A Survey On Robust Visual Simultaneous Localization and Mapping

*

1st Ruotong Jia
Department of Mechanical Engineering
Northwestern University
Evanston, United States
ruotongjia2020@u.northwestern.edu

2nd Ying Wu
Department of Electrical Engineering
Northwestern University
Evanston, United States
ruotongjia2020@u.northwestern.edu

***Abstract*—In the last few decades, visual Simultaneous Localization (visual SLAM) and Mapping and Structure from Motion (SfM) have been a hotspot of research in 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. Most SfM and visual SLAM techniques are developed based on the assumption of static environment. These methods can be categorized as Robust Visual SLAM methods. In this survey, we present multiple techniques that are commonly used in major steps of Robust Visual SLAM. We identify five major steps of Robust Visual SLAM: feature extraction, feature matching, motion segmentation, localization, and 3D reconstruction. In each section, we present the core ideas of every method and we discuss their advantages and disadvantages. Finally, the future research trends in Robust VSLAM and conclusion of this survey is discussed.**

***Index Terms*—Robust Visual SLAM, Feature Extraction, Feature Matching, Motion Segmentation, Localization, 3D Reconstruction**

We thank The Master Of Science in Robotics Program at Northwestern University for their support in the completion of this paper.

I. Introduction

In the last few decades, the problems of camera pose estimation and 3D environment reconstruction has been a popular research topic in both the robotics and the computer vision communities. These problems are named "Visual Simultaneous and Localization"(VSLAM, online) or "Structure from Motion" (SfM, offline) in both communities. In this article, we follow the name VSLAM to address these problems. In many different applications, such as aerial vehicle surveying and driver-less car navigation, the VSLAM problem is always encountered, and is approached by many different methods.

In the early years of this field, the robotics and computer vision communities approached the problem separately. Longuet-Higgins's [4] work on the 8-point method was probably the initiation of a wave of SfM techniques. In their work, Longuet and Higgins proposed an approach to estimate the relative camera pose by solving the fundamental matrix of the camera model under epipolar geometry, using 8 point

correspondences between two consecutive image frames. Afterwards, other different perspectives for solving the problem including factorization [5], [6] and rotation averaging [2], [9] appeared. Also, some methods that worked in batch mode were developed, such as Bundler [10], [11] or Visual SfM [12].

The first approach that bridged the separation between the robotics and computer vision communities was MonoSLAM [3], which converted a general SLAM problem into a pure vision one. In MonoSLAM, image patches were processed to find landmarks [3], then the first-order uncertainty associated to the camera pose and landmark positions is propagated through a Bayesian filter framework, with real-time constraints on fast computation.

Nister et al. have presented an approach known as "visual odometry". This method estimates the motion of a stereo head or a simple camera in real-time, using visual data only: its aim is to provide guidance for robots [56]. Then, the emergence of PTAM [13] and the work of Mouragnon et al. [43] led to incremental SfM that can operate in real time using bundle adjustment (or variants such as Local Bundle Adjustment [43]), which was later shown to be more accurate than filter-based Visual SLAM methods given the same amount of computation time [57]. As the need for detailed maps and the availability of affordable RGB-D cameras grow, dense or semi-dense map based solutions were developed, such as KinectFusion [15] and LSD-SLAM [16].

II. WHAT IS ROBUST VISUAL SLAM

Among the existing Visual SLAM methods, many methods are implemented based on the assumption of static environment, i.e, the environment sensed by the on-board camera(s) is mostly composed of static features. The name Robust Visual SLAM refers to such a category of methods that use solely static features to achieve the robot pose estimate and the 3D structure of the robot. As a robustness problem, the Visual SLAM algorithm remains accurate despite dynamic features.

From a general classification perspective, SLAM methods can be categorized as direct and indirect methods [51]. In direct methods, we estimate camera pose by comparing intensity (depth can also be included) between the pixel in one image and its warped projection in another image, i.e, photo-metric error. In indirect methods, motion estimation is conducted by first computing some stable and intermediate geometric representations such as key-point [52],edge-lets [53], and optical flow [54]. Geometric error, often re-projection error between the actual image coordinates of the features and the image coordinates of the re-projected estimates, is then minimized using these reliable geometric representations. Among these geometric representations, popular choices include sliding-window or global bundle adjustment [55].Therefore, as static features are often required, Robust Visual SLAM is always a subset of indirect methods.

It is noteworthy to point out that even though rejecting dynamic features as outliers might be more time-efficient for computing, Robust Visual SLAM is prone to failure in highly dynamic environment. The reason for such a failure is that dynamic features usually result in occlusions or false correspondences [17]. Therefore, good techniques for segmenting static and dynamic features in images is highly necessary.

III. STRUCTURE OF ROBUST VISUAL SLAM

The general structure of Robust Visual SLAM is illustrated in Figure 1. The first step is to receive a sequence of images, from which salient features should be extracted. After detecting features from multiple frames, feature matches, also known as feature correspondences, are made to find static features by estimating the probability distribution or to optimize over selected images. Later on, from the observed

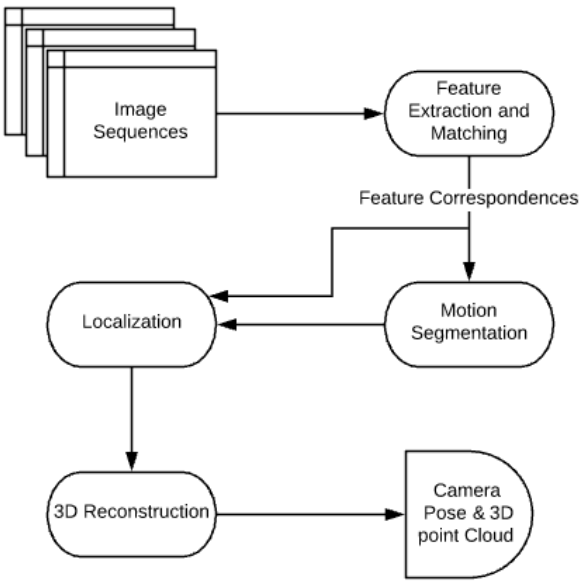

Fig. 1. An overview of the structure of this survey

static features, the pose of the robot and the structure of the 3D environment can be recovered, using filter-based approaches such as Kalman Filter or bundle adjustment(BA) [2].

In some literature, an extra step, loop closure detection is conducted to determines whether the camera has passed somewhere before. In loop closure detection, if the similarity of two frames is higher than a threshold, loop closure is detected, and the reconstructed 3D scene will have more confidence [47].

In the following sections, we further identify a sequence of actions and options in each action. We also classify options based on their similarity of their fundamental techniques. Here is the outline of the remainder of the review:

1) Feature Extraction
    a) Harris Corner Detector
    b) SIFT and SURF
    c) BRIEF
2) Feature Matching
    a) Kanade-Lucas-Tomasi Tracker (KLT)
    b) Random Sample Consensus on Highly Discriminative Feature Descriptors
    c) Visual-Inertial Feature Tracking Techniques
3) Motion Segmentation
    a) Background Initialization
    b) Geometric Constraints
    c) Optical Flow
4) Localization
    a) 8-point And 5-point Method
    b) Trifocal Tensor
    c) Ego-Motion Constraint Methods
5) 3D Reconstruction
    a) Bundle Adjustment

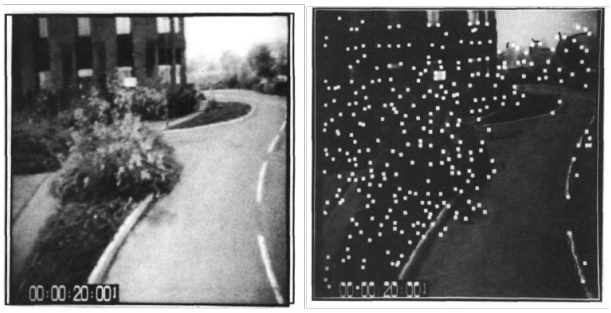

Fig. 2. Harris Corner Detector

IV. Feature Extraction

In feature-based visual SLAM, salient features are extracted to solve the image correspondence problem. In the early days of SfM [60], including the prominent "Visual Odometry" [56], Harris Corner detector was employed widely. In this section, Harris, SIFT, SURF, and BRIEF are introduced due to their extremely popular uses in VSLAM. However, feature extraction methods such as ORB, FREAK, KAZE are also popular [1].

A. Harris Corner Detector

As shown in Fig2, Harris Corner Detector was first introduced by Chris Harris and Mike Stephens in 1988 upon the improvement of Moravec's corner detector [50]. Compared with its predecessor, Harris' corner detector detects large variation in intensity by computing image gradient in each window directly, instead of using shifting patches for every 45-degree angles. This technique has been proved to be more accurate in distinguishing between edges and corners. Since then, it has been adopted in many Visual SLAM algorithms, one of them being the well-known work of Nister et al. in Visual Odometry [56]. Nister et al. choose to use Harris Corner Detector because it has the advantage of giving detection that is relatively stable under small to moderate image distortions [56].

The basic idea of Harris Corner Detector is that a corner could be recognized easily by examining intensity values within a small window. Also, shifting the window in any direction should yield a large change in appearance. In short, the mathematical principle of Harris Corner Detector can be summarized as follows:

1) W is a "window" detects a certain type of corner, and the change of intensity for the shift $[u, v]$ on the image is

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \quad (1)$$

Where $w(x,y)$ is the window function, and $I(x,y)$ is the intensity of the pixel (x,y)

2) Apply Taylor Expansion on Equation1, and we are able use image gradient in x or y directions I_x, I_y to write

$$E(u,v) \approx [u,v]M[u,v]' \quad (2)$$

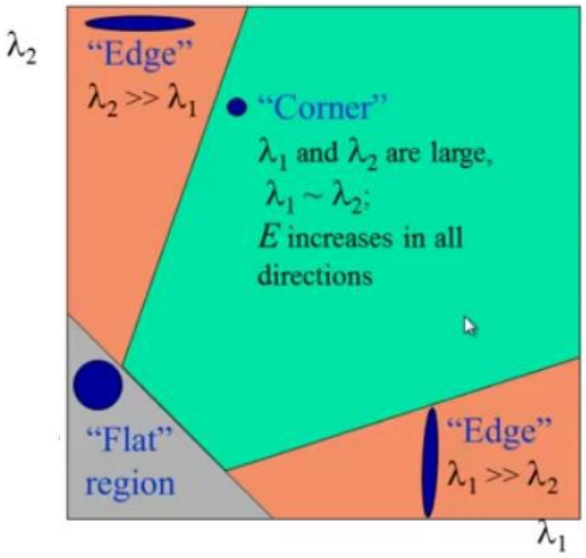

Fig. 3. Harris Feature Descriptions Based on Eigenvalues of M Matrix

Where

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (3)$$

3) Find all pixels $[u, v]$ 1. whose $E(u, v)$ exceeds a certain threshold 2. are the local maximas within their window.
4) For each pixel from step 3, use eigenvalues of its M to determine its type of feature as shown in Fig3, for example a corner, or an edge, etc.

B. Scale-Invariant-Feature-Transform (SIFT)

The SIFT algorithm [35] can be summarized as the following:

1) Scale-Space Peak Selction: Finding potential features
2) Keypoint Localization: Accurately locating feature points
3) Orientation Assignment: Assigning orientation to keypoints
4) Keypoint Descriptor: Describing the keypoints as a high dimensional vector

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{4}$$

In step 1, SIFT uses Gaussian convolution kernel (Difference of Gaussian) to construct multi-scale blurred image with key points as illustrated in Fig4. Then, in each octave of the image pyramid, SIFT generates Difference of Gaussian (DoG) images for finding interesting keypoints. This is achieved by taking the difference between two Gaussian blurred images with different σ in the Gaussian Blur Operator, as represented in Equation4. Noted that the purpose for Gaussian Blurring is to reduce DoG's sensitivity to noise, which is an approximation of Laplacian of Gaussian (LoG) [35]. In step 2, interest keypoints are localized by finding local maxima among its neighbors in three neighboring scales. Afterwards, SIFT discards edges and points that do not have enough contrast. This image-pyramid based method will help the SIFT algorithm achieve scale invariance. In step 3, the orientation of each feature point is assigned according to the gradient direction histogram, which is rotation-invariant, and is robust to the influence of noise and illumination changes. In step 4, each keypoint generates a feature descriptor with 128 histogram bins of image gradients. However, due to the high dimension of its feature descriptor, it usually takes a long time to extract features [47].

The invention of SIFT features initiated a flurry of its applications in visual SLAM. Sim et al.uses analgorithm combining SIFT features and FastSLAM filtering [68] to achieve particularly large-scale vision-only SLAM mapping. The real-time implementation of MonoSLAM [67] also employs SIFT descriptors to extract sparse natural features for localization. Despite being processor-intensive for vision only SLAM mapping [67], it is clear that invariant features such as SIFT provide a high level of performance in matching and

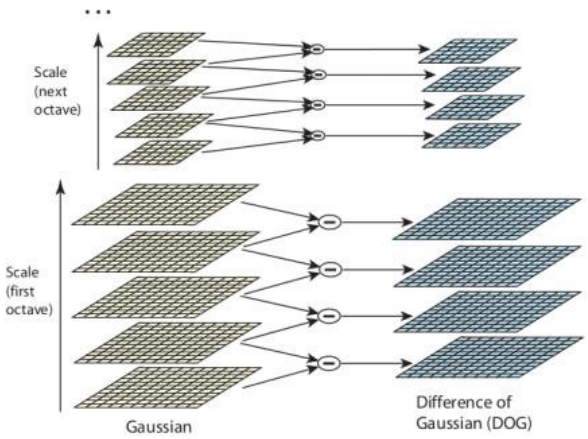

Fig. 4. Difference of Gaussian and Image Pyramid in SIFT

gives high fidelity "location recognition" in the same way as they were designed for use in visual object recognition.

C. Speeded-Up-Robust-Features(SURF)

The SURF algorithm was proposed by Bay et al. [36] as an improvement on the basis of the SIFT algorithm. There are three aspects of SURF that make it faster than its predecessor: the use of Determinant of Hessian (DoH) and box filtering to find feature points, parallelized image pyramid construction, and smaller feature descriptors. With all these factors combined, SURF can achieve 3 times the speed of SIFT and similar performance [36].

$$H(x, y) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \tag{5}$$

$$I_\Sigma((x, y)) = \sum_{i=0}^{i \le x} \sum_{j=0}^{j \le y} I(i, j) \tag{6}$$

The first factor that accelerates SURF from SIFT is the use of Determinant of Hessian (DoH) and box filtering in keypoint localization [36]. SURF combines gaussian kernel convolution and Hessian of a pixel $H(x, y)$ (Equation5) into the Hessian matrix $H((x, y), \sigma)$ (Equation7). then $H((x, y), \sigma)$ can be approximated as box filters, and the resulting Gaussian second-

order derivatives can be evaluated with low computational cost by using integral images (Equation6). The approximated box filters of Gaussian second-order derivatives are illustrated in Fig5.

$$H(x,y,\sigma) = \begin{bmatrix} L_{xx}((x,y),\sigma), L_{xy}((x,y),\sigma) \\ L_{xy}((x,y),\sigma), L_{yy}((x,y),\sigma) \end{bmatrix} \quad (7)$$

The second factor that accelerates SURF is image pyramid building. To build an image pyramid, after smoothing images with a Gaussian Kernel, SIFT sub-samples the image to achieve a higher level and use the same Gaussian Kernel to smooth it. However, with the help of box filters, SURF do not need to iteratively apply the same filter to the previously smoothed layer, but instead can apply different box filters on the original image in a parallel manner [36].

The third factor that speeds up SURF is feature description. SURF calculates the sum of Haar-wavelets responses in x and y-directions in a circular neighborhood around the keypoint, using integral images for fast computation [36]. Then SURF changes the orientation of scanning, e.g, add $pi/3$ to the x, y-direction of Haar wavelets and recalculate until the orientation with the largest sum value is found, as shown in 6. This orientation is the main orientation of the feature descriptor. Finally, a 4×4 square feature descriptor is constructed in alignment with the main orientation, and cell in the feature descriptor contains the Haar-wavelet response in x,y-directions and their absolute values. As a result, the feature descriptor of each keypoint can be flattened into a 64×1 array, which is half the size of SIFT feature descriptor.

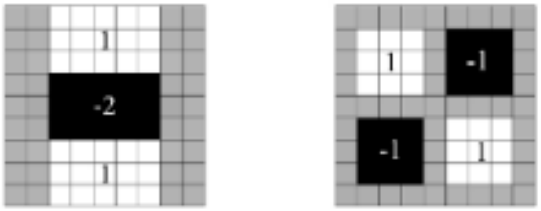

Fig. 5. Left to right: box Filters in y-direction, xy-direction as approximations for Gaussian Second order derivatives [36]

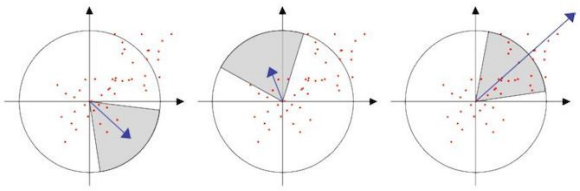

Fig. 6. Left to right: Orientation assignment in SURF is done by rotating Haar Wavelets and finding the sum of their responses [36]

D. BRIEF

Binary Robust Independent Elementary Features (BRIEF) is an efficient feature descriptor that is very popular in Visual SLAM, and it could be used in combination with interest point identification algorithms in SURF, SIFT, etc [71]. Rublee et al.combined the Features from Accelerated Test (FAST) [75] and BRIEF and proposed the Oriented Fast and Rotation algorithm (ORB) [45]. ORB was successfully applied to the famous ORB-SLAM [46], as its speed is 30-50 times faster than SIFT. However, a disadvantage of BRIEF is that it lacks scale and rotation invariance [78] and it can be only used in 2D environment.

After interest points identification, BRIEF builds feature descriptors in the following way:

1) Patch Generation: define a fixed size neighborhood of an interest point
2) Patch Smoothing: reduce sensitivity using Gaussian Kernel as information based on single pixel locations are used, which is sensitive to noise
3) Pair Selection: choose a set of pixel pairs in the patch

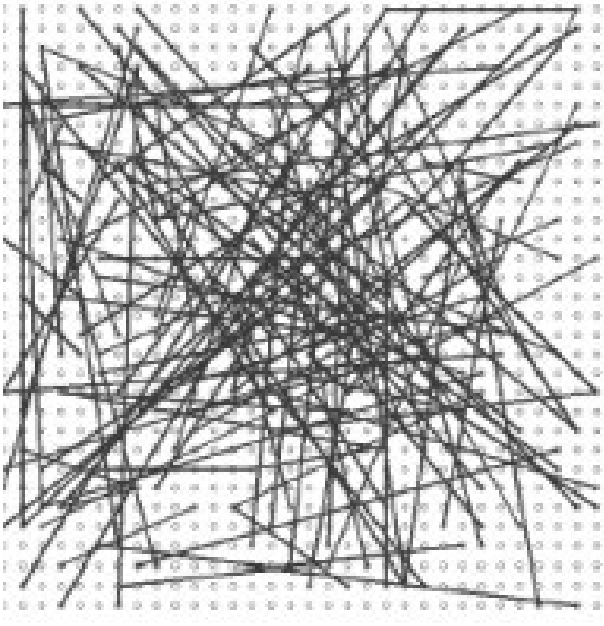

Fig. 7. Random Pairs of Pixels are Compared in BRIEF

that uniquely defines a set of binary set for intensity comparison. This is usually done in a randomized manner [71] as shown in Fig7.

4) Binary Test Intensity Comparison: given pair sequence (x,y), if $Intensity(x) > Intensity(y)$, outputs 1, else, outputs 0.

Note that in step 4, a binary test is defined in Equation 8, where $p(x)$ is the intensity value at pixel x. For the subsequent feature matching step, Calonder et al. suggest that a good distance metric to use is Hamming Distance given a set of binary test [17].

$$\tau(p;x,y) = \begin{cases} 1, & \text{if } p(x) < p(y) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

V. Feature Matching

To find feature 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, because the drift in orientation features is not significant, optical flow-based techniques(e.g., Kanade-Lucas-Tomashi (KLT) tracker [34]) can be used for matching. On the contrary, for long baselines, highly discriminative feature descriptors (e.g., SIFT [35], SURF [36], ORB [45], BRISK [38], etc.) are necessary to find correspondences. To find if two features are matched, the dissimilarity of feature vectors is calculated and compared using distance metrics. For example, for SIFT and SURF, the similarity between key points is measured by the Euclidean distance of the characterization vectors. For binary feature descriptors such as ORB and FREAK, the similarity is measured by Hamming distance.Two feature vectors are considered matched if their distance is smaller than a preset threshold.

In this section, two key feature tracking techniques are introduced: the KLT tracker and Random Sample Consensus (RANSAC). Please note that RANSAC is always used in conjunction with highly descriptive feature descriptors as presented in Section IV Feature Extraction. Also, an overview on matching search algorithms will be delivered.

A. Kanade-Lucas-Tomashi Tracker (KLT)

The KLT tracker is an opticalflow based tracker that tracks the features of a target object from first frame to last frame of a video sequence. Given two images I, J, the goal of KLT is to minimize the residual function $\epsilon(d)$, where $[d_x, d_y]$ is the optical flow at $[u_x, u_y]$.

$$\epsilon(d) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{x=u_y-w_y}^{u_y+w_y} (I(x,y) - J(x+d_x, y+d_y))^2 \tag{9}$$

Then, to find the set of $[d_x, d_y]$ that minimizes $\epsilon(d)$, $[d_x, d_y]$ are adjusted using a pyramidal approach [34]. In each level of the pyramid, given a guess on the location of the feature, one can calculate $\frac{\partial \epsilon(d)}{\partial d}$, take the first order Taylor Expansion

on $\frac{\partial \epsilon(d)}{\partial d}$, and through substituting image gradient into $\frac{\partial \epsilon(d)}{\partial d}$, one can derive Equation10 to find the best local optical flow $[d_x, d_y]$.

$$\frac{\partial \epsilon(d)}{\partial d} = 0 \approx Gd - b \tag{10}$$

Where

$$d = [d_x, d_y]$$

$$G = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{x=u_y-w_y}^{u_y+w_y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \sigma) \end{bmatrix}$$

$$b = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{x=u_y-w_y}^{u_y+w_y} \begin{bmatrix} \delta I(x,y) I_x \\ \delta I(x,y) I_x \end{bmatrix}$$

d is the final optical flow vector for the current level in pyramid, and the guess of feature location for the next level in the pyramid, can be updated with d.

In conclusion, The KLT tracker finds the best warping parameters by using the the partial derivative of the taylor expansion of the KLT residual function. The use of pyramid can effectively reduce the error caused by Taylor Expansion with large step size. In terms of robustness, it can deal with small pixel displacement and performs well for short camera baseline (optical centers of two cameras) [1].

B. Random Sample Consensus (RANSAC) with Highly Discriminative Feature Descriptors

If the camera system has long baselines, since KLT tracker is only suitable for pixel small displacement, highly discriminative (e.g., SIFT [35], SURF [36], ORB [45],BRISK [38], etc.) are necessary to find correspondences. However, using these feature-matching techniques is not sufficient for discovering correspondences robustly, especially when there are outlier features.There are two main sources of outlier features: a) point mismatching; b) moving targets in the scene, which can have a significant impact on motion estimation, and must be removed in order to obtain accurate results of motion estimation.

To exclude the outliers, robust estimators such as Random Sample Consensus (RANSAC) should also be utilized. RANSAC uses an iterative approach to estimate the parameters of a mathematical model from a set of observed data containing outliers. RANSAC [18] finds the set of matching point pair that defines the fundamental matrix, which can find a match for most points and fit most number of inliers. When determining if a feature point is an inlier or an outlier, specific distance metrices such as Sampson Distance can be used [19].

The disadvantage of RANSAC is that all the feature points of the rough matching are iteratively calculated, so the calculation efficiency is low. Therefore, faster alternatives were developed: MLESAC [39], PROSAC, etc. [40].

Specifically, to exclude outliers generated by moving targets, RANSAC could be combined with Graph-Cut method and spatial constraints. Raza et al. [61] proposed a Graph-Cut RANSAC scheme that uses an interior point judgment method, combining spatial consistency and graph cut algorithm to solve the optimization problem of interior point judgment. Graph-Cut RANSAC is more accurate than state-of-the-art methods on a range of problems, such as, line fitting, homography, affine transformation, fundamental and essential matrix estimation.

C. Visual-Inertial Feature Tracking Techniques

As an alternative to applying Highly Discriminative Feature Descriptors and RANSAC, IMU-based techniques, also known as visual-inertial feature tracking Techniques, are better suited for highly dynamic or occluded environment [62], [63]. Conventionally, in Visual-inertial SLAM, visual and inertial

information is fused to update the robot and landmark poses at the update stage. However, since inertial information helps achieve accurate short term pose estimate, it can be used to generate better feature matching results [79].

The work of Bloesch et al. in robust visual-inertial odometry (ROVIO) is an visual-inertial odometry framework that tightly fuses inertial measurements with visual data from one or more cameras, on the basis of iterated extended Kalman filter [79]. They developed a photometric error that is directly integrated as an innovation term in the filter update step using image patches as landmark descriptors [79]. The following is the general feature matching workflow and a visual illustration is presented in Fig 8

1) A patch feature (blue square) is extracted for the landmark (blue dot).
2) The estimated landmark image coordinates (yellow dot) and the corresponding covariance (yellow ellipse) are provided by the filter's IMU-driven process model.
3) Depending on the magnitude of the uncertainty, a number of candidates (yellow dots) are initialized.
4) For each candidate, an iterative update (black arrow) is performed, which integrates patch intensity errors together with the motion prior. Outlier detection and quality checks are performed to select the best valid tracking (green vs. red squares). Steps 3 and 4 are completely integrated in the iterative filter updates.

Chen et al. states that ROVIO's feature matching capability has been shown to have improved efficiency and robustness based on integrated gyroscope measurements. Also, this capability can easily extended to work in conjunction with new algorithms such as multi-threaded map building, semantic SLAM, and positioning [78].

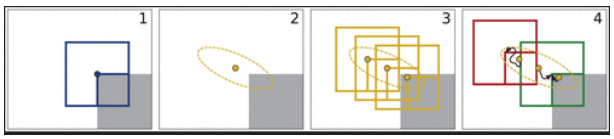

Fig. 8. Feature Tracking Using a Iterated-Kalman-Filter Based Visual-Inertial System

D. Matching Search Algorithm

An important issue involved in the feature matching is the matching search algorithm. The most common matching search algorithm is Approximate Nearest Neighbor (ANN) [47]. ANN can exchange for a significant increase in search speed at the expense of less accuracy, such as BBF, Spill-tree, Hierarchical K-means tree, and Randomized KD-trees. In addition, the Bag of Words [76] adopted by ORB-SLAM and FAB-MAP are also efficient matching search algorithms. In order to speed up the search, the certain constraints can be added to the search area, such as motion model constraints, 3D feature point position constraints and polar constraints [77].

VI. Motion Segmentation

Motion segmentation detects moving parts in the image by classifying the features into two different groups, static and dynamic [1].

Mathematically, given a set of feature points

$$W = \{x_i \in \mathbb{R}^2\}_{i=1}^{n}$$

we want to classify the features into

$$W_1 = \{x_1, x_2, ...x_m\}$$

$$W_2 = \{x_{m+1}, x_{m+2}, ...x_n\}$$

Where

$$W_1 \cap W_2 = \phi$$

To achieve motion segmentation, a variety of methods have been proposed, including background-foreground initializa-

tion, violation of geometric constraints, and violation of optical flow constraints. A short tabular summary of these methods are provided at the end of this survey in Appendix.

A. Background initialization and GPCA

The idea of background initialization is to assume that the system has prior knowledge about the environment and leverages that information to segment static and dynamic features. This prior knowledge is usually attached to background objects (static features). A similar class of approaches named "foreground initialization" which utilizes knowledge about the type or the shape of the object that moves in front of the camera. Due to high number of potential misclassifications in foreground initialization [1], background initialization is a more suitable approach for Robust Visual SLAM.

In the work of Zhang et al. about background initialization [64], a background model (static features) can be built in a similar way as background subtraction techniques [80]. First, when the visual localization is first initialized, we can assume that no foreground object is included in the scene, which allows us to initialize the background with all the detected features [64]. Then, when a new frame is processed, 3D motion segmentation is applied using Generalized Principal Component Analysis (GPCA) [65], which is the core technique in background initialization. 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.

1) GPCA: In short, Generalized Principal Component Analysis (GPCA) is able to segment an unknown number of subspaces of unknown and varying dimensions from sample data points [65]. As shown in Fig 9, one can use GPCA to find all the subspaces in the feature points, and the basis vectors describing the complement of each subspace. The first step of GPCA is to find subspaces, GPCA applies polynomial division to recursively select points in the data set that minimize their distance to the algebraic set [65]. The second step is to recover the basis of the complement of each subspace by applying standard Principal Component Analysis (PCA) [65]. In 3D motion segmentation, GPCA represents feature points of the same motion as a subspace, and finds the basis vectors for the complement of each subspace in combination with factorization:

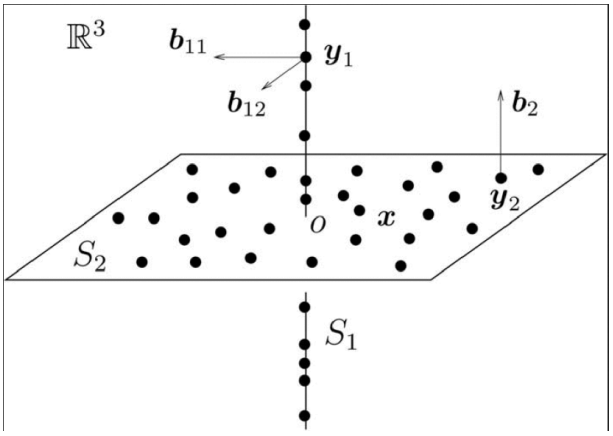

Fig. 9. GPCA is able to find an unknown number of subspaces and the basis vectors of their complements [65]

$$W = MS^T$$

where W contains n points taken across F frames, $A_f \in \mathbb{R}^{2x4}$ is the affine camera matrix for frame f

$$\begin{bmatrix} x_{11} \dots & x_{1n} \\ x_{21} \dots & x_{2n} \\ \dots & \\ x_{F1} \dots & x_{Fn} \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_F \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix}^T$$

As $rank(W) \leq 4$, each moving object spans a different d-dimensional subspace, where $d = 2, 3$,or 4. Solving the motion segmentation problem is hence equivalent to finding

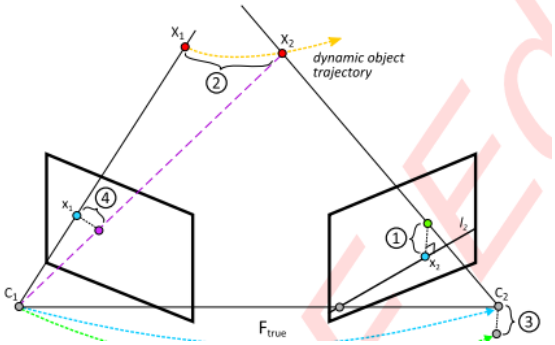

Fig. 10. Violation of geometric constraints in a dynamic environment

a basis for each one of such subspaces without knowing which points belong to which subspace. This problem can be solved by applying GPCA to the image measurements projected onto a subspace of dimension $D = d_{max} + 1 = 5$. That is, if $W = U\Sigma V^T$ is the SVD of the data matrix, then we can solve the motion segmentation problem by applying GPCA to the first five columns of V^T [65].

B. Geometric Constraints

Techniques that rely on geometric constraints leverage epipolar geometry properties to segment static and dynamic features . The constraints can be derived from the equation of epipolar lines, triangulation, fundamental matrix estimation, or reprojection error. These techniques have their roots in the fact that dynamic features will violate standard constraints defined in multiple-view geometry for static scenes.

The violation of geometric constraints in a dynamic environment comes from: (1)in the subsequent frame, the tracked feature lies too far from the epipolar line, as illustrated in Fig10. (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. A common disadvantage of geometric constraint methods is that occlusion is not being handled [1].

Kundu et al. [22] construct the fundamental matrix from robot odometry to define two geometric constraints. The main constraint is that a physically static 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 [22]. Let the image coordinates of a point in view 1 and view 2 be x_1 and x_2, the fundamental matrix of the camera model is F, then the corresponding epipolar constraint for coplanarity is $x_2^T F x_1^T = 0$. The main constraint alone will fail in degenerate motion, where the camera and the point move in the same direction [22]. Therefore, a second constraint known as Flow Vector Bound (FVB) was developed, which uses the knowledge of the robot motion to estimate a bound in the position of image pixel along the epipolar line [22]. 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 a recursive Bayes filter [22].

Migliore et al. [23] segment static and dynamic features by the principle of triangulation. They first save the viewing rays of a point from three different camera locations at different times, and classify the point as static if the three viewing rays have the same intersection, as illustrated in Fig 11. However, since the sensor measurement is noisy, Migliore et al. employ uncertain projective geometry [24] 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.

Another geometric approach is to leverage the reprojection error, the distance between the projected and the actual feature

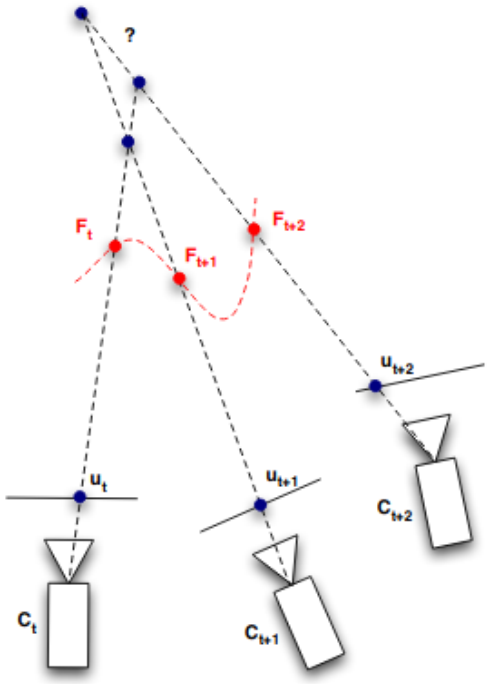

Fig. 11. Triangulation Constraint Violation by a Dynamic Point

location. In the work of Zou et al. [25], given the previous camera pose and the estimate of the current camera pose, features from the previous frame can be projected into the current frame using triangulation, and the reprojection error can be calculated with a distance metric, as shown in Fig 12 [25]. Features with a small reprojection error are regarded as static, while the ones with a large reprojection error is regarded as ”uncertain”, which could be either false correspondences or dynamic features. For an uncertain point, it could be further classified by triangulation using multiple cameras and checking if the viewing rays can intersect [25]. A good choice for the distance metric in reprojection error calculation is the Mahalanobis distance, as this takes into account the uncertainty of the camera pose as well as the actual Euclidean distance [25]. This method is robust to false correspondence, but is not robust to occlusion.

Similar to the work of Zou et al., Tan et al. [17] use the projection principle to detect dynamic features. However, they

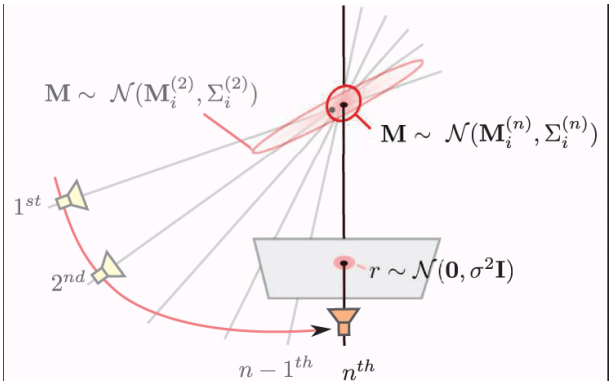

Fig. 12. Reprojection Error Calculation With Mahalanobis Distance

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 [17]. 3D points occluded by those conditions will be kept and used to robustly estimate the camera pose.

C. Optical Flow

The definition of optical flow is the apparent motion of brightness patterns computed from two consecutive images [26]. Although this definition does not correspond to the motion field in an image, optical flow might be used to define a constraint that loosely identifies moving objects. It should be noted that computing optical flow on every pixel is computationally expensive, which is a disadvantage for this family of methods.

Klappstein [27] defined the likelihood of a moving object based on to what extent the optical flow from the previous image is violated. In a static scene, all flow vectors are parallel to the corresponding epipolar lines and point away from the epipole (focus of expansion) [27]. If a measured optical flow vector deviate from this expected flow direction (i.e., if the

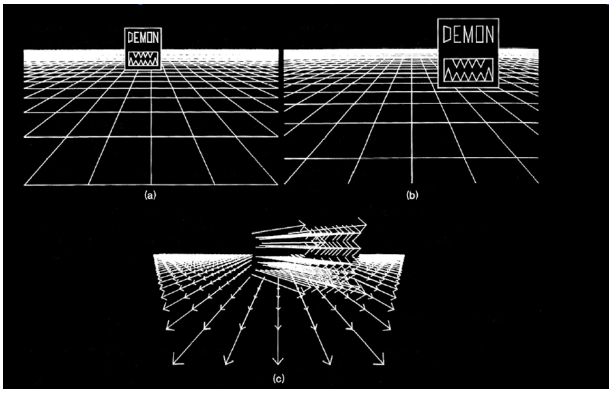

Fig. 13. violation of the Optical Flow Constraint by a Moving Object

angle between the measured and the expected directions is not zero), the corresponding 3D point is moving, as illustrated in Fig13. Optical Flow constraint actually explores a subset of constraints on mono-view reprojection, positive depth, positive height, and the aforementioned triangulation constraints proposed by Migliore [27]. The mono-view constraint states that viewing rays from two frames must meet, the positive depth constraint states that if viewing rays of two frames intersect behind the camera, the actual3D point must be moving, and the positive height constraint states that if viewing rays intersect underneath the road the actual 3D point must be moving [27]. A motion metric for each point can be calculated to measure the extent to which the optical flow constraint is violated. Finally, Klappstein uses a graph-cut algorithm is utilized to segment the moving objects [27].

Alcantarilla et al. [66] segment moving objects based on the modulus of the 3D motion vector in scene flow (3D version of optical flow) through ”residual motion likelihood” using Mahalanobis distance. Residual motion likelihood takes into account: 1. difference between the 3D positions of two corresponding image points, considering the images from two cameras are rectified and undistorted; 2. measurement uncertainty in computing scene flow based on dense optical flow

Fig. 14. Derome et al. Optical Flow Workflow For Motion Detection

and stereo reconstruction. If the residual motion likelihood is low, the feature point most likely belongs to a static object. By thresholding on the residual motion likelihoods, the feature points that reside on a moving object can be deleted from the SLAM process, making visual odometry estimation more robust.

Derome et al. [30], [31] compute optical flow at the previous time frame by calculating the residual between the predicted image with the observed image from a stereo camera. By processing backward in time, the predicted image at the previous time frame 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. The workflow of Derome’s method is illustrated in 14

VII. Localization

Localization refers to the estimation of relative camera pose (translation t and rotation R) from multiple images. Standard visual SLAM achieves this by implementing epipolar geometry on feature correspondences [19]. In Robust Visual SLAM, instead of computing the camera pose from all feature correspondences, only static features resulting from techniques described in the Motion Segmentation Section are employed. All dynamic features are regarded as outliers and excluded from the computation. As a side note, deep learning techniques can process the image sequences directly without computing

feature correspondences. Due to the very distinctive nature of deep learning in the computer vision community, this section discusses only feature-based techniques and would like to discuss deep-learning-based approaches as part of future work.

In some robust visual SLAM techniques, localization and 3D reconstruction of the scene are usually processed together. However, this survey decomposes this generally holistic step into two individual entities based on their functional differences.

1) Filters and Graph Optimization: Many Visual SLAM Methods apply probabilistic filter based methods, and the typical representatives are Extended Kalman Filter SLAM (EKF-SLAM), Unscented Kalman Filter SLAM (UKF-SLAM), and Rao-Blackwellized Particle Filtering SLAM (RBPF) [47].One milestone work, Monocular SLAM (MonoSLAM) in early 2000s by Davison et al. uses image features (local image patches) to represent landmarks in the map, iteratively updates the probability density of the feature depth by frame-to-frame matching to recover their 3-D positions and hence initialize feature-based sparse map, and updates the full state vector (robot pose plus feature 3-D locations) within an Extended Kalman Filter (EKF) framework [47]. Davison's work in some way set the standard framework for traditional Bayesian filtering based visual SLAM implementation.

Graph optimization uses the observation data of the current time and the previous time to estimate robot pose at all times, and describe state estimation problem in the form of graphs. Graphs can visually represent the relationship between optimization variables and the error term, and can apply to large-scale environments [47]. After data association and loop closure detection, the optimization problem needs to be modeled. Finally, back end optimizes the graph model which has noise to get the global trajectory and map [47].

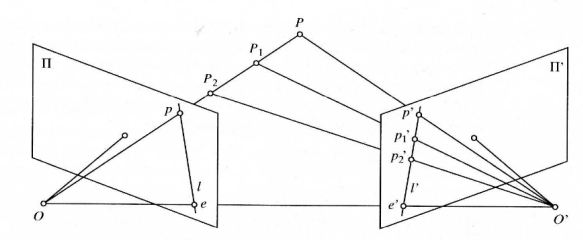

Fig. 15. Illustration of Epipolar Constraints

A. Geometric Methods

Once the image correspondences are known, the relative pose between two or three images can be recovered with one free variable up to a scale. One way to recover robot pose is through enforcing the epipolar constraint. If the robot has two cameras on board, the pose from two views can be computed by the 8-point [4] or the 5-point algorithm [28]. If the robot has three cameras, the trifocal tensor [29] can be utilized for better overall accuracy [69]. 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 [33]). A great advantage of geometric methods is camera pose can be computed without prior knowledge of the internal and external camera matrices. A disadvantage is that these methods require more computing to find the best image correspondences (e.g, using RANSAC) than methods that uses external constraints to achieve a simplified fundamental matrix.

1) 8-Point And 5-Point Algorithms: The 8-point algorithm was first developed by Longuet et al. using the epipolar constraint illustrated in Fig15 [4]. The primary motivation for the 8-point algorithm is that given a point in one image, multiplying by the essential/fundamental matrix will tell us which epipolar line to search along in the second view. In 2004, Nister et.al proposed the 5-point algorithm to achieve a similar performance with less computation time. Given known static feature correspondences, one can recover camera pose

by solving for fundamental matrix using the 8-Point or 5-Point methods. The advantage of these methods is that there is no constraint on how the camera moves.

In short, the 8-point algorithm selects 8 point correspondences between camera views, and enforces epipolar constraint $x^T F x = 0$ to evaluate the fundamental matrix$F$. In Fig 15, lines $Op, O'p', OO'$ are coplanar, and correspondingly, we can derive $p^T[T \times \dot{R}p'] = 0$. It is not difficult to see that essential matrix can be represented as $E = T \times R$, and when one camera's projection matrix is $P = [I0]$, the essential matrix and the fundamental matrix of the camera pair are the same. Then, to solve for the fundamental matrix$F$, we set one entry as a free variable and we can write out the epipolar constraint with 8 selected point correspondences' image coordinates $[u_i, v_i]$.

$$\begin{bmatrix} u_1u'1 & u_1v'_1 & u_1 & v_1u'_1 & v_1v'_1 & v_1 & u'_1 & v'_1 & 1 \\ \vdots & & & & & & & & \\ u_8u'_8 & u_8v'_8 & u_8 & v_8u'_8 & v_8v'_8 & v_8 & u'_8 & v'_8 & 1 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ \vdots \\ F_{32} \\ 1 \end{bmatrix} = 0$$

Given W, we can find a least square solution of F using Singular Value Decomposition (SVD).

The key idea of 5-point algorithm is to have fully-calibrated cameras so that the essential Matrix can be used for motion estimation [28]. In that case, an essential matrix is a faithful representation of 3D motion (translation, rotation) up to a scale, and it has only 5 DOFs. Also, it must satisfy two constraints, which can be characterezied as 2 linearly independent equations in the following relationship:

$$2EE^TE - tr(EE^T)E = 0 \tag{11}$$

Furthermore, a valid fundamental matrix F must follow the cubic singularity condition: $det(F) = 0$. Therefore, given 5 points, we have 5 epipolar equations, 2 equations from constraints, and 1 equation from cubic singularity, which are enough for estimating the essential matrix.

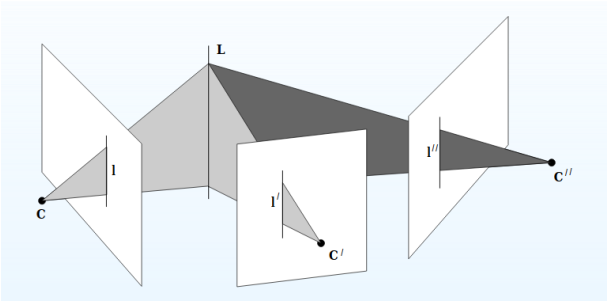

Fig. 16. Line Incidence in Trifocal Tensor

2) Trifocal Tensor: Trifocal Tensor is the three-camera equivalent for the fundamental matrix in a two-camera system [69]. It captures the complete projective geometric relations between the three views, and is uniquely defined by the internal and external camera properties. Like other geometric methods, it can be computed directly from image correspondences without the knowledge of the internal and external camera matrices.

As a short summary, the core principle of trifocal tensor is line incidence as depicted in Fig16, i.e, the back-projection of the line in three camera views should be the same line [29]. First, we can write out the projection matrix of each camera P_1, P_2, P_3 using the camera relative positions. Second, we back-project images of a line from each camera view, l, l', l'' and we are able to derive the back-projection planes π, π', π'' with P_1, P_2, P_3 and l, l', l''. Third, since the back-projected projected planes coincide, any back-projected plane is a linear combination of the other two. This relationship finally helps us represent l, l', l'' using trifocal tensors $[T_1, T_2, T_3]$, and each trifocal tensor T_i is can be represented by the relative translation of cameras and the column i of the relative rotation between cameras [29].

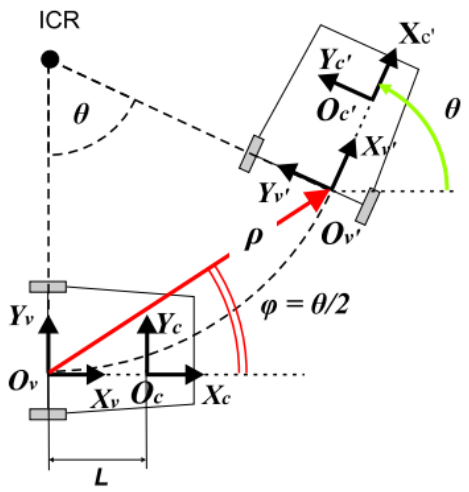

Fig. 17. Differential Drive Ego Motion Constraint

B. Localization with Nonholonomic Ego-Motion Constraints

Many mobile bases, such as cars and differential-drive robots, use typical wheels that do not slip sideways. The constraints that these wheels impose onto the robots are defined as nonholonimic constraints, which refers to velocity constraints that cannot be integrated into pose configurations [70]. In robotics, differential drive constraints and Ackermann Steering constraints are two very common non-holonomic constraints, as illustrated in Fig17 and Fig18 [70].

As previously presented, Standard SfM and visual SLAM compute the motion of the camera by means of the 8-point [4] or the 5-point algorithms, which have their roots in the epipolar constraint for co-planarity [28]. This general ego-motion estimation is calculated without making any assumption on how the camera moves. Another way to estimate the camera pose is to include an additional assumption that the camera moves according to particular parameterization given external information, for example wheel odometry information. By enforcing nonholonomic ego-motion constraints, classifying static features can be achieved by fitting feature points that match with the camera motion constraints. A great advantage of ego-motion based localization techniques is that they greatly reduces amount of computation for finding best point correspondences. The disadvantage, however, is that they are very specific to the type of nonholonomic constraint the robot is subject to.

In addition to the epipolar constraint, Scaramuzza [30] proposed to use nonholonomic constraints of differential-drive robots 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, if frames are taken at a fast enough rate, planar vehicle motion between two frames can be modelled as rotation with varying radius and center. These parameters can be estimated by applying the non-holonomic constraints on differential drive or Ackermann Steering with instantaneous wheel odometry [30]. Correspondingly, the essential matrix can be simplified down to two parameters given the rotation radius and center [30]. The feature point that best satisfies the estimated camera motion are considered as a static feature, and by applying the 1-point algorithm [31], one can calculate both rotation parameters.

The ego-motion constraint can be used further for estimating eoru motion, which is defined as the combination of the robot ego motion and 3D multi-object motion [32]. The first step is to estimate the camera pose through one-point algorithm [31]. The second step is to generate hypotheses of the multi-object motions using two-point algorithm [48] through a multi-RANSAC like scheme. In each hypotheses, there are multiple motions and one motion corresponds to one group of points [32]. As the final step, each hypotheses is evaluated by comparing reprojection errors of all their points with respect to their motions, and the hypotheses with the smallest reprojection error is chosen to be the best hypotheses [32]. Compared with Multibody-SfM methods, the ego-motion constraint method does not require initial motion segmentation between static and dynamic features.

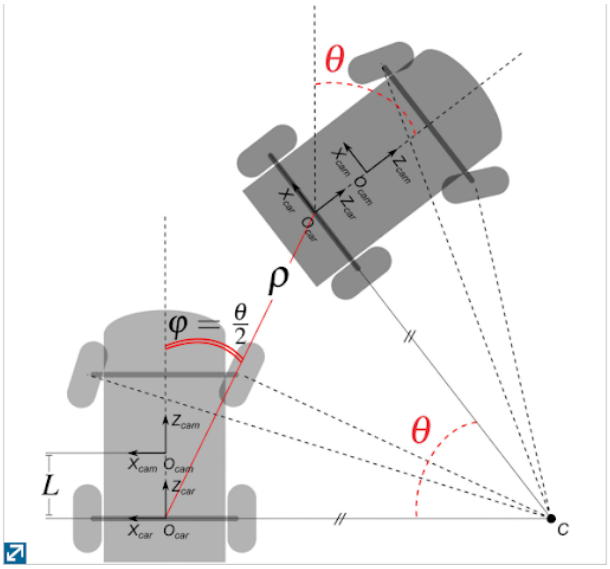

Fig. 18. Ackermann Steering Ego Motion Constraint

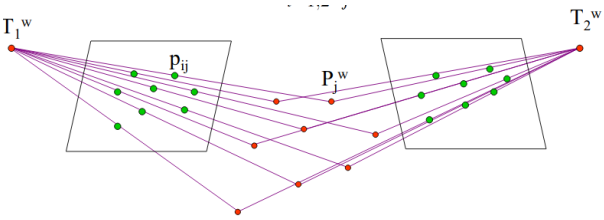

Fig. 19. Bundle Adjustment

1) Overview of the Mathematical Principle of Bundle Adjustment: As illustrated in Fig19, after achieving point correspondences between two camera frames, one can achieve a set estimates of robot pose and 3D point coordinates. This set of estimate should minimize the reprojection error E , which is a function of 1. SE(3) world frame pose of the two cameras T_i^w, 2. 3D point world frame coordinates P_j^w, and 3.image coordinate of jth point in ith image frame p_{ij}.

$$E(T_1^w, T_2^w, \{P_j\}) := \sum_{i=1,2} \sum_j ||h(T_i^w, P_j^w) - p_{ij}|| \quad (12)$$

Where $h(T_i^w, P_j^w)$ is the projection of P_j^w onto image frame i, therefore $h(T_i^w, P_j^w)$ can be expanded as a function of the rotation R_i^w and translation t_i^w of camera frame i in the world

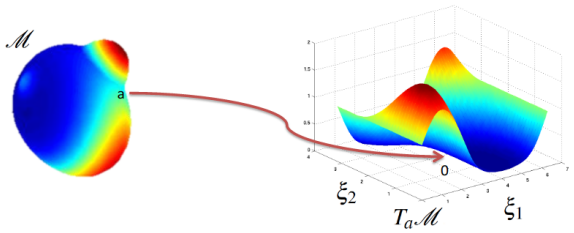

Fig. 20. Lie(3) Manifold and its Tangent Space

frame:

$$\hat{p} = h(T_i^w, P_j^w) := \pi((R_i^w)^T(P^w - t_i^w)) \quad (13)$$

Note that Equation13 is not directly linearizable as T_i^w forms a Lie(3) Group Manifold M, and any perturbation twist ξ_i should be converted to its exponential form e^{ξ}. Therefore, to linearize Equation13, it is necessary to define generalized Taylor Expansion that belongs to the tangent space of M, $f(ae^{\xi}) \approx f(a) + f'(a)\xi$ as illustrated Fig20. Then, the $h(T_i^w, P_j^w)$ can be approximated as:

$$h(T_c^w e^{\xi}, P^w + \sigma) \approx h(T_i^w, P_j^w) + h_1(T_c^w, P^w)\xi + h_2(T_c^w, P^w)\delta \quad (14)$$

where ξ is a 6×1 perturbation twist to T_i^w, $h_1(T_c^w, P^w)$ is the 2×6 Jacobian matrix of M evaluated at the current camera pose, $h_2$is the projection matrix evaluated at the current camera pose, δ is a 2×3 perturbation to the 3D point world coordinates.

With a proper linearized model of $E(T_1^w, T_2^w, \{P_j\})$, one can represent Equation13 in the form of:

$$E(T_1^w, T_2^w, \{P_j\}) =:= \sum_{i=1,2} \sum_j ||Ax - b|| \quad (15)$$

Finally, this error function can be optmized using Gauss-Newton or Levenberg-Marquardt Optimization Techniques.

2) Overview of Gauss-Newton and Levenberg-Marquardt Optimization: Gauss-Newton optimization iteratively linearize, solve normal equations on tangent space, and update Lie group elements. The basic procedure is as follows:

1) Start with a good initial estimate $\theta^0 = T_1^w, T_2^w, P_j$
2) Use Equation 14 to obtain A and b for 15
3) Solve for x= ξ_1, ξ_2, δ_j using the normal equations

$$(A'A)x = A'b$$

we get the perturbation that minimizes $E(T_1^w, T_2^w, \{P_j\})$ is $x = (A'A)^{-1}A'b$, Where $A'A$ is the Gauss-Newton approximation
4) Update the nonlinear estimate θ^{t+1} with $T_1^w \leftarrow T_1^w e^{\xi_1}$, $T_2^w \leftarrow T_2^w e^{\xi_2}$, $P_j \leftarrow p_j + \delta_j$
5) If θ^{t+1} does not converge, go back to step 2

When perturbation ξ_i or δ_i is large, the Taylor approximation might be inaccurate so that the error $E(T_1^w, T_2^w, \{P_j\})$ might not decrease. On the other hand, the gradient descent method guarantees decrease but might be slow. Therefore, as an enhancement of Gauss-Newton Optmization, Levenberg-Marquardt (L-M) Optimization uses a combination of both.

$$x = (A'A + \lambda I)^{-1}A'b \tag{16}$$

When λ is small, Equation16 behaves close to Gaussian-Newton Optimization. When λ is large, $A'A$ can be neglected and perturbation x is small. In the context of Bundle Adjustment, the following basic procedure can be adopted for L-M Optimization:

1) Start with a good initial estimate $\theta^0 = T_1^w, T_2^w, P_j$, and λ
2) Use Equation 14 to obtain A and b for Equation15
3) Solve for perturbation x= ξ_1, ξ_2, δ_j using the normal equations

$$(A'A + \lambda I)x = A'b$$

Where $A'A$ is the Gauss-Newton approximation. If $E(T_1^w e^{\xi_1}, T_2^w e^{\xi_2}, \{p_j + \delta_j\}) > E(T_1^w, T_2^w, \{P_j\})$, increase$\lambda$ and repeat step3. Else, decrease λ and proceed to step 4
4) Update the nonlinear estimate θ^{t+1} with $T_1^w \leftarrow T_1^w e^{\xi_1}$, $T_2^w \leftarrow T_2^w e^{\xi_2}$, $P_j \leftarrow p_j + \delta_j$
5) If θ^{t+1} does not converge, go back to step 2

To summarize, Gauss-Newton optimization is the ground framework for optimizing the reprojection error $E(T_1^w, T_2^w, \{P_j\}) =:= \sum_{i=1,2}\sum_j ||Ax - b||$. L-M optimization is an improvement of Gauss-Newton Optimization with variable step λ that guarantees to decrease the re-projection error with a suitable perturbation to camera and 3D structure variables.

VIII. Conclusions

In the past few decades, Robust Visual SLAM techniques have been a focus of research and significant progress have been made. The SLAM algorithm based on feature points are steadily increasing in terms of calculation speed and accuracy of feature extraction and matching. This survey highlights the some of the existing approaches in each key step of Robust Visual SLAM. We have classified approaches according to the type of problem they solve and their corresponding applications. The main goal of each step and various approaches to achieve the goal have been analyzed and compared for their advantages and disadvantages.

Further research is needed to enable practical implementations of simultaneous localization and reconstruction in dynamic environment. In highly dynamic environment, Robust Visual SLAM is prone to failure due to misclassification of dynamic and static features, as well as point mismatches between

two frames or two camera views. Also, specular reflection, illumination variation will greatly degrade image characteristics for data association. Furthermore, handling missing, noisy, and outlier data is always a future challenge for most of the discussed techniques. Although statistics-based techniques can tackle this problem due to their recursive sampling approach, they have to trade off accuracy for computation cost. Also, most techniques also have difficulty in dealing with degenerate and dependent motion.

Another challenge is real-time implementation for dynamic object segmentation and 3D tracking, due to their high computational cost and the offline nature of the algorithms. 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. Recently, the availability of RGB-D Cameras enlightens a new path to more accurate feature extraction. Meanwhile, with LSD-SLAM and DSO being proposed, Visual SLAM based on direct method has also get more attention, which partly compensates for the defects of the feature point method. Moreover, collaborative Visual SLAM with multiple cameras has also shown great potential for better feature extraction in applications such as swarm robotics.

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. Also, deep-based approach uses object information to construct semantic maps, and expands the application range of SLAM systems. Although these methods have not yet become mainstream, with more and more further research, the VSLAM algorithm will reach a new height and will play a vital role in the fields of robots, virtual reality, and unmanned aerial vehicles.

IX. APPENDIX

Summary of Motion Segmentation Approaches [1]		
Name of Approach and Author	Camera	Practical Considerations
Optical Flow Derome et al.	SfM, S, P	RT, NP, TU, OH
Optical Flow Klappstein et al.	SfM, S/M, P	NT, NP, TU
Optical Flow Alcantarilla et al.	SfM, S, P	RT, OT, NP
Geometric Constraints Migliore et al.	Filter, M, P	RT, OT NP, TU
Geometric Constraints Zou et al.	SfM, M, P	RT,OT, NP, TU
Geometric Constraints Tan et al.	SfM, M, P	RT, OT, NP, OH
Geometric Constraints Kundu et al.	SfM, M, P	RT, OT, NP, TU, DM
Background Initialization Zhang et al.	SfM, M, P	RT, NB, OT,TS,TU

Note: This table is a summary of motion segmentation methods explored in Section VI. All methods in this table can be implemented on a mobile base. Under Camera tab, S is Stereo Camera, M is Monocular Camera, P is Perspective Model. Under Practical Consideration Tab, RT is real time,

NP is No prior Knowledge (e.g, background information, camera motion), TU: Takes into account uncertainty, OH: occlusion handling, OT: Handling outliers due to false feature correspondences, NT: Near real time. NB: Need to be batched

ACKNOWLEDGMENT

This work is under the supervision of Dr. Ying Wu and is supported by the Master of Science in Robotics Program at Northwestern University, Evanston, Illinois, United States

REFERENCES

[1] 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.

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

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

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

[5] 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.

[6] 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.

[7] Terrance E. Boult and Lisa Gottesfeld Brown. 1991. Fa ctorization-based segmentation of motions. In IEEE Work. Vis. Motion.

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

[9] 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.

[10] Noah Snavely, Steven Seitz, and Richard Szeliski. 2006. PhotoTourism: Exploring photo collections in 3D. In SIG-GRAPH Conf. Proc. 835–846.

[11] 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.

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

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

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

[15] 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.

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

[17] 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.

[18] 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

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

[20] 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.

[21] 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.

[22] Abhijit Kundu, K. Madhava Krishna, and Jayanthi Sivaswamy. 2009. Moving object detection by multi-view geometric techniques from a 1 single camera mounted robot. In IEEE/RSJ Int. Conf. Intell. Robot. Syst. 4306–4312.

[23] 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.

[24] 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

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

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

[27] 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.

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

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

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

[31] 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

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

[33] 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.

[34] 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.

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

[36] 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.

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

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

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

[40] 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.

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

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

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] A. Li, X. Ruan, J. Huang, X. Zhu and F. Wang, "Review of vision-based Simultaneous Localization and Mapping," 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 2019, pp. 117-123.

[48] D. Ortin and J. Montiel, "Indoor robot motion based on monocular images," Robotica, vol. 19, no. 3, pp. 331–342, 2001.

[49] M. Zuliani, C. S. Kenney, and B. Manjunath, "The multi-RANSAC algorithm and its application to detect planar homographies," in Proc. IEEE Int. Conf. Image Process., 2005, vol. 3, pp. 150–153.

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

[51] T. Shen, Z. Luo, L. Zhou, H. Deng, R. Zhang, T. Fang, and L. Quan, "Beyond Photometric Loss for Self-Supervised Ego-Motion Estimation," 2019 International Conference on Robotics and Automation (ICRA), 2019.

[52] R. Mur-Artal and J. D. Tardos. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras.IEEE Transactions on Robotics, 2017.

[53] G. Klein and D. Murray. Improving the agility of keyframe-based slam. InECCV, 2008.

[54] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun. Dense monocular depth estimation in complex dynamic scenes. InCVPR, 2016.

[55] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon.Bundle adjustment a modern synthesis. InInternational workshop on vision algorithms, 1999.

[56] David Nistér, Oleg Naroditsky, and James Bergen. 2004. Visual odometry. In IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.

652–659.

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

[58] P. F. Alcantarilla, A. Bartoli, A. J. Davison, "KAZE features," in 12th European Conference on Computer Vision, 2012, pp. 214-227.

[59] P. F. Alcantarilla, J. Nuevo, A. Bartoli, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," International Conference on Control, Automation and Systems, 2013, pp. 704-709.

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

[61] M. A. Raza, M. Awais Rehman, I. M. Qureshi and M. Habib Mahmood, A Simplified approach to Visual Odometry using Graph Cut RANSAC," 2018 5th International Multi-Topic ICT Conference (IMTIC), Jamshoro, 2018, pp. 1-7.

[62] 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.

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

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

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

[66] 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.

[67] A. J. Davison, I. D. Reid, N. D. Molton, "Mono-SLAM: real-time single camera SLAM," IEEE transactions on pattern analysis and machine intelligence, vol. 29, no. 6, pp. 1052 1067, 2007.

[68] R. Sim, P. Elinas, M. Griffin, and J.J. Little, "Vision-Based SLAMUsing the Rao-Blackwellised Particle Filter,"Proc. IJCAI WorkshopReasoning with Uncertainty in Robotics,2005.

[69] L. F. Julià and P. Monasse, "A Critical Review of the Trifocal Tensor Estimation," Image and Video Technology Lecture Notes in Computer Science, pp. 337–349, 2018.

[70] K. M. Lynch and F. C. Park, Modern robotics: mechanics, planning, and control. Cambridge (Reino Unido): Cambridge University Press, 2018.

[71] Bill Triggs, Philip Mclauchlan, Richard Hartley, Andrew Fitzgibbon. Bundle Adjustment – A ModernSynthesis. International Workshop on Vision Algorithms, Sep 2000, Corfu, Greece. pp.298–372, 10.1007/3-540-44480-721, inria-00548290

[72] H. P. Gavin,"[PDF] Levenberg-Marquardt method for nonlinear least squares curve-fitting problems c ©: Semantic Scholar," [PDF] The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems c — Semantic Scholar, 01-Jan-1970 [online] Available: https://www.semanticscholar.org/paper/The-Levenberg-Marquardt-method-for-nonlinear-least-Gavin/1d83aa91810162891c9f04e9a62da69197c78602. Accessed: 16-Mar-2020.

[73] Z. Zhang and Y. Shan. Incremental motion estimation through local bundle adjustment.Technical Report MSR-TR-01-54, Microsoft Research, May 2001.

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

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

[76] D. Galvez-López and J. D. Tardos, "Bags of Binary Words for Fast Place Recognition in Image Sequences," in IEEE Transactions on Robotics, vol. 28, no. 5, pp. 1188-1197, Oct. 2012.

[77] Z. CHEN, O. LAM, A. Jacobson, et al, "Convolutional neural network-based place recognition," Computer Science, 2014.

[78] C. Chen, H. Zhu, M. Li, and S. You, "A Review of Visual-Inertial Simultaneous Localization and Mapping from Filtering-Based and Optimization-Based Perspectives," Robotics , vol. 7, no. 3, p. 45, 2018.

[79] Bloesch, M.; Burri, M.; Omari, S.; Hutter, M.; Siegwart, R. Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback. Int. J. Robot. Res. 2017, 36, 1053–1072.

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