 3D point is moving along an unknown 3D line. Then, the reconstruction problem

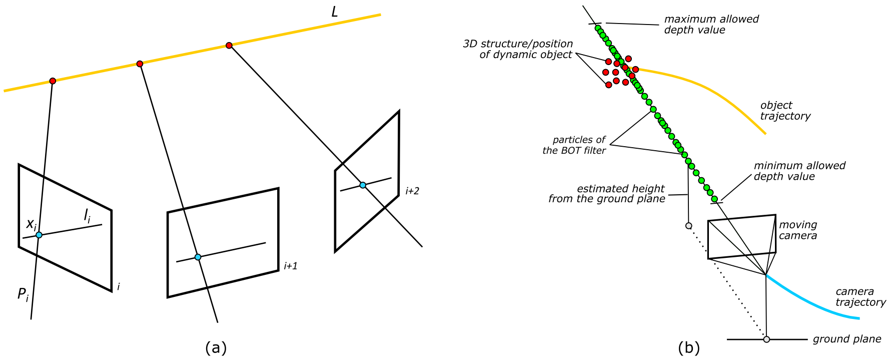

L
x_i
l_i
P_i
i
i+1
i+2
(a)
maximum allowed depth value
3D structure/position of dynamic object
object trajectory
particles of the BOT filter
minimum allowed depth value
estimated height from the ground plane
moving camera
camera trajectory
ground plane
(b)

Fig. 6. (a) Illustration of a moving point along the line L and the projection of the point and the line to the image plane i by the projection matrix P_i. (b) Illustration of particle filter technique for tracking dynamic object. The particles are spread along the ray of projection and are constrained by the estimated/predefined ground plane and maximum/minimum allowed depth value.

is turned into the problem of finding a 3D line that intersects projected rays from t views. In order to have a unique solution, at least $t = 5$ is required since the set of intersecting lines from three views will form a quadric surface that makes the ray from the fourth view intersect at two points. Thus, five views result in a unique solution.

Specifically, let $A = [1, X_A, Y_A, Z_A] \in \mathbb{P}^3$ and $B = [1, X_B, Y_B, Z_B] \in \mathbb{P}^3$ be the 3D points on the line L represented in homogeneous coordinates. If x_i and l_i are the projection of the 3D point and the line L on frame i, respectively, it is clear that

$$x_i^T l_i = 0, \tag{7}$$

since x_i lies on l_i (see Figure 6(a)). Line L can be represented in a Plucker coordinate as follows:

$$\tilde{L} = A \wedge B = [X_A - X_B, Y_A - Y_B, Z_A - Z_B, X_A Y_B - Y_A X_B, X_A Z_B - Z_A X_B, Y_A Z_B - Z_A Y_B]. \tag{8}$$

By using Plucker representation, projection matrix P_i can be transformed into a 3×6 matrix $\tilde{P}_i$ such that $li \cong \tilde{P}_i \tilde{L}$, where

$$\tilde{P}_i = \begin{bmatrix} P^2 \wedge P^3 \\ P^3 \wedge P^1 \\ P^1 \wedge P^2 \end{bmatrix}, \tag{9}$$

and P^k represents the kth row of projection matrix P_i. Subsequently, Equation (7) becomes the following equation, which is linear in $\tilde{L}$:

$$x_i^T \tilde{P}_i \tilde{L} = 0. \tag{10}$$

By stacking Equation (10) from five frames, $\tilde{L}$ can be estimated by least squares. Finally, each moving 3D point on the line $\tilde{L}$ can be found by the intersecting ray from each frame with the line $\tilde{L}$ [5].

Instead of assuming that the object is moving along a line, Shashua et al. [144] assumed that the object is moving over a conic section. Nine views are required to get a unique solution, although seven views are adequate if the type of conic is known, such as a circle in 3D Euclidean space. They solved the nonlinear optimization problem by fitting a random conic to the moving points in 2D space or by minimizing the error of estimated conic radius in 3D such that the a priori constraint can be enforced. Based on previous works, Kaminski and Teicher [69, 70] generalized trajectory triangulation by representing a curve as a family of hypersurfaces in the projective space. This polynomial representation transforms the nonlinear trajectory problem into a linear problem in the unknown parameters. On the other hand, to handle missing data, Park et al. [124] represented a 3D trajectory as a linear combination of trajectory basis vectors such that the recovery of 3D points can be estimated robustly using least squares. They also proposed *reconstructability* criteria by analyzing the relationship among ego-motion, point motion, and trajectory basis vectors. Since reconstructability is inversely proportional to 3D reconstruction error, this criterion allows precise inspection of the possibility of accurate reconstruction [125].

4.2.2 Particle Filter. Due to the *observability* issue (the distance between the observer and the target cannot be observed), the problem of tracking moving objects in 3D using monocular cameras can be seen as the Bearing-only-Tracking (BOT) problem. A monocular camera can be viewed as a BOT sensor since it can only provide bearing information of the tracked feature points (e.g., the angle between observed features in the previous and the current frame with respect to camera center) on the moving object. A filter-based approach is preferable for the BOT problem since it can model the uncertainty of the position and velocity of the observer and the target and has been studied widely as a target motion analysis problem [1, 16].

Kundu et al. [83] employed particle filters to estimate the position and velocity of the moving objects. Instantaneous constant velocity motion model and Lie algebra are used to model the unknown motion and parameterize the rigid transformation of the objects, respectively. In initialization, the moving object is segmented by geometric constraints and Flow Vector Bound (FVB) as in [84] and [82] and the particles are spread uniformly along the ray of projection. An estimated ground plane from the 3D point cloud of the static scene and the maximum allowed depth value are leveraged to constrain the space of the particles (see Figure 6(b)). For importance sampling, the weight of the particle is updated by projecting each particle into the current frame and computing the projection error compared to the actual feature position. As particles with lower error or higher weight have a higher probability to be resampled, they concentrate on the depth value that gives the smallest reprojection error.

5 JOINT MOTION SEGMENTATION AND RECONSTRUCTION

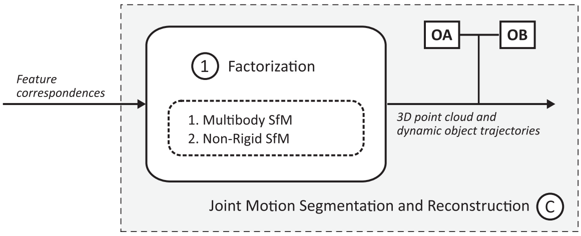
Instead of performing multibody motion segmentation and reconstructing the 3D structure of dynamic objects as a separate and sequential task, factorization can do both simultaneously. Given the feature correspondences, dynamic object segmentation and reconstruction produce the motion of the segmented features as well as their 3D structures. Figure 7 describes the flow of this joint motion segmentation and reconstruction task. Although factorization can produce both segmented objects and their 3D structures, generally, the output from applications A (OA) and B (OB) can be combined to have a similar result as this technique.

5.1 Factorization

Factorization is probably one of the most prominent techniques in SfM. It has an elegant mathematical formulation and can solve the problem of segmentation and reconstruction simultaneously. It was first formulated by Tomasi and Kanade [154] based on the rank theorem in 1992. The theorem states that in short sequences of static scenes, a measurement matrix, a matrix containing all tracked feature points through all frames, is at most of rank four (or rank three if using the orthographic projection model under Euclidean coordinates) [24].

Specifically, let $W \in \mathbb{R}^{2f \times p}$ be a measurement matrix where f is the number of frames and p is the number feature points. W can be factorized into motion matrix $M \in \mathbb{R}^{2f \times 4}$ and shape matrix $S \in \mathbb{R}^{4 \times p}$ such that

$$W = MS$$


Feature correspondences
1 Factorization
1. Multibody SfM
2. Non-Rigid SfM
OA
OB
3D point cloud and dynamic object trajectories
Joint Motion Segmentation and Reconstruction C

Index	References
C.1.1.	[24], [25], [44], [54], [80], [93], [182]
C.1.2.	[3], [27], [65], [80], [123], [176], [178]

Fig. 7. The flow diagram of the third application, joint motion segmentation and reconstruction. Solid rounded rectangles indicate an action and dashed rounded rectangles show existing approaches for a specific action module. A square box shows the output of a particular module. Solid arrows denote data transfer, and dashed arrows reflect an optional input. The table at the bottom shows the list of relevant literature references for each approach.

$$\begin{bmatrix} u_{11} & \cdots & u_{1p} \\ \vdots & \ddots & \vdots \\ u_{f1} & \cdots & u_{fp} \\ v_{11} & \cdots & v_{1p} \\ \vdots & \ddots & \vdots \\ v_{f1} & \cdots & v_{fp} \end{bmatrix} = \left[\begin{array}{c|c} i_1^T & t_{x_1} \\ \vdots & \vdots \\ i_f^T & t_{x_f} \\ j_1^T & t_{y_1} \\ \vdots & \vdots \\ j_f^T & t_{y_f} \end{array}\right] \Big[\, S_1, \ldots, S_p \,\Big], \tag{11}$$

where (u_{fp}, v_{fp}) are the position of feature points in the image space; i_f^T and j_f^T are the first and the second row of rotation matrix $R \in SO(3)$, respectively; and (t_{x_f}, t_{y_f}) are the coordinates of the translation vector in x and y directions. Exploiting the rank constraint, W can be decomposed using Singular Value Decomposition (SVD) such that

$$W = U'\Sigma'V'^T, \tag{12}$$

where $\Sigma' \in \mathbb{R}^{4\times4}$ is a diagonal matrix containing the four biggest eigenvalues and $U' \in \mathbb{R}^{2f\times4}$ and $V' \in \mathbb{R}^{p\times4}$ are eigenvectors corresponding to the four biggest eigenvalues. Subsequently, both motion and shape matrices are estimated as $\hat{M} \equiv U'\Sigma'^{1/2}$ and $\hat{S} \equiv \Sigma'^{1/2}V'^T$. Since the decomposition in Equation (12) is not unique, the exact value of M and S should be computed by finding matrix A such that $W = MS = (\hat{M}A)(A^{-1}\hat{S})$. Matrix A can be found by enforcing rotation and translation constraints and solving the resulting linear equation through least squares [25, 154].

This basic formulation of a static scene can be used to reconstruct a moving object in front of the camera as long as the scene is static. Nonetheless, it can also be extended to multibody formulation for a moving camera depending on the camera model (orthography, affine, or perspective) or the type of motion (rigid or nonrigid). If the scene contains n motions, then the columns of measurement matrix can be sorted such that

$$\bar{W} = W\Gamma = [\, W_1, \ldots, W_n \,], \tag{13}$$

where $\Gamma \in \mathbb{R}^{p\times p}$ is an unknown permutation matrix. Without noise, each W_i, where $i = \{1, 2, \ldots, n\}$, lies in a subspace of at most rank four [25]. Then, as each W_i can be factorized into a

Table 3. Summary of Existing Approaches on 3D Tracking of Dynamic Objects and Joint Motion Segmentation and Reconstruction

	Camera[a]				Motion[b]		Practical Considerations[c]						
Author(s)	**T**	**ST**	**M**	**SQ**	**N**	**T**	**RT**	**SO**	**CK**	**OK**	**ND**	**OT**	**MD**
Trajectory Triangulation (Section 4.2.1)													
Avidan et al. [5]	M	M	P	M	S	R	NB	✓	✓	-	-	-	-
Shashua et al. [144]	M	S,M	P	M	S	R	NB	✓	✓	-	-	-	-
Avidan et al. [6]	M	S,M	P	M	S	R	NB	✓	✓	-	-	-	-
Kaminski et al. [69, 70]	M	S,M	P	M	S	R	NB	✓	✓	✓	✓	-	-
Ozden et al. [121]	M	M	P	M	S	R	FO	✓	✓	-	✓	-	-
Park et al. [124]	M	M	P	M	M	R,N	FO	✓	✓	-	✓	-	✓
Zheng et al. [186]	M	M	P	M	M	R	FO	✓	✓	-	-	-	-
Park et al. [125]	M	M	P	M	M	R,N	FO	✓	✓	-	✓	-	✓
Particle Filter (Section 4.2.2)													
Kundu et al. [83]	M	M	P	M	M	R	RT	✓	-	-	✓	✓	-
Factorization (Section 5.1)													
Tomasi et al. [154]	M	S	O	S	S	R	FO	-	✓	-	✓	-	-
Morita et al. [106]	M	S	O	M	S	R	NT	✓	✓	-	✓	-	-
Costeira et al. [24]	M	S	O	M	M	R	FO	-	✓	-	-	-	-
Sturm et al. [151]	M	S	P	M	S	R	FO	-	✓	-	✓	-	-
Costeira et al. [25]	M	S,M	A	M	M	R	FO	-	✓	✓	✓	-	-
Gear [44]	M	S,M	O	M	M	R	FO	-	-	✓	✓	-	-
Ichimura [65]	M	S,M	A	M	S	R	FO	-	-	✓	✓	✓	-
Bregler et al. [13]	M	S	O	L	S	N	FO	-	✓	✓	-	-	-
Hartley et al. [58]	M	S	A,P	M	S	R	FO	-	✓	✓	-	-	✓
Xiao et al. [176]	M	M	A	M	S	N	FO	-	✓	✓	✓	-	-
Han et al. [54]	M	M	O,A	M	M	R	FO	-	✓	-	✓	-	-
Li et al. [93]	M	M	P	M	M	R	FO	-	✓	-	-	-	-
Akhter et al. [3]	M	M	O	L	S	N	FO	-	✓	✓	-	-	-
Yan et al. [178]	M	M	O,A	L	S	R,N,A	FO	-	✓	✓	✓	✓	-
Paladini et al. [123]	M	M	O	M	S	N,A	FO	-	✓	-	✓	-	✓
Murakami et al. [114]	M	M	P	M	S	R	FO	-	✓	-	-	-	-
Zappella et al. [182]	M	M	O	M	M	R	FO	-	✓	-	✓	✓	✓
Dai et al. [27]	M	S,M	O	M	S	N	FO	-	✓	✓	✓	-	-
Kumar et al. [80]	M	M	O	M	M	N	FO	-	✓	✓	✓	-	-

[a] Camera Type (**T**): Monocular (**M**), Stereo (**S**), RGB-D (**R**). Camera State (**ST**): Static (**S**), Moving (**M**). Camera Model (**M**): Orthography (**O**), Affine (**A**), Perspective (**P**). Camera Sequences (**SQ**): Short (**S**, $f < 11$), Medium (**M**, $10 < f < 501$), Long (**L**, $f > 500$), where f is the number of images.

[b] Number of Moving Objects (**N**): Single (**S**), Multiple (**M**). Motion Type (**T**): Rigid (**R**), Nonrigid (**N**), Articulated (**A**).

[c] **CT**: Computation Time (**RT**: Real time, **NT**: Near real time, **NB**: Need to be batched, **FO**: Fully offline), **SO**: Supports sequential operation, **CK**: No knowledge about the camera motion (e.g., trajectory, velocity), **OK**: No knowledge about the moving objects (e.g., number, dimension, rank, trajectory), **ND**: Handles noise in data (e.g., Gaussian noise), **OT**: Handles outliers (e.g., due to false correspondence), **MD**: Handles missing data (e.g., due to occlusion, lost tracks, motion blur).

motion and shape matrix, $\bar{W}$ has a canonical form as follows:

$$\bar{W} = \bar{M}\bar{S} = [\,M_1, \ldots, M_n\,] \begin{bmatrix} S_1 & & \\ & \ddots & \\ & & S_n \end{bmatrix}. \tag{14}$$

The problem of motion segmentation and reconstruction is then transformed into the problem of finding the correct permutation matrix Γ in Equation (13) such that matrix $\bar{S}$ has block diagonal. In general, the techniques addressing this problem can be divided based on the motion types, namely, rigid and nonrigid motion.

5.1.1 Multibody Structure from Motion (MBSfM). Multibody Structure from Motion (MBSfM) generalizes standard SfM for a rigid camera motion into n bodies of rigid motions. To solve the MBSfM problem, under the affine camera model, Costeira and Kanade [25] introduced the shape interaction matrix, a mathematical construct of object shapes that is invariant to object motions and coordinate systems selection. This shape interaction matrix was found to be preserving the original subspace structure. Let $\bar{W} = U\Sigma V^T$ be rank-r SVD decomposition of measurement matrix such that $U \in \mathbb{R}^{2f\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, and $V \in \mathbb{R}^{p\times r}$. The shape interaction matrix Q is defined as follows:

$$Q = VV^T \in \mathbb{R}^{p\times p}. \tag{15}$$

Equation (15) has an interesting property that the entry is zero if feature trajectory a and b belong to different objects. This property has been proved mathematically by Kanatani [72] as well. Based on this observation, motion segmentation and reconstruction can be done by sorting and thresholding the entries of Q.

Costeira and Kanade [25] cluster the structure by maximizing the sum-of-squares entries of a block diagonal subject to the constraint that each block represents a physical object. Ichimura [65] used a discriminant criterion [120] to separate the sorted rows of Q into different motions that maximize separation among subspaces. On the other hand, instead of clustering the subspace through SVD, Gear [44] showed that echelon canonical form provides direct information on the grouping of points to the subspaces.

For factorization using projective cameras, the problem is trickier since factorization cannot be done without first recovering an unknown scale factor $\lambda \in \mathbb{R}$ called projective depth. Let $\{X_j \in \mathbb{P}^3\}_{j=1}^{p}$ and $\{x_{ij} \in \mathbb{P}^2\}_{i=1,\ldots,f}^{j=1,\ldots,p}$ be a set of p 3D points and p feature points in f frames, both represented in homogeneous coordinates. If $\{P_i \in \mathbb{R}^{3\times 4}\}_{i=1}^{f}$ are a set of projection matrices that map all 3D points to feature points for every frame i, then the image projection equation is calculated as $\lambda_{ij}x_{ij} = P_iX_j$. The complete image projection matrices can be written as follows:

$$W = \begin{bmatrix} \lambda_{11}x_{11} & \cdots & \lambda_{1p}x_{1p} \\ \vdots & \ddots & \vdots \\ \lambda_{f1}x_{f1} & \cdots & \lambda_{fp}x_{fp} \end{bmatrix} = \begin{bmatrix} P_1 \\ \vdots \\ P_f \end{bmatrix} \begin{bmatrix} X_1, \ldots, X_p \end{bmatrix}. \tag{16}$$

Sturm and Trigss [151] recover the projective depths in Equation (16) based on the computation of fundamental matrices and epipoles. By choosing an arbitrary initial depth value (such as $\lambda_{1p} = 1$), the overall projective depth can be recovered up to an arbitrary initial value using the following equation:

$$\lambda_{mp} = \frac{(e_{mn} \wedge x_{mp}).(F_{mn}x_{np})}{||e_{mn} \wedge x_{mp}||}\lambda_{np}, \tag{17}$$

where $m, n \in \{1, 2, \ldots, f\}$, and F_{mn} and e_{mn} are the fundamental matrix and the epipole computed from frame m and n, respectively. Once the projective depth is obtained, shape and motion can be recovered using SVD.

Hartley and Schaffalitzky [58] generalized the factorization based on a perspective camera for missing and uncertain data. They developed an iterative method based on power factorization to approximate data with missing entries with a low-rank matrix. Li et al. [93] iterate between motion segmentation using subspace separation and projective depth estimation in order to get a convergence result. The projective depth is estimated by minimizing the reprojection errors, followed by iterative refinement. On the contrary, Murakami et al. [114] tried to avoid the computation of projective depth by formulating depth-estimation-free conditions. The computation of Equation (17) is unnecessary if two conditions are met. First, the origins of the camera coordinates are on a plane. Second, the axes of the coordinate systems point to the perpendicular direction of the plane.

5.1.2 *Nonrigid Structure from Motion (NRSfM).*

In 2000, Bregler et al. [13] proposed Nonrigid Structure from Motion (NRSfM) technique based on Tomasi-Kanade factorization under a scaled orthography camera model for the first time. They represented a nonrigid object as a k key frame basis set $\{B_i\}_{i=1}^{k}$, where each B_i denotes a $3 \times p$ matrix describing p feature points. The linear combination of this basis set forms the shape of a specific configuration such that $B = \sum_{i=1}^{k} l_i.B_i$, where $B, B_i \in \mathbb{R}^{3\times p}$ and $l_i \in \mathbb{R}$. By normalizing the feature points as in [154] and eliminating the translation vector, the measurement matrix becomes

$$\tilde{W} = NB = \begin{bmatrix} l_{11}R'_1 & \cdots & l_{1k}R'_1 \\ \vdots & \ddots & \vdots \\ l_{f1}R'_f & \cdots & l_{fk}R'_f \end{bmatrix} . \begin{bmatrix} B_1 \\ \vdots \\ B_k \end{bmatrix}, \tag{18}$$

where R' is the first two rows of rotation matrix R (due to the orthogonal projection of the orthographic camera model, the last row of R can be estimated by computing the cross-product of the first and the second row of R). The factorization of $\tilde{W}$ can be done using SVD by choosing the first $3k$ singular vectors and singular values. The estimated rotation matrix R'_f and the shape basis weights l_i can be recovered from N by reordering the entries of N and factorizing it using SVD. Finally, orthonomality constraints are enforced such that there is a matrix G that maps R'_f and B_k into a unique solution [13].

As an alternative to orthonomality constraints, Xiao et al. [176] introduce basis constraints such that the nonrigid factorization problem can be solved in a closed-form solution. Instead of enforcing the metric constraints directly, Paladini et al. [123] project the motion matrix onto the manifold of matrix constraints and thus the factorization can be done iteratively through least squares. In contrast, Akhter et al. [3] propose a dual solution by presenting a trajectory-space-based technique such that the computation of basis vectors is not needed. The Discrete Cosine Transform (DCT) is used to compactly describe the body motions. Following these results, Dai et al. [27] tried to remove any additional constraints for nonrigid reconstruction (e.g., information about the nonrigid scene, nonrigid shape basis, the coefficients, the deformations, the shapes, etc.) by proposing a prior-free method using low-rank constraints only. Recently, Kumar et al. [80] proposed to fuse together MBSfM and NRSfM into a multibody nonrigid deformations system. They modeled the feature trajectories as a union of multiple linear or affine subspaces. It enables one to jointly optimize nonrigid reconstruction and nonrigid motion segmentation using the alternating direction method of multipliers (ADMM).

6 DISCUSSION OF ADVANTAGES AND DISADVANTAGES

Despite the recent advances in visual SLAM and Structure from Motion in dynamic environments, each proposed approach comes with advantages and disadvantages. Tables 1, 2, and 3 list the summary of approaches for motion segmentation, dynamic object segmentation and 3D tracking, and joint motion segmentation and reconstruction, respectively. We define some practical considerations for each class and point out whether the proposed method has considered these practical aspects. The table is populated to the best of our knowledge given the information provided in the article.

6.1 Motion Segmentation

The main advantage of background/foreground initialization is the ability to keep track of the moving objects when they are temporarily stationary. This capability enables the system to easily retrack the moving object without the need to perform a new segmentation when the temporarily stopped object starts to move again. Moreover, thanks to the tracking-by-detection scheme, this approach will have no problem in dealing with degenerate motion (e.g., the object moves along the epipolar plane and its direction and velocity are similar to the camera). However, this approach has two main drawbacks. First, information related to the background or the object needs to be defined beforehand. Second, this tracking-by-detection scheme may hinder the real-time capability, especially when the environment contains many moving objects since it needs to exhaustively match all objects with the detector except when a cascade architecture is carefully implemented [68].

Compared to background and foreground initialization, geometric constraints do not possess the capability to handle temporary stopping since the determination of segmentation is based on the motion only, which is indicated by the high geometric error. Another shortcoming is that this approach cannot differentiate between the residual error caused by the moving object or caused by the false correspondence (outliers) since both conditions result in high geometric errors. Furthermore, some techniques cannot handle motion in degenerate conditions unless other measures are imposed. Kundu et al. [84] set a fixed threshold on the flow vector such that the degenerate motion lies outside the bound. Although this approach works well for particular motions, it is not applicable in general conditions since arbitrary object motion may violate the threshold. However, a geometric constraints-based approach needs no prior knowledge about the background or the moving objects. Moreover, since all computations of the residual errors are part of the standard visual SLAM or SfM technique, there is no additional computational burden in performing the segmentation, and thus real-time implementation is common.

Optical flow-based techniques have similar properties with geometric-based approaches. They need no prior knowledge about the environment and can work in real time. However, they work based on the brightness constancy assumption, which is sensitive to changes in lighting conditions [62]. Without proper implementation of the image pyramid, it is also sensitive to a large pixel movement [11]. Moreover, it has difficulty in handling degenerate motion since when the object is moving at the same plane, direction, and velocity with the camera, the flow vector will be small and the moving object looks like a part of static background. New segmentation is also needed when the object starts to move after a stop.

Ego-motion constraints can easily segment static features from dynamic ones by fitting features that conform with the defined ego-motion. This approach can run in real time. It also can handle degenerate motion since it relies on external information and thus only static features will conform with the correct ego-motion. However, it needs prior knowledge about the motion of the camera. Since it fits directly features that satisfy the ego-motion, when the object is temporary stopping, it will be viewed as part of the static scene.

6.2 Dynamic Object Segmentation

There are a few advantages of the statistical model selection technique for dynamic object segmentation. First, it can handle degenerate motion as long as the system allows the computation of lower-dimensional or lower-degree-of-freedom motion. Second, it needs no prior knowledge about the environment (the number of moving objects is automatically captured when the whole data is described by n different motion models). Third, since statistic-based approaches fit a model based on the cardinality of the inlier set, noise and outliers are automatically tackled. Nevertheless, by explicitly estimating the scale of noise, model fitting can have better performance [139]. Finally, statistical model selection can be implemented as a sequential algorithm that processes one new image at a time, although real-time implementation remains difficult.

The main problem of statistical model selection is that fitting a motion model from randomly sampled data is computationally expensive. Under RANSAC, the number of iterations required to guarantee a correct solution is

$$N = \frac{log(1-p)}{log(1-(1-\varepsilon)^s)},$$

where s is the number of data points, ε is the percentage of outliers, and p is the probability of success (confidence level) [39]. If we assume that a motion lies in 20% of the whole data and we want a 99% probability of success, then it needs 359,777 iterations for computing the fundamental matrix to fit the motion correctly (fundamental matrix needs minimal 7 points) [139]. One of the most effective ways to reduce the number of iterations is by using a motion model with lower minimal point requirements such as in [132] and [133]. However, this approach needs an assumption of how the camera moves over time. Finally, dependent motion remains a challenging problem for statistical model selection since a group of features can be part of two different motion models. Incorporating overlapping motions in the joint likelihood function [139] can tackle the problem, although it remains difficult in the presence of outliers.

Compared to statistical model selection, most subspace clustering methods are relatively cheaper in computation time because they are mostly based on an algebraic method (particularly SVD, which needs $O(fp^2)$ operations, where f is the number of frames and p is the number of points). Furthermore, some recent techniques allow intersection between subspaces, thus allowing it to deal with dependent motions [35, 128, 165]. However, current developments of subspace clustering have several limitations. First, they cannot run sequentially (except [161, 185]) or in real time since they need the whole sequence to be available before processing (batch mode). Second, some methods need the information about the number of motions in the scene or the dimension where the subspace lies, although a recent technique provides a means to find it [34, 177, 179]. Third, most approaches make use of the affine camera model, which will fail if the scene contains a major perspective effect. Using the affine camera model, the motion is assumed to lie in a linear or affine manifold. Under perspective projection, the problem becomes more difficult since a motion might lie in a nonlinear manifold. Fourth, although recent techniques specify how to handle noise [35, 177], outliers [35, 185], and missing data [128, 179], they only work to some extent and a practical implementation to long sequences remains difficult.

Unlike subspace clustering techniques that can only segment data in a linear or affine subspace, the geometry-based approach works under the perspective camera model; thus, it can handle data that lies in a nonlinear manifold. However, since the current approach extends standard multiple-view geometry to static scenes, it only supports the fundamental matrix as the motion model, which means that there is no way to handle degenerate motion. Another problem is that the number of image pairs needed for computing a multibody fundamental matrix grows exponentially with respect to the number of motions (e.g., for $n = 1, 2, 3, 4$, the required number of images is 8, 35, 99, and 225). For large motions, the number of images can reach $O(n^4)$ [166], an effect of transforming multibody epipolar constraints into a linear representation. Finally, the effect of noise, outliers, and missing data have not been well studied.

6.3 3D Tracking of Dynamic Objects

One advantage of trajectory triangulation is that it can work incrementally, although it might need several frames for each iteration (5, 9, etc., depending on the trajectory assumption [6]). Prior knowledge about the camera motion is not needed, although some approaches [5, 6] assume that the camera pose is available. The main limitation of trajectory triangulation is that the object trajectory should be known or at least should follow a specific parametric form. This assumption limits the application of trajectory triangulation for arbitrary object motion, although some researchers attempt to extend it into general motion [69, 70, 124]. Moreover, it remains difficult to handle outliers and missing data because it needs several image sequences in order to have a unique solution. Finally, most techniques can only reconstruct rigid body motion (except [124, 125]), which limits the application into a specific problem.

Particle filters are probably the only technique for doing 3D reconstruction and tracking of dynamic objects that can work in real time so far, although they are strictly limited to a small number of moving objects (computationally expensive for many objects) [83]. Knowledge about the object trajectory is not needed, although the assumption of the object velocity is required since it is used for the prediction. Some constraints also need to be enforced to limit the spread of the particles in the space. Moreover, it is probably difficult to extend it into nonrigid or articulated reconstruction since nonrigid and articulated motion may not conform to the constant velocity motion model.

6.4 Joint Motion Segmentation and Reconstruction

The main advantage of the factorization-based approach is that the problem of motion segmentation and reconstruction can be solved simultaneously. Knowledge about the camera motion is not needed and it can be extended elegantly into nonrigid reconstruction. However, factorization has some limitations. First, most approaches work based on orthography or the affine camera model, which prevents its implementation in conditions with a large perspective effect, a condition often found in exploratory tasks. Reconstruction with a large perspective effect is still possible, although it remains a challenging problem since projective depth should be recovered first [114, 151]. Second, it cannot run in real time (or even incrementally, except [106]) because all feature point trajectories should be available beforehand (batch mode). Additionally, most approaches derive their technique based on SVD, which needs $O(fp^2)$ complexity. Third, some techniques may need prior knowledge, such as the number of moving objects in the scene, rank of the measurement matrix, or the dimension of the object [25, 58, 80]. Fourth, it is sensitive to noise and outliers since the segmentation and the reconstruction are generally based on thresholding on the entries of the interaction matrix. Finally, missing data is also a problem since the entries of the measurement matrix should be complete before doing SVD.

6.5 Deep Learning

The key advantages of the deep-learning-based approach is that it can eliminate the hand-engineered feature extraction step [88], which results in the reduction of problems in feature correspondences such as noisy correspondences, outliers, and missing data due to losing track or occlusions. Deep learning also does not need to specify the camera model, which currently limits approaches like subspace clustering or factorization to be applied for the general perspective camera model. Moreover, the ability to learn the nonlinear representation of the data gives an opportunity to generalize well in different environments, a problem that remains difficult to handle using standard feature-based approaches that typically manually fine-tune the algorithm parameters for different environments.

However, since techniques in dynamic object segmentation and reconstruction involve some geometry computations, it remains a challenge to construct a DNN architecture that can understand this geometry and gain competitive accuracy compared to standard feature-based techniques. Current approaches such as [96] still need the help of conventional feature extraction techniques since the extracted spatiotemporal feature is not precise and does not understand the geometry of the moving objects. The approaches in [168] and [15] show that training the network to segment motions can be done, although a certain number of motions in the image are required. It also needs camera intrinsic parameters in order to predict depth. Moreover, the technique is developed based on the optical flow principle, yet optical flow has difficulty detecting degenerate object motion.

7 CONCLUSIONS

Significant progress has been made in the past few decades to solve the problem of visual simultaneous localization and reconstruction in dynamic environments. This article surveys and highlights existing approaches and connects the field of SfM and visual SLAM with dynamic object segmentation and tracking. We have classified approaches according to the type of problem they solve and their corresponding applications. Various approaches, both feature based and deep learning based, are presented and critically discussed from a practical perspective.

Further research is needed to enable practical implementations of simultaneous localization and reconstruction in dynamic environments. In general, handling missing, noisy, and outlier data remains a future challenge for most of the discussed techniques. Although statistical-based techniques can tackle this problem due to their recursive sampling approach, they have to trade off accuracy for computation cost. Most techniques also have difficulty in dealing with degenerate and dependent motion. While some subspace clustering techniques allow the intersection among motions, it is still limited to the case of noiseless data. Moreover, real-time implementation remains a difficult problem for dynamic object segmentation and 3D tracking due to the offline nature of the algorithms and their high computational cost. In order for dynamic object segmentation and 3D tracking techniques to be fused with standard visual SLAM, this real-time problem should be solved first. Finally, the deep-learning-based approach opens a new perspective by casting the localization and 3D reconstruction problem as a learning problem and eliminating the hand-crafted feature engineering step and the need to specify the camera model. Deep learning approaches, however, are still in their infancy, and the area presents a plethora of interesting challenges for future work.

ACKNOWLEDGMENTS

The authors would like to thank anonymous reviewers for their helpful suggestions towards improving our paper. Muhamad Risqi U. Saputra was supported by the Indonesia Endowment Fund for Education (LPDP).

REFERENCES

[1] Vincent J. Aidala and Sherry E. Hammel. 1983. Utilization of modified polar coordinates for bearings-only tracking. *IEEE Trans. Automat. Contr.* 28, 3 (1983), 283–294.

[2] Hirotogu Akaike. 1973. Information theory and an extension of the maximum likelihood principle. In *Int. Symp. Inf. Theory*. 267–281.

[3] Ijaz Akhter, Sohaib Khan, Yaser Sheikh, and Takeo Kanade. 2008. Nonrigid structure from motion in trajectory space. In *Adv. Neural Inf. Process. Syst.*, Vol. 1. 1–8.

[4] Pablo F. Alcantarilla, José J. Yebes, Javier Almazán, and Luis M. Bergasa. 2012. On combining visual slam and dense scene flow to increase the robustness of localization and mapping in dynamic environments. In *IEEE Int. Conf. Robot. Autom.* 1290–1297.

[5] Shai Avidan and Amnon Shashua. 1999. Trajectory triangulation of lines: Reconstruction of a 3D point moving along a line from a monocular image sequence. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Vol. 2. 66.

[6] Shai Avidan and Amnon Shashua. 2000. Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence. *IEEE Trans. Pattern Anal. Mach. Intell.* 22, 4 (2000), 348–357.

[7] Mohammadreza Babaee, Duc Tung Dinh, and Gerhard Rigoll. 2017. A deep convolutional neural network for background subtraction. In *arXiv:1702.01731.*

[8] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. 2008. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.* 110, 3 (2008), 346–359.

[9] Paul A. Beardsley, Andrew Zisserman, and David W. Murray. 1994. Navigation using affine structure from motion. In *Eur. Conf. Comput. Vis.* 85–96.

[10] Francisco Bonin-Font, Alberto Ortiz, Gabriel Oliver, Francisco Bonin-font Alberto, and Ortiz Gabriel. 2008. Visual navigation for mobile robots: A survey. *J. Intell. Robot. Syst.* 53 (2008), 263–296.

[11] Jean-Yves Bouguet. 2000. Pyramidal implementation of the affine Lucas Kanade feature tracker - Description of the algorithm. *Intel Corp. Microprocess. Res. Labs.*

[12] Terrance E. Boult and Lisa Gottesfeld Brown. 1991. Factorization-based segmentation of motions. In *IEEE Work. Vis. Motion.*

[13] Christoph Bregler, Aaron Herzmann, and Henning Biermann. 2000. Recovering non-rigid 3D shape from image streams. In *IEEE Conf. Comput. Vis. Pattern Recognit.*

[14] Michael D. Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. 2011. Online multi-person tracking-by-detection from a single, uncalibrated camera. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 9 (2011), 1820–1833.

[15] Arunkumar Byravan and Dieter Fox. 2017. SE3-Nets: Learning rigid body motion using deep neural networks. In *IEEE Int. Conf. Robot. Autom.*

[16] Jean-pierre L. E. Cadre and Olivier Tremois. 1998. Bearings-only tracking for maneuvering sources. *IEEE Trans. Aerosp. Electron. Syst.* 34, 1 (1998), 179–193.

[17] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. 2010. BRIEF: Binary robust independent elementary features. In *Eur. Conf. Comput. Vis.* 778–792.

[18] Robert O. Castle, Georg Klein, and David W. Murray. 2011. Wide-area augmented reality using camera tracking and mapping in multiple regions. *Comput. Vis. Image Underst.* 115, 6 (2011), 854–867.

[19] Stephen M. Chaves, Ayoung Kim, and Ryan M. Eustice. 2014. Opportunistic sampling-based planning for active visual SLAM. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*

[20] Jinhui Chen and Jian Yang. 2014. Robust subspace segmentation by low-rank representation. *IEEE Trans. Cybern.* 44, 8 (2014), 1432–1445.

[21] Falak Chhaya, Dinesh Reddy, Sarthak Upadhyay, Visesh Chari, M. Zeeshan Zia, and K. Madhava Krishna. 2016. Monocular reconstruction of vehicles: Combining SLAM with shape priors. In *IEEE Int. Conf. Robot. Autom.* 5758–5765.

[22] Ondrej Chum and Jiri Matas. 2005. Matching with PROSAC-Progressive Sample Consensus. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 220–226.

[23] Burcu Cinaz and Holger Kenn. 2008. HeadSLAM - Simultaneous localization and mapping with head-mounted inertial and laser range sensors. In *IEEE Int. Symp. Wearable Comput.*

[24] Joao Costeira and Takeo Kanade. 1995. A multi-body factorization method for motion analysis. In *Int. Conf. Comput. Vis.* 1071–1076.

[25] João Paulo Costeira and Takeo Kanade. 1998. A multibody factorization method for independently moving objects. *Int. J. Comput. Vis.* 29, 3 (1998), 159–179.

[26] Mark Cummins and Paul Newman. 2008. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *Int. J. Rob. Res.* 27, 6 (2008), 647–665.

[27] Yuchao Dai, Hongdong Li, and Mingyi He. 2014. A simple prior-free method for non-rigid structure-from-motion factorization. *Int. J. Comput. Vis.* 107, 2 (2014), 101–122.

[28] Danping Zhou and Ping Tan. 2012. CoSLAM: Collaborative visual SLAM in dynamic environments. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 2 (2012), 354–366.

[29] Andrew J. Davison. 2003. Real-time simultaneous localisation and mapping with a single camera. In *IEEE Int. Conf. Comput. Vis.*

[30] Maxime Derome, Aurelien Plyer, Martial Sanfourche, and Guy Le Besnerais. 2015. Moving object detection in real-time using stereo from a mobile platform. *Unmanned Syst.* 3, 4 (2015), 253–266.

[31] Maxime Derome, Aurelien Plyer, Martial Sanfourche, and Guy Le Besnerais. 2014. Real-time mobile object detection using stereo. In *13th Int. Conf. Control Autom. Robot. Vis. (ICARCV'14)*. 1021–1026.

[32] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. 2016. Deep image homography estimation. In *arXiv:1606.03798*.

[33] Alexey Dosovitskiy, Philipp Fischery, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. 2016. FlowNet: Learning optical flow with convolutional networks. In *IEEE Int. Conf. Comput. Vis.*, Vol. 11-18-Dece. 2758–2766.

[34] Ehsan Elhamifar and Rene Vidal. 2009. Sparse subspace clustering. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.* 2790–2797.

[35] Ehsan Elhamifar and Rene Vidal. 2013. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 11 (2013), 2765–2781.

[36] Jakob Engel, Thomas Sch, and Daniel Cremers. 2014. LSD-SLAM: Direct monocular SLAM. In *Eur. Conf. Comput. Vis.* 834–849.

[37] Martin A. Fischler and Robert C. Bolles. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24 (1981), 381–395.

[38] Katerina Fragkiadaki, Pablo Arbelaez, Panna Felsen, and Jitendra Malik. 2015. Learning to segment moving objects in videos. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 4083–4090.

[39] Friedrich Fraundorfer and Davide Scaramuzza. 2012. Visual odometry: Part II - matching, robustness, optimization, and applications. *IEEE Robot. Autom. Mag.* 19, 2 (2012), 78–90.

[40] Jorge Fuentes-Pacheco, Jose Ruiz-Ascencio, and Juan Manuel Rendon-Mancha. 2012. Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev.* 43, 1 (2012), 55–81.

[41] Dorian Galvez-Lopez and Juan D. Tardos. 2012. Bags of binary words for fast place recognition in image sequences. *IEEE Trans. Robot.* 28, 5 (2012), 1188–1197.

[42] Xiao Shan Gao, Xiao Rong Hou, Jianliang Tang, and Hang Fei Cheng. 2003. Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Anal. Mach. Intell.* 25, 8 (2003), 930–943.

[43] Emilio Garcia-Fidalgo and Alberto Ortiz. 2015. Vision-based topological mapping and localization methods: A survey. *Rob. Auton. Syst.* 64 (2015), 1–20.

[44] C. W. Gear. 1998. Multibody grouping from motion images. *Int. J. Comput. Vis.* 29, 2 (1998), 133–150.

[45] Andreas Geiger, Julius Ziegler, and Christoph Stiller. 2011. StereoScan: Dense 3D reconstruction in real-time. In *IEEE Intell. Veh. Symp.* 1–9.

[46] Arturo Gil, Oscar Reinoso, Monica Ballesta, and Miguel Julia. 2010. Multi-robot visual SLAM using a Rao-Blackwellized particle filter. *Rob. Auton. Syst.* 58, 1 (2010), 68–80.

[47] Georgia Gkioxari and Jitendra Malik. 2015. Finding action tubes. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*

[48] Susanna Gladh, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. 2016. Deep motion features for visual tracking. In *Int. Conf. Pattern Recognit.*

[49] Alvina Goh and Rene Vidal. 2007. Segmenting motions of different types by unsupervised manifold clustering. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*

[50] Venu Madhav Govindu. 2001. Combining two-view constraints for motion estimation. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*

[51] H. M. Gross, H. J. Boehme, C. Schroeter, S. Mueller, A. Koenig, Ch. Martin, M. Merten, and A. Bley. 2008. Shopbot: Progress in developing an interactive mobile shopping assistant for everyday use. In *IEEE Int. Conf. Syst. Man Cybern.* 3471–3478.

[52] Yanming Guo, Yu Liu, Ard Oerlemans, Songyang Lao, Song Wu, and Michael S. Lew. 2015. Deep learning for visual understanding: A review. *Neurocomputing* 187 (2015), 27–48.

[53] Hugh C. Longuet-Higgins. 1981. A computer algorithm for reconstructing a scene from two projections. *Nature* 293 (1981), 133–135.

[54] Mei Han and Takeo Kanade. 2004. Reconstruction of a scene with multiple linearly moving objects. *Int. J. Comput. Vis.* 59, 3 (2004), 285–300.

[55] Ankur Handa, Michael Bloesch, Viorica Patraucean, Simon Stent, John McCormac, and Andrew Davison. 2016. gvnn: Neural network library for geometric computer vision. In *arXiv:1607.07405*.

[56] Chris Harris and Carl Stennett. 1990. RAPID - A video rate object tracker. In *Br. Mach. Vis. Conf.*

[57] Chris Harris and Mike Stephens. 1988. A combined corner and edge detector. In *Alvey Vis. Conf.* 147–151.

[58] Richard Hartley and Frederik Schaffalitzky. 2003. PowerFactorization: 3D reconstruction with missing or uncertain data. In *Aust. Adv. Work. Comput. Vis.*, Vol. 74. 1–9.

[59] Richard Hartley and Andrew Zisserman. 2004. *Multiple View Geometry in Computer Vision* (2nd ed.). Cambridge University Press.

[60] Richard I. Hartley and Peter Sturm. 1997. Triangulation. *Comput. Vis. Image Underst.* 68, 2 (1997), 146–157.

[61] Stephan Heuel and Wolfgang Förstner. 2001. Matching, reconstructing and grouping 3D lines from multiple views using uncertain projective geometry. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*

[62] Berthold K. P. Horn and Brian G. Schunck. 1981. Determining optical flow. *Artif. Intell.* 17, 1–3 (1981), 185–203.

[63] Stefan Hrabar, Gaurav S. Sukhatme, Peter Corke, Kane Usher, and Jonathan Roberts. 2005. Combined optic-flow and stereo-based navigation of urban canyons for a UAV. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* 302–309.

[64] Thomas S. Huang and Arun N. Netravali. 1994. Motion and structure from feature correspondences: A review. *Proc. IEEE* 82, 2 (1994), 252–268.

[65] Naoyuki Ichimura. 1999. Motion segmentation based on factorization method and discriminant critea. In *IEEE Int. Conf. Comput. Vis.*

[66] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. 2017. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conf. Comput. Vis. Pattern Recognit.*

[67] Eagle S. Jones and Stefano Soatto. 2011. Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *Int. J. Rob. Res.* 30, 4 (2011), 1–38.

[68] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. 2012. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 7 (2012), 1409–1422.

[69] Jeremy Yirmeyahu Kaminski and Mina Teicher. 2002. General trajectory triangulation. In *Eur. Conf. Comput. Vis.* 823–836.

[70] Jeremy Yirmeyahu Kaminski and Mina Teicher. 2004. A general framework for trajectory optimization. *J. Math. Imaging Vis.* 21 (2004), 27–41.

[71] Kenichi Kanatani. 1996. *Statistical Optimization for Geometric Computation: Theory and Practice.* Elsevier.

[72] Kenichi Kanatani. 2001. Motion segmentation by subspace separation and model selection. In *IEEE Int. Conf. Comput. Vis.* 586–591.

[73] Kenichi Kanatani and Chikara Matsunaga. 2002. Estimating the number of independent motions for multibody motion segmentation. In *Asian Conf. Comput. Vis.*

[74] Jens Klappstein, Tobi Vaudrey, Clemens Rabe, Andreas Wedel, and Reinhard Klette. 2009. Moving object segmentation using optical flow and depth information. In *Pacific-Rim Symp. Image Video Technol.* 611–623.

[75] Georg Klein and David Murray. 2007. Parallel tracking and mapping for small AR workspaces. In *IEEE ACM Int. Symp. Mix. Augment. Real.*

[76] Georg Klein and David Murray. 2009. Parallel tracking and mapping on a camera phone. In *8th IEEE Int. Symp. Mix. Augment. Real.* 83–86.

[77] Kishore Konda and Roland Memisevic. 2013. Unsupervised learning of depth and motion. In *arXiv:1312.3429*.

[78] Kishore Konda and Roland Memisevic. 2015. Learning visual odometry with a convolutional network. In *Int. Conf. Comput. Vis. Theory Appl.* 486–490.

[79] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Adv. Neural Inf. Process. Syst.* 1–9.

[80] Suryansh Kumar, Yuchao Dai, and Hongdong Li. 2016. Multi-body non-rigid structure-from-motion. In *Int. Conf. 3D Vis.* 148–156.

[81] Rainer Kummerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. 2011. G2o: A general framework for graph optimization. In *IEEE Int. Conf. Robot. Autom.* 3607–3613.

[82] Abhijit Kundu, K. Madhava Krishna, and C. V. Jawahar. 2010. Realtime motion segmentation based multibody visual SLAM. In *7th Indian Conf. Comput. Vision, Graph. Image Process.* 251–258.

[83] Abhijit Kundu, K. Madhava Krishna, and C. V. Jawahar. 2011. Realtime multibody visual SLAM and tracking with a smoothly moving monocular camera. In *IEEE Int. Conf. Comput. Vis.*

[84] Abhijit Kundu, K. Madhava Krishna, and Jayanthi Sivaswamy. 2009. Moving object detection by multi-view geometric techniques from a single camera mounted robot. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* 4306–4312.

[85] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. 2016. Deeper depth prediction with fully convolutional residual networks. In *Int. Conf. 3D Vis.* 239–248.

[86] Quoc V. Le, Alexandre Karpenko, Jiquan Ngiam, and Andrew Y. Ng. 2011. ICA with reconstruction cost for efficient overcomplete feature learning. In *Adv. Neural Inf. Process. Syst.* 1–9.

[87] Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, and Andrew Y. Ng. 2011. Building high-level features using large scale unsupervised learning. In *Int. Conf. Mach. Learn.* 38115.

[88] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2016. Deep learning. *Nature* 521 (2016), 436–444.

[89] Kuan Hui Lee, Jenq Neng Hwang, Greg Okapal, and James Pitton. 2014. Driving recorder based on-road pedestrian tracking using visual SLAM and constrained multiple-kernel. In *17th IEEE Int. Conf. Intell. Transp. Syst.* 2629–2635.

[90] Kuan-hui Lee, Jenq-neng Hwang, Greg Okopal, and James Pitton. 2016. Ground-moving-platform-based human tracking using visual SLAM and constrained multiple kernels. *IEEE Trans. Intell. Transp. Syst.* 17, 12 (2016), 3602–3612.

[91] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. 2011. BRISK: Binary robust invariant scalable keypoints. In *IEEE Int. Conf. Comput. Vis.* 2548–2555.

[92] Stefan Leutenegger, Paul Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. 2013. Keyframe-based visual-inertial SLAM using nonlinear optimization. *Int. J. Rob. Res.* 34, 3 (2013), 314–334.

[93] Ting Li, Vinutha Kallem, Dheeraj Singaraju, and Rene Vidal. 2007. Projective factorization of multiple rigid-body motions. In *IEEE Conf. Comput. Vis. Pattern Recognit.*

[94] Hyon Lim, Jongwoo Lim, and H. Jin Kim. 2014. Real-time 6-DOF monocular visual SLAM in a large-scale environment. In *IEEE Int. Conf. Robot. Autom.*

[95] Kuen-Han Lin and Chieh-Chih Wang. 2010. Stereo-based simultaneous localization, mapping and moving object tracking. In *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*

[96] Tsung Han Lin and Chieh-Chih Wang. 2014. Deep learning of spatio-temporal features with geometric-based moving point detection for motion segmentation. In *IEEE Int. Conf. Robot. Autom.* 3058–3065.

[97] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. 2013. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 1 (2013), 171–184.

[98] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recognit.* 3431–3440.

[99] David G. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* 60, 2 (2004), 91–110.

[100] Bruce D. Lucas and Takeo Kanade. 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. In *DARPA Image Underst. Work.* 121–130.

[101] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. 2016. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE Conf. Comput. Vis. Pattern Recognit.*

[102] Christopher Mei, Gabe Sibley, Mark Cummins, Paul Newman, and Ian Reid. 2011. RSLAM: A system for large-scale mapping in constant-time using stereo. *Int. J. Comput. Vis.* 94, 2 (2011), 198–214.

[103] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. 2017. Relative camera pose estimation using convolutional neural networks. In *arXiv:1702.01381.*

[104] Davide Migliore, Roberto Rigamonti, Daniele Marzorati, Matteo Matteucci, and Domenico G. Sorrenti. 2009. Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments. In *ICRA Work. Safe Navig. Open Dyn. Environ. Appl. to Auton. Veh.*

[105] Vikram Mohanty, Shubh Agrawal, Shaswat Datta, Arna Ghosh, Vishnu Dutt Sharma, and Debashish Chakravarty. 2016. DeepVO: A deep learning approach for monocular visual odometry. In *arXiv:1611.06069.*

[106] Toshihiko Morita and Takeo Kanade. 1993. A sequential factorization method for recovering shape and motion from image streams. *Proc. Natl. Acad. Sci.* 90, 21 (1993), 9795–9802.

[107] Pierre Moulon, Pascal Monasse, and Renaud Marlet. 2013. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *IEEE Int. Conf. Comput. Vis.* 3248–3255.

[108] Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. 2006. Monocular vision based SLAM for mobile robots. In *18th Int. Conf. Pattern Recognit.*

[109] Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. 2006. Real time localization and 3D reconstruction. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 1–8.

[110] Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. 2007. Generic and real-time structure from motion. In *Br. Mach. Vis. Conf.* 64.1–64.10.

[111] Etienne Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. 2009. Generic and real-time structure from motion using local bundle adjustment. *Image Vis. Comput.* 27, 8 (2009), 1178–1193.

[112] Peter Muller and Andreas Savakis. 2017. Flowdometry: An optical flow and deep learning based approach to visual odometry. In *IEEE Winter Conf. Appl. Comput. Vis.*

[113] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardos. 2015. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robot.* 31, 5 (2015), 1147–1163.

[114] Yohei Murakami, Takeshi Endo, Yoshimichi Ito, and Noboru Babaguchi. 2012. Depth-estimation-free projective factorization and its application to 3D reconstruction. In *Asian Conf. Comput. Vis.* 150–162.

[115] Richard A. Newcombe, David Molyneaux, David Kim, Andrew J. Davison, Jamie Shotton, Steve Hodges, Andrew Fitzgibbon, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *IEEE Int. Symp. Mix. Augment. Real.* 127–136.

[116] David Nister. 2004. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Mach. Intell.* 26, 6 (2004), 756–770.

[117] David Nistér, Oleg Naroditsky, and James Bergen. 2004. Visual odometry. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 652–659.

[118] John Oliensis. 2000. A critique of structure-from-motion algorithms. *Comput. Vis. Image Underst.* 80, 2 (2000), 172–214.

[119] D. Ortín and J. Montiel. 2001. Indoor robot motion based on monocular images. *Robotica* 19, 3 (2001), 331–342.

[120] Nobuyuki Otsu. 1979. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man. Cybern.* SMC-9, 1 (1979), 62–66.

[121] Kemal Egemen Ozden, Kurt Cornelis, Luc Van Eycken, and Luc Van Gool. 2004. Reconstructing 3D trajectories of independently moving objects using generic constraints. *Comput. Vis. Image Underst.* 96, 3 (2004), 453–471.

[122] Kemal E. Ozden, Konrad Schindler, and Luc Van Gool. 2010. Multibody structure-from-motion in practice. *IEEE Trans. Pattern Anal. Mach. Intell.* 32, 6 (2010), 1134–1141.

[123] Marco Paladini, Alessio Del Bue, Marko Stošić, Marija Dodig, João Xavier, and Lourdes Agapito. 2009. Factorization for non-rigid and articulated structure using metric projections. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 2898–2905.

[124] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 2010. 3D reconstruction of a moving point from a series of 2D projections. In *Eur. Conf. Comput. Vis.* 158–171.

[125] Hyun Soo Park, Takaaki Shiratori, Iain Matthews, and Yaser Sheikh. 2015. 3D trajectory reconstruction under perspective projection. *Int. J. Comput. Vis.* 115, 2 (2015), 115–135.

[126] Massimo Piccardi. 2004. Background subtraction techniques: A review. In *EEE Int. Conf. Syst. Man Cybern.*, Vol. 4. 3099–3104.

[127] Jouni Rantakokko, Joakim Rydell, Peter Strömbäck, Peter Händel, Jonas Callmer, David Törnqvist, Fredrik Gustafsson, Magnus Jobs, and Mathias Grudén. 2011. Accurate and reliable soldier and first responder indoor positioning: Multisensor systems and cooperative localization. *IEEE Wirel. Commun.* 18, 2 (2011), 10–18.

[128] Shankar Rao, Roberto Tron, Rene Vidal, and Yi Ma. 2010. Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. *IEEE Trans. Pattern Anal. Mach. Intell.* 32, 10 (2010), 1832–1845.

[129] Jorma Rissanen. 1984. Universal coding, information, prediction, and eestimation. *IEEE Trans. Inf. Theory* 30, 4 (1984), 629–636.

[130] Edward Rosten and Tom Drummond. 2006. Machine learning for high-speed corner detection. In *Eur. Conf. Comput. Vis.*, Vol. 1. 430–443.

[131] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *IEEE Int. Conf. Comput. Vis.* 2564–2571.

[132] Reza Sabzevari and Davide Scaramuzza. 2014. Monocular simultaneous multi-body motion segmentation and reconstruction from perspective views. In *IEEE Int. Conf. Robot. Autom.* 23–30.

[133] Reza Sabzevari and Davide Scaramuzza. 2016. Multi-body motion estimation from monocular vehicle-mounted cameras. *IEEE Trans. Robot.* 32, 3 (2016), 638–651.

[134] Muhamad Risqi Utama Saputra, Widyawan, and Paulus Insap Santosa. 2014. Obstacle avoidance for visually impaired using auto-adaptive thresholding on Kinect's depth image. In *11th IEEE Int. Conf. Ubiquitous Intell. Comput.* 337–342.

[135] Lawrence K. Saul and Sam T. Roweis. 2003. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *J. Mach. Learn. Res.* 4, 1999 (2003), 119–155.

[136] Davide Scaramuzza. 2011. 1-point-RANSAC structure from motion for vehicle-mounted cameras by exploiting nonholonomic constraints. *Int. J. Comput. Vis.* 95, 1 (2011), 74–85.

[137] Davide Scaramuzza, Friedrich Fraundorfer, and Roland Siegwart. 2009. Real-time monocular visual odometry for on-road vehicles with 1-point RANSAC. In *IEEE Int. Conf. Robot. Autom.* 4293–4299.

[138] Konrad Schindler and David Suter. 2005. Two-view multibody structure-and-motion with outliers. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*

[139] Konrad Schindler and David Suter. 2006. Two-view multibody structure-and-motion with outliers through model selection. *IEEE Trans. Pattern Anal. Mach. Intell.* 28, 6 (2006), 983–995.

[140] Konrad Schindler, David Suter, and Hanzi Wang. 2008. A model-selection framework for multibody structure-and-motion of image sequences. *Int. J. Comput. Vis.* 79, 2 (2008), 159–177.

[141] Konrad Schindler, James U., and Hanzi Wang. 2006. Perspective n-view multibody structure-and-motion through model selection. In *Eur. Conf. Comput. Vis.*, Vol. 1. 606–619.

[142] Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-motion revisited. In *IEEE Conf. Comput. Vis. Pattern Recognit.* 4104–4113.

[143] Gideon Schwarz. 1978. Estimating the dimension of a model. *Ann. Stat.* 6, 2 (1978), 461–464.

[144] Amnon Shashua, Shai Avidan, and Michael Werman. 1999. Trajectory triangulation over conic sections. In *IEEE Int. Conf. Comput. Vis.*

[145] Gabe Sibley, Christopher Mei, Ian Reid, and Paul Newman. 2010. Vast-scale outdoor navigation using adaptive relative bundle adjustment. *Int. J. Rob. Res.* 29, 8 (2010), 958–980.

[146] Karen Simonyan and Andrew Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. In *Adv. Neural Inf. Process. Syst.* 1–9.

[147] Noah Snavely, Steven Seitz, and Richard Szeliski. 2006. PhotoTourism: Exploring photo collections in 3D. In *SIGGRAPH Conf. Proc.* 835–846.

[148] Noah Snavely, Steven M. Seitz, and Richard Szeliski. 2008. Modeling the world from internet photo collections. *Int. J. Comput. Vis.* 80, 2 (2008), 189–210.

[149] Joan Solà. 2007. *Towards Visual Localization, Mapping and Moving Objects Tracking by a Mobile Robot: A Geometric and Probabilistic Approach.* Ph.D. Dissertation. Institut National Politechnique de Toulouse.

[150] Hauke Strasdat, J. M. M. Montiel, and Andrew J. Davison. 2012. Visual SLAM: Why filter? *Image Vis. Comput.* 30, 2 (2012), 65–77.

[151] Peter Sturm and Bill Triggs. 1996. A factorization based algorithm for multi-image projective structure and motion. In *Eur. Conf. Comput. Vis.*, Vol. 1065. 710–720.

[152] Wei Tan, Haomin Liu, Zilong Dong, Guofeng Zhang, and Hujun Bao. 2013. Robust monocular SLAM in dynamic environments. In *IEEE Int. Symp. Mix. Augment. Real.*

[153] Ninad Thakoor, Jean Gao, and Venkat Devarajan. 2010. Multibody structure-and-motion segmentation by branch-and-bound model selection. *IEEE Trans. Image Process.* 19, 6 (2010), 1393–1402.

[154] Carlo Tomasi and Takeo Kanade. 1992. Shape and motion from image streams under orthography: A factorization method. In *Int. J. Comput. Vis.*, Vol. 9. 137–154.

[155] Philip H. S. Torr. 1998. Geometric motion segmentation and model selection. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* 356, 1740 (1998), 1321–1340.

[156] Philip H. S. Torr and Andrew Zisserman. 1997. Robust parameterization and computation of the trifocal tensor. *Image Vis. Comput.* 15, 8 (1997), 591–605.

[157] Philip H. S. Torr and Andrew Zisserman. 1999. Feature based methods for structure and motion estimation. In *Int. Work. Vis. Algorithms.*

[158] Philip H. S. Torr and Andrew Zisserman. 2000. MLESAC: A new robust estimator with application to estimating image geometry. *Comput. Vis. Image Underst.* 78, 1 (2000), 138–156.

[159] Roberto Tron and Rene Vidal. 2007. A benchmark for the comparison of 3-D motion segmentation algorithms. In *IEEE Conf. Comput. Vis. Pattern Recognit.* 1–8.

[160] Sepehr Valipour, Mennatullah Siam, Martin Jagersand, and Nilanjan Ray. 2017. Recurrent fully convolutional networks for video segmentation. In *IEEE Winter Conf. Appl. Comput. Vis.* 1–12.

[161] René Vidal. 2006. Online clustering of moving hyperplanes. In *Adv. Neural Inf. Process. Syst.* 1433–1440.

[162] Rene Vidal. 2011. Subspace clustering. *IEEE Signal Process. Mag.* 28, 2 (2011), 52–68.

[163] René Vidal and Richard Hartley. 2008. Three-view multibody structure from motion. *IEEE Trans. Pattern Anal. Mach. Intell.* 30, 2 (2008), 214–227.

[164] René Vidal, Yi Ma, and Shankar Sastry. 2005. Generalized principal component analysis (GPCA). In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 40, 12 (2005), 1945–1959.

[165] René Vidal, Yi Ma, and Shankar Sastry. 2005. Generalized principal component analysis (GPCA). *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 12 (2005), 1945–1959.

[166] René Vidal, Yi Ma, Stefano Soatto, and Shankar Sastry. 2006. Two-view multibody structure from motion. *Int. J. Comput. Vis.* 68, 1 (2006), 7–25.

[167] René Vidal, Stefano Soatto, Yi Ma, and Shankar Sastry. 2002. Segmentation of dynamic scenes from the multibody fundamental matrix. In *ECCV Work. Vis. Model. Dyn. Scenes.*

[168] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. 2017. SfM-Net: Learning of structure and motion from video. In *arXiv:1704.07804.*

[169] Chieh-Chih Wang and Chuck Thorpe. 2002. Simultaneous localization and mapping with detection and tracking of moving objects. In *IEEE Int. Conf. Robot. Autom.*, Vol. 3. 2918–2924.

[170] Chieh-Chih Wang, Charles Thorpe, Sebastian Thrun, M. Hebert, and H. Durrant-Whyte. 2007. Simultaneous localization, mapping and moving object tracking. *Int. J. Rob. Res.* 26, 9 (2007), 889–916.

[171] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. 2017. DeepVO: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *IEEE Int. Conf. Robot. Autom.*

[172] Yin Tien Wang, Ming Chun Lin, and Rung Chi Ju. 2010. Visual SLAM and moving-object detection for a small-size humanoid robot. *Int. J. Adv. Robot. Syst.* 7, 2 (2010), 133–138.

[173] Somkiat Wangsiripitak and David W. Murray. 2009. Avoiding moving outliers in visual SLAM by tracking moving objects. In *IEEE Int. Conf. Robot. Autom.*

[174] Changchang Wu. 2013. Towards linear-time incremental structure from motion. In *Int. Conf. 3D Vis.* 127–134.

[175] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz. 2011. Multicore bundle adjustment. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* 3057–3064.

[176] Jing Xiao, Jin-xiang Chai, and Takeo Kanade. 2004. A closed-form solution to non-rigid shape and motion recovery. In *Eur. Conf. Comput. Vis.* 573–587.

[177] Jingyu Yan and Marc Pollefeys. 2006. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *Eur. Conf. Comput. Vis.*

[178] Jingyu Yan and Marc Pollefeys. 2008. A factorization-based approach for articulated nonrigid shape, motion, and kinematic chain recovery from video. *IEEE Trans. Pattern Anal. Mach. Intell.* 30, 5 (2008), 865–877.

[179] Congyuan Yang, Daniel Robinson, and Rene Vidal. 2015. Sparse subspace clustering with missing entries. In *Int. Conf. Mach. Learn.* 2463–2472.

[180] Georges Younes, Daniel Asmar, and Elie Shammas. 2016. A survey on non-filter-based monocular visual SLAM systems. In *arXiv:1607.00470.*

[181] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. 2015. An overview to visual odometry and visual SLAM: Applications to mobile robotics. *Intell. Ind. Syst.* 1, 4 (2015), 289–311.

[182] Luca Zappella, Alessio Del Bue, Xavier Lladó, and Joaquim Salvi. 2013. Joint estimation of segmentation and structure from motion. *Comput. Vis. Image Underst.* 117, 2 (2013), 113–129.

[183] Hendrik Zender, Patric Jensfelt, and Geert Jan M. Kruijff. 2007. Human- and situation-aware people following. In *IEEE Int. Work. Robot Hum. Interact. Commun.* 1131–1136.

[184] Dong Zhang and Ping Li. 2012. Visual odometry in dynamical scenes. *Sensors Transducers J.* 147, 12 (2012), 78–86.

[185] Teng Zhang, Arthur Szlam, and Gilad Lerman. 2009. Median K-flats for hybrid linear modeling with many outliers. In *Int. Conf. Comput. Vis. Work.* 234–241.

[186] Enliang Zheng, Ke Wang, Enrique Dunn, and Jan Michael Frahm. 2014. Joint object class sequencing and trajectory triangulation (JOST). In *Eur. Conf. Comput. Vis.* 599–614.

[187] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. 2017. Unsupervised learning of depth and ego-motion from video. In *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*

Received August 2017; revised December 2017; accepted December 2017