Perception-aided Visual-Inertial Integrated Positioning in Dynamic Urban Areas

Xiwei Bai, Bo Zhang, Weisong Wen, Li-Ta Hsu*
The Hong Kong Polytechnic University
Kowloon, Hong Kong
Correspondence: lt.hsu@polyu.edu.hk

Huiyun Li
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
Shenzhen, China
hy.li@siat.ac.cn

Abstract— **Visual-inertial navigation systems (VINS) have been extensively studied in the past decades to provide positioning services for autonomous systems, such as autonomous driving vehicles (ADV) and unmanned aerial vehicles (UAV). Decent performance can be obtained by VINS in indoor scenarios with stable illumination and texture information. Unfortunately, applying the VINS in dynamic urban areas is still a challenging problem, due to the excessive dynamic objects which can significantly degrade the performance of VINS. Detecting and removing the features inside an image using the deep neural network (DNN) that belongs to unexpected objects, such as moving vehicles and pedestrians, is a straightforward idea to mitigate the impacts of dynamic objects on VINS. However, excessive exclusion of features can significantly distort the geometry distribution of visual features. Even worse, excessive removal can cause the unobservability of the system states. Instead of directly excluding the features that possibly belong to dynamic objects, this paper proposes to remodel the uncertainty of dynamic features. Then both the healthy and dynamic features are applied in the VINS. The experiment in a typical urban canyon is conducted to validate the performance of the proposed method. The result shows that the proposed method can effectively mitigate the impacts of the dynamic objects and improved accuracy is obtained.**

Keywords—Visual Odometry; INS; VINS; Navigation; Positioning; Urban Areas

I. INTRODUCTION

Robust and accurate positioning is undoubtedly one of the most fundamental factors for a wide range of applications, such as unmanned aerial vehicles (UAV) [1] and autonomous driving vehicles (ADV) [2]. The visual-inertial navigation systems (VINS) [3-6] is extensively studied in the past decades. Different from the expensive LiDAR-based positioning [7, 8], the VINS is cost-effective, light-weight and almost ubiquitous for autonomous systems. Accurate and high-frequency positioning [3] can be obtained using VINS in indoor scenes with sufficient texture information and stable illumination conditions. However, the performance of VINS can be significantly degraded in dynamic outdoor scenes [9, 10]. This is mainly caused by unexpected dynamic objects, such as the double-decker bus, cars and even moving pedestrians. In fact, the performance of VINS relies heavily on the assumption that the surrounding visual features are static which is hard to be satisfied in dynamic environments such as city downtowns. Fig. 1 shows a typical scene of urban areas in Hong Kong with numerous unexpected dynamic participants. According to a recent review paper [11], the development of VINS is extensively discussed. However, the visual system based navigation in dynamic environments is still a challenging problem. To mitigate the impacts of dynamic objects on VINS, the major researches can be divided into two groups [12]: (1) detecting the dynamic objects using deep learning-based method [13], and removing the features that belong to the dynamic objects from VINS and, (2) detecting the dynamic objects using conventional motion tracking [14] or sampling-based methods [15], and removing the features belong to the dynamic objects as well.

The deep learning-based semantic segmentation method is employed to segment the pixels belong to the dynamic objects in [13], then the dynamic part is removed from the image. The Deepvo [16] proposed an end-to-end visual positioning method to mitigate the effects of dynamic objects. The Detect-SLAM [17] proposed an object detector based on a deep neural network (DNN) to recognize moving objects, then remove unreliable features. The motion tracking-based approach [18] is studied to detect the features belong to the dynamic objects. [15] and [19] proposed to make use of the random sample consensus (RANSAC) algorithm to remove outliers (dynamic features). Improved performance is obtained after excluding the outliers. In addition, [20] presented a motion segmentation method to track moving objects to further remove the outliers from dynamic objects.

In short, the existing methods to cope with dynamic objects tends to directly exclude the features from dynamic objects. In fact, the performance of visual odometry relies heavily on the distribution of the features [12]. Unfortunately, the excessive exclusion of features can severely distort the geometry distribution of features with respect to the VINS system. In addition, in our previous research [21], the majority of the features can even arise from dynamic objects in some cases (e.g. see Fig. 1). Fully exclusion of dynamic features can even significantly degrade the observability [11] of the system state. Therefore, excluding all the dynamic features is not acceptable in dynamic urban scenarios. Interestingly, similar positioning problems can also be seen in global navigation satellite systems (GNSS) which based on the signals received from multiple satellites. The non-line-of-sight (NLOS) receptions are similar to the dynamic feature

978-1-7281-0244-3/20/$31.00 ©2020 IEEE

points in visual-based positioning, the VINS, as both of them belong to the unhealthy or polluted measurements. Fully exclusion of NLOS satellites will severely distort the geometry distribution of satellites, the horizontal dilution of precision (HDOP), and even cause a lack of satellites for GNSS positioning in deep urban areas [22-26]. Instead of excluding the NLOS, our latest research in [27] proposed a method that remodels the uncertainty of NLOS satellites with smaller weighting in GNSS positioning.

Inspired by this, we propose to mitigate the effects of dynamic objects by remodeling the uncertainty of dynamic feature point (DFP) using a novel weighting scheme to achieve adaptively tuning of visual measurement model, after detecting the dynamic objects (perception) using the state-of-the-art deep neural network (DNN), the YOLO (you only look once) network [28]. Firstly, we use the monocular camera to detect moving objects that are represented by bounding boxes, then the DFP within a bounding box with respect to a moving object can be recognized. Secondly, we remodel the uncertainty of DFP based on a novel weighting scheme. According to our research in [21], the feature being tracked for more times is likely to be a healthy static one. Therefore, we propose to estimate the weighting of the feature by combining both the object detection result and the quality of the vision feature being tracked. As a result, DFP could also still be used accordingly. Thirdly, a state-of-the-art pre-integration method is employed to get the transformation between consecutive frames using raw measurements from the inertial navigation system (IMU) [29]. Then the improved visual measurement factor is derived based on the derived weighting scheme. Finally, we make use of the Google Ceres solver [30] to solve the factor graph optimization of VINS.

The remainder of this paper is structured as follows. An overview of the proposed method is given in Section II. Section III presents the visual-inertial integration using factor graph optimization. The self-tuning visual measurement modeling and its application in VINS are described in Section IV before the experimental evaluation is presented in Section V. Finally, the conclusions and future work are drawn in Section VI.

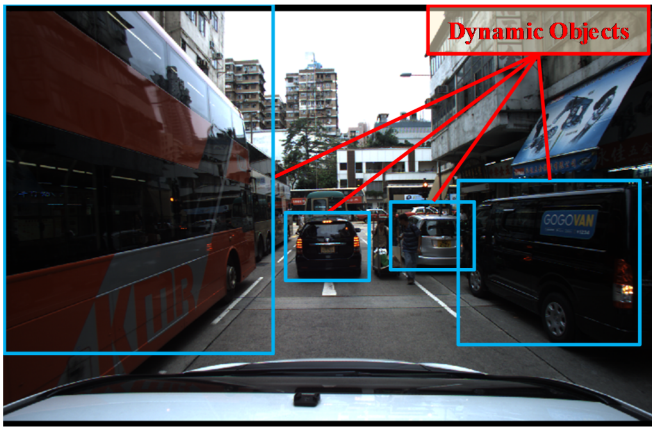


Fig. 1. Illustration of a typical urban scene in Hong Kong with numerous dynamic objects.

II. Overview of the Proposed Method

The overview of the proposed VINS integration is shown in Fig. 2. The inputs of the system include the measurements from both an IMU and a camera. The state-of-the-art pre-integration algorithm [31] is employed to integrate the high-frequency IMU measurements, the linear acceleration and gyroscope measurements. Then the IMU factor is obtained based on the pre-integration. Regarding the image processing pipeline, the feature extraction is performed to detect representative visual features. The feature tracking is conducted to track the same features in different frames. On the other hand, the DNN employed to segment the image to further find the dynamic objects. Then the uncertainty of DFP is modeled by considering the quality of feature tracking and the object detection result. The reprojection factor is obtained based on the feature tracking process and estimated uncertainty of visual measurements. Finally, the optimal system state is estimated using factor graph optimization (FGO) based on the IMU factor and the reprojection factor.

The major contributions of this paper are listed as follows:

(1) Instead of directly removing the DFP, this paper proposed a novel weighting scheme to mitigate the effects of the DFP in VINS.

(2) This paper evaluates the performance of the proposed Visual/Initial integration in typical urban canyons of Hong Kong

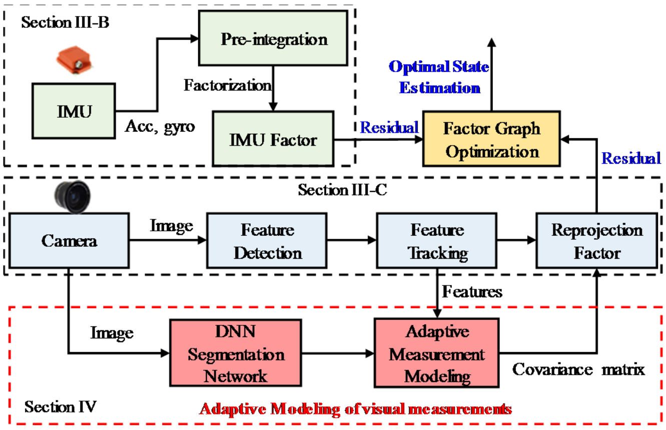


Fig. 2. Framework of the proposed VINS aided by adaptive modeling of visual measurements.

III. Visual/Inertial Integration Via Factor Graph Optimization

This section firstly presents the system definition. Then the IMU pre-integration factor and camera factor are derived subsequently. Finally, the optimization is described.

A. System Definition

The objective of factor graph optimization is to minimize the residuals derived from multiple sensor measurements [32]. In this paper, the residuals include the one from the IMU measurements and the one from visual measurements. The state vector considered in this paper is defined as follows,

$$\chi = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, \mathbf{x}_c^b, \lambda_1, \lambda_2, \dots \lambda_M] \quad (1)$$

$$\mathbf{x}_k = [\mathbf{P}_{b_k}^w, \mathbf{V}_{b_k}^w, \mathbf{q}_{b_k}^w, \mathbf{b}_{a,k}, \mathbf{b}_{g,k}], k\epsilon[1, n] \quad (2)$$

$$\mathbf{x}_c^b = [\mathbf{P}_c^b, \mathbf{q}_c^b] \quad (3)$$

where the superscript w denotes the world frame. b_k is the body frame (same as IMU frame) while taking the kth image. $\mathbf{x}_k$ is the IMU state at the kth image. It contains the position ($\mathbf{P}_{b_k}^w$), velocity ($\mathbf{V}_{b_k}^w$), and orientation that is represented by quaternion ($\mathbf{q}_{b_k}^w$) in the world frame, and acceleration bias ($\mathbf{b}_{a,k}$) and gyroscope bias ($\mathbf{b}_{g,k}$) in the IMU body frame. n is the total number of keyframes considered for optimization and M is the total number of features considered. λ_l is the inverse depth of the lth feature observed for the first time, $l\in(1, M)$. $\mathbf{x}_c^b$ is the extrinsic parameter that transforms the camera frame into the IMU frame. To guarantee the computation efficiency, we only make use of the measurements inside a sliding window (which can be seen in Figure 3) to estimate the states. The images inside the sliding window are between the frame b_k and b_{k+n}, with the time of t_k and t_{k+n}, respectively. Regarding the implementation of the VINS, we make reference to the framework in [3].

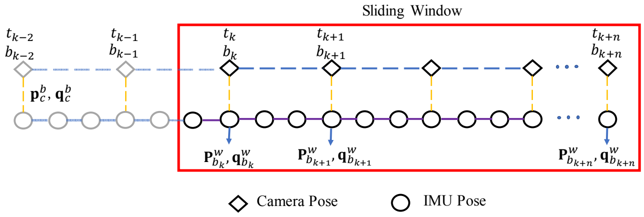

Sliding Window
t_{k-2}
b_{k-2}
t_{k-1}
b_{k-1}
t_k
b_k
t_{k+1}
b_{k+1}
t_{k+n}
b_{k+n}
$\mathbf{p}_c^b, \mathbf{q}_c^b$
$\mathbf{P}_{b_k}^w, \mathbf{q}_{b_k}^w$
$\mathbf{P}_{b_{k+1}}^w, \mathbf{q}_{b_{k+1}}^w$
$\mathbf{P}_{b_{k+n}}^w, \mathbf{q}_{b_{k+n}}^w$
◇ Camera Pose
○ IMU Pose

Fig. 3. Illustration of the sliding window used in the proposed system.

B. IMU Pre-integration Factor

This section presents the IMU measurement modeling. IMU measurements are given in the body frame, which is affected by the additive noise and bias of acceleration and gyroscope. The raw accelerometer and gyroscope measurements are given in the body frame at a given time t, respectively by,

$$\hat{\mathbf{a}}_t = \mathbf{a}_t + \mathbf{R}_w^t \mathbf{g}^w + \mathbf{b}_{a_t} + \mathbf{n}_a \quad (4)$$

$$\hat{\boldsymbol{\omega}}_t = \boldsymbol{\omega}_t + \mathbf{b}_{\omega_t} + \mathbf{n}_\omega \quad (5)$$

where $\hat{\mathbf{a}}_t$, $\hat{\boldsymbol{\omega}}_t$ are the raw measurements of IMU, $\mathbf{a}_t$, $\boldsymbol{\omega}_t$ are the expected measurements of acceleration and angular velocity of IMU. $\mathbf{g}^w = [0 \quad 0 \quad g]^T$ is the gravity vector in the world frame. $\mathbf{R}_w^t$ denotes the rotation matrix that transforms the world frame into the body frame at time t. $\mathbf{b}_{a_t}$, $\mathbf{b}_{\omega_t}$ are the acceleration bias and gyroscope bias. $\mathbf{n}_a$, $\mathbf{n}_\omega$ are the additive noise, which is assumed that is Gaussian white noise, $\mathbf{n}_a \sim \mathcal{N}(0, \sigma_a^2)$, $\mathbf{n}_\omega \sim \mathcal{N}(0, \sigma_\omega^2)$. The values of the $\mathbf{n}_a$ and $\mathbf{n}_\omega$ are determined based on the specification of IMU.

The IMU measurements can be employed to constrain the motion between two epochs using the standard IMU mechanism [33], which can work efficiently in the filtering-based sensor fusion, such as the extended Kalman filter (EKF) [34]. However, the standard IMU mechanism [33] can cause a high computation load in sensor fusion using FGO [35], due to the high frequency of IMU measurement. We employ the state-of-the-art IMU pre-integration technique [36, 37] to integrate the IMU measurements, which can effectively alleviate the high computation load in FGO and the accuracy is guaranteed, by integrating multiple IMU measurements into a single factor in FGO. There are several inertial measurements in time interval $t \in [t_k, t_{k+1}]$ between two consecutive frames b_k and b_{k+1}. Given the bias estimation, the IMU pre-integration is integrated in the b_k frame as follows [38],

$$\boldsymbol{\alpha}_{b_{k+1}}^{b_k} = \iint_{t\in[t_k, t_{k+1}]} \mathbf{R}_t^{b_k} (\hat{\mathbf{a}}_t - \mathbf{b}_{a_t}) dt^2 \quad (6)$$

$$\boldsymbol{\beta}_{b_{k+1}}^{b_k} = \int_{t\in[t_k, t_{k+1}]} \mathbf{R}_t^{b_k} (\hat{\mathbf{a}}_t - \mathbf{b}_{a_t}) dt \quad (7)$$

$$\boldsymbol{\gamma}_{b_{k+1}}^{b_k} = \int_{t\in[t_k, t_{k+1}]} \frac{1}{2} \Omega(\hat{\boldsymbol{\omega}}_t - \mathbf{b}_{\omega_t}) \boldsymbol{\gamma}_t^{b_k} dt \quad (8)$$

$$\Omega(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y & \omega_x \\ \omega_z & 0 & -\omega_x & \omega_y \\ -\omega_y & \omega_x & 0 & \omega_z \\ \omega_x & \omega_y & \omega_z & 0 \end{bmatrix} \quad (9)$$

where $\boldsymbol{\alpha}_{b_{k+1}}^{b_k}$, $\boldsymbol{\beta}_{b_{k+1}}^{b_k}$, and $\boldsymbol{\gamma}_{b_{k+1}}^{b_k}$ are the pre-integration terms between frames b_k and b_{k+1}, which represent the changes of position, velocity, and orientation, respectively. $\mathbf{R}_t^{b_k}$ is the rotation matrix that transforms the body frame at time t into the reference frame b_k. In fact, this is one of the major difference compared with the standard IMU mechanism, as the pre-integration is performed in the body frame b_k and the standard IMU mechanism is conducted with respect to the world frame. $\boldsymbol{\gamma}_t^{b_k}$ is a quaternion that transforms the body frame at time t into the reference frame b_k. The ω_x, ω_y and ω_z denote the angular velocities in the body frame.

The IMU pre-integration between the two consecutive frames takes the b_k as the reference frame. Based on the information, the position, velocity, and orientation in the world frame can be derived as follows,

$$\mathbf{P}_{b_{k+1}}^w = \left(\mathbf{P}_{b_k}^w + \mathbf{V}_{b_k}^w \Delta t_k - \frac{1}{2}\mathbf{g}^w \Delta t_k^2\right) + \mathbf{R}_{b_k}^w \boldsymbol{\alpha}_{b_{k+1}}^{b_k} \quad (10)$$

$$\mathbf{V}_{b_{k+1}}^w = \left(\mathbf{V}_{b_k}^w - \mathbf{g}^w \Delta t_k\right) + \mathbf{R}_{b_k}^w \boldsymbol{\beta}_{b_{k+1}}^{b_k} \quad (11)$$

$$\boldsymbol{\gamma}_{b_{k+1}}^{b_k} = \mathbf{q}_w^{b_k} \otimes \mathbf{q}_{b_{k+1}}^w \quad (12)$$

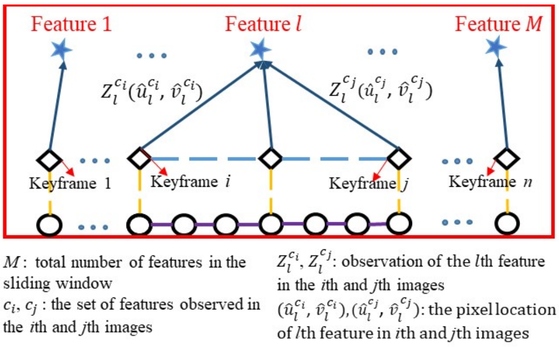
According to the two known states of b_k and b_{k+1}, the residual for IMU pre-integration measurement in the two consecutive frames b_k and b_{k+1} can be defined as follows [38],

$$\mathrm{r}_{\mathcal{B}}\left(\hat{\mathbf{Z}}_{b_{k+1}}^{b_k}, \chi\right) = \begin{bmatrix} \delta\boldsymbol{\alpha}_{b_{k+1}}^{b_k} \\ \delta\boldsymbol{\beta}_{b_{k+1}}^{b_k} \\ \delta\boldsymbol{\theta}_{b_{k+1}}^{b_k} \\ \delta\mathbf{b}_a \\ \delta\mathbf{b}_\omega \end{bmatrix} = \begin{bmatrix} \mathbf{R}_w^{b_k}\left(\mathbf{P}_{b_{k+1}}^w - \mathbf{P}_{b_k}^w + \frac{1}{2}\mathbf{g}^w \Delta t_k^2 - \mathbf{V}_{b_k}^w \Delta t_k\right) - \boldsymbol{\alpha}_{b_{k+1}}^{b_k} \\ \mathbf{R}_w^{b_k}\left(V_{b_{k+1}}^w + \mathbf{g}^w \Delta t_k - \mathbf{V}_{b_k}^w\right) - \boldsymbol{\beta}_{b_{k+1}}^{b_k} \\ 2\left[\mathbf{q}_{b_k}^{w^{-1}} \otimes \mathbf{q}_{b_{k+1}}^w \otimes \left(\boldsymbol{\gamma}_{b_{k+1}}^{b_k}\right)^{-1}\right]_{xyz} \\ \mathbf{b}_{a,b_{k+1}} - \mathbf{b}_{a,b_k} \\ \mathbf{b}_{\omega,b_{k+1}} - \mathbf{b}_{\omega,b_k} \end{bmatrix} \quad (13)$$

where $\hat{\mathbf{Z}}_{b_{k+1}}^{b_k}$ represents the observation measurements of IMU between frames b_k and b_{k+1}. The operator $[.]_{xyz}$ is used for extracting the vector part of a quaternion $\mathbf{q}$ for the orientation difference. $\otimes$ means multiplication operation between two quaternions. $\Delta\boldsymbol{\theta}_{b_{k+1}}^{b_k}$ represents the orientation constraint. The $\delta\boldsymbol{\alpha}_{b_{k+1}}^{b_k}$ represents the derived position constraint. The $\delta\boldsymbol{\beta}_{b_{k+1}}^{b_k}$ denotes the velocity constraint. The $\delta\mathbf{b}_a$ and $\delta\mathbf{b}_\omega$ denote the accelerometer and gyroscope biases constraints, respectively. The $\left[\boldsymbol{\alpha}_{b_{k+1}}^{b_k}, \boldsymbol{\beta}_{b_{k+1}}^{b_k}, \boldsymbol{\gamma}_{b_{k+1}}^{b_k}\right]$ represents pre-integrated measurements. When the estimation of bias changes, the IMU measurements will be repropagated under the new bias estimation.

C. Camera Factor

This section presents the visual measurement modeling. The direct raw measurement from the camera is the raw image at a given epoch t. Similar to the work in [38], we formulate the visual measurement residual based on a reprojection error. For a given new image, the features are detected by the Shi-Tomasi [39] corner detection algorithm. Meanwhile, the Kanade-Lucas-Tomasi (KLT) sparse optical flow algorithm [40] is employed to track the features. The derivation of the reprojection error relies heavily on the quality of feature tracking. To guarantee that enough features are detected in a frame of the image, new corner features are also detected. During the feature tracking, only certain images, the keyframes, are employed to perform the feature tracking to enforce the efficiency. The keyframes are chosen based on two criteria: 1) The first one is the average parallax criteria: if the average parallax of the tracked features between the current frame and the latest keyframe override a certain threshold, then the current frame is treated as a new keyframe. 2) if the number of the tracked features inside the current image is lower than a certain threshold, then this frame is regarded as a new keyframe. Figure 4 denotes the feature tracking processing where n is the total number of keyframes in the sliding window. The lth feature is first observed in the ith image. The $\mathrm{Z}_l^{c_i}(\hat{u}_l^{c_i}, \hat{v}_l^{c_i})$ represents the first observation of lth feature in ith image. The $\mathrm{Z}_l^{c_j}(\hat{u}_l^{c_j}, \hat{v}_l^{c_j})$ denotes the observation of the same feature in jth image. We can see from Fig. 4 that the feature is tracked for several times.


Feature 1
Feature l
Feature M
$Z_l^{c_i}(\hat{u}_l^{c_i}, \hat{v}_l^{c_i})$
$Z_l^{c_j}(\hat{u}_l^{c_j}, \hat{v}_l^{c_j})$
Keyframe 1
Keyframe i
Keyframe j
Keyframe n

M: total number of features in the sliding window

c_i, c_j : the set of features observed in the ith and jth images

$Z_l^{c_i}$, $Z_l^{c_j}$: observation of the lth feature in the ith and jth images

$(\hat{u}_l^{c_i}, \hat{v}_l^{c_i})$, $(\hat{u}_l^{c_j}, \hat{v}_l^{c_j})$: the pixel location of lth feature in ith and jth images

Fig. 4. Illustration of the relationship between keyframe and feature of the camera images used for the visual odometry.

The conventional reprojection residual is defined in the image plane, but this model is not suitable for most of the camera models. In this paper, we follow the work [38], the residual is defined on an unit sphere, which is applicable for almost all of the camera models. The unit vector for observation of the feature l which is projected into the unit sphere is as follows:

$$\hat{\bar{p}}_l^{c_j} = [\mathrm{I}_1 \quad \mathrm{I}_2]^T \cdot \pi_c^{-1}\left(\begin{bmatrix}\hat{u}_l^{c_j} \\ \hat{v}_l^{c_j}\end{bmatrix}\right) \tag{14}$$

where $[\mathrm{I}_1 \quad \mathrm{I}_2]^T$ are two arbitrarily selected orthogonal bases on the tangent plane corresponding to the feature observation. The π_c^{-1} is the back-projection function, which turns a pixel location into a unit vector using camera intrinsic parameters. To formulate the residual corresponding to the measurement $\hat{\bar{p}}_l^{c_j}$, the expected observation $p_l^{c_j}$ is needed. The direct method is to derive the $p_l^{c_j}$ based on the current state χ. To make the best use of the feature tracking process which provide continuous geometry constraints, we derive the $p_l^{c_j}$ based on the keyframe i. For the sake of clearer explanation, we divide the formulation into several steps as follows:

Step 1: get the feature l from the pixel position in image i to the body frame (IMU frame) as follows:

$$\mathrm{S}_1 = \mathbf{R}_c^b \frac{1}{\lambda_l} \pi_c^{-1}\left(\begin{bmatrix}\hat{u}_l^{c_i} \\ \hat{v}_l^{c_i}\end{bmatrix}\right) + \mathbf{P}_c^b \tag{15}$$

The $\mathbf{R}_c^b$ and $\mathbf{P}_c^b$ represent the rotation matrix and translation matrix from the camera frame to the body frame. Then the pixel location $(\hat{u}_l^{c_i}, \hat{v}_l^{c_i})$ in the ith image is transformed into the body frame.

Step 2: get the feature l in the ith image from the body frame to the world frame, and then translated to the jth image in the world frame as follows:

$$\mathrm{S}_2 = \mathbf{R}_{b_i}^w(\mathrm{S}_1) + \mathbf{P}_{b_i}^w - \mathbf{P}_{b_j}^w \tag{16}$$

The $\mathbf{R}_{b_i}^w$ and $\mathbf{P}_{b_i}^w$ are the rotation matrix and translation matrix which transforms the lth feature detected in the ith image from the body frame to the world frame. The $\mathbf{P}_{b_j}^w$ is the translation matrix which transforms the lth feature detected in the jth image from the body frame to the world frame.

Step 3: get the feature l in the jth image from world frame to the body frame, and then transformed into the camera frame as follows:

$$\mathrm{S}_3 = \mathbf{R}_w^{b_j}(\mathrm{S}_2) - \mathbf{P}_c^b \tag{17}$$

$$p_l^{c_j} = \mathbf{R}_b^c(\mathrm{S}_3) \tag{18}$$

The $\mathbf{R}_w^{b_j}$ represents the rotation matrix which transforms the same feature in the jth image from the world frame to the body frame. The $\mathbf{P}_c^b$ is the translation matrix that transforms the camera frame to the body frame. The $p_l^{c_j}$ denotes the predicted feature measurement on the unit sphere by transforming its first observation in the ith image to jth image. The $\mathbf{R}_b^c$ is the rotation matrix that transforms the body frame to the camera frame.

Step 4: Therefore, the residual for lth feature measurement in keyframe j is defined as follows,

$$\mathrm{r}_C\left(\hat{\mathrm{Z}}_l^{c_j},\chi\right) = [\mathrm{I}_1 \quad \mathrm{I}_2]^T\cdot\left(\hat{\bar{p}}_l^{c_j} - \frac{p_l^{c_j}}{\left\|p_l^{c_j}\right\|}\right) \tag{19}$$

The $\mathrm{r}_C(*)$ represents the residual of the lth feature measurement in the jth image. $\hat{Z}_l^{c_j}$ denotes the observation measurement of lth feature in the jth image. Be noted that the degree of freedom of the feature is two dimensions, therefore the residual is projected in the tangent plane. The $\hat{\bar{p}}_l^{c_j}$ denotes the unit vector for the observation of the lth feature in the jth frame.

D. Marginalization

Each feature measurement corresponds to a factor in FGO. Therefore, the computational complexity will increase dramatically over time. The straightforward way is to remove part of old states and their associated measurements. However, this will fail to make use of historical data. In order to reduce the computational loads and maintain the accuracy, the marginalization is used. The process of marginalization is to marginalize some older visual measurements. During the system optimization, some of the unsatisfactory IMU states and features are marginalized out from the sliding window into a prior. The two strategies proposed [38] to select marginalized measurements. Firstly, if the second latest frame is a keyframe, it will be kept in the sliding window, meanwhile, the oldest frame is marginalized out with its corresponding measurements. Conversely, if the second latest frame is a non-keyframe, the visual measurements will be left out, and IMU measurements are kept that connect to this non-keyframe, which can maintain the sparsity of the system. The marginalization is carried out by the Schur complement [41]. A new prior is constructed based on all marginalized measurements related to the removed state and the residual for the prior factor can be derived accordingly.

E. Factor Graph Optimization (FGO)

Based on the derived residuals from 1) IMU pre-integration, 2) the visual measurement and 3) marginalization, we can construct a factor graph. The objective of the FGO is to minimize the sum of prior and the Mahalanobis norm of all measurement residuals to obtain a maximum posterior estimation. The cost function of the system is as follows:

$$\min_{\chi}\left\{\left\|\mathrm{r}_p - \mathrm{H}_p\chi\right\|^2 + \sum_{k\epsilon\mathcal{B}}\left\|\mathrm{r}_{\mathcal{B}}\left(\hat{Z}_{b_{k+1}}^{b_k},\chi\right)\right\|_{\mathrm{P}_{b_{k+1}}^{b_k}}^2 + \sum_{(l,j)\epsilon C}\left\|\mathrm{r}_C\left(\hat{Z}_l^{c_j},\chi\right)\right\|_{\mathrm{P}_l^{c_j}}^2\right\} \tag{20}$$

where $\{\mathrm{r}_p, \mathrm{H}_p\}$ is the prior information from the marginalization operation. $\mathrm{r}_{\mathcal{B}}(.)$ is the residual term for IMU pre-integration. $\mathrm{r}_C(.)$ is the residual term for visual re-projection. $\mathcal{B}$ is the set of all IMU measurements, C is the set of features that have been observed at least twice in the current sliding window. $\mathrm{P}_{b_{k+1}}^{b_k}$ is the information matrix for IMU pre-integration. $\mathrm{P}_l^{c_j}$ is the information matrix for visual re-projection, which represents the uncertainty of feature measurements. In [38], the $\mathrm{P}_l^{c_j}$ is fixed and is correlated with the focal length. The information matrix is the inverse of the covariance matrix. The fixed information matrix can work well in an ideal scenario. Unfortunately, the positioning result will be significantly misled by unmodeled outliers. Therefore, in the next section, we propose to adaptively estimate the uncertainty of visual measurements.

IV. Self-tuning Visual Measurement Modeling

This section presents the adaptive visual measurement modeling with the help of DFP detection using DNN.

A. Object Detection Using Deep Neural Network

To detect the dynamic objects, the YOLO is employed to segment the image [28]. The YOLO is a single neural network which results in its outperforming efficiency. Assuming that a set of tracked features at a given epoch t from the jth image are denoted by $\mathbf{F}_t^j$ as follows:

$$\mathbf{F}_t^j = \{f_{t,1}^j, f_{t,2}^j, \cdots, f_{t,m}^j\} \tag{21}$$

where m represents the number of features in the jth image. Each feature $f_{t,l}^j$ is represented by $f_{t,l}^j = \{u_{t,l}, v_{t,l}, N_{f,t,l}\}$. The $u_{t,l}$ and $v_{t,l}$ denote the pixel position of the feature in the image. The $N_{f,t,l}$ denotes the number of times that the feature l is tracked. After applying the YOLO, the feature can be classified into two sets, the static feature point SFP and DFP as $\mathbf{F}_{t,SFP}^j = \{f_{t,1,SFP}^j, f_{t,2,SFP}^j, \cdots, f_{t,m,SFP}^j\}$ and $\mathbf{F}_{t,DFP}^j = \{f_{t,1,DFP}^j, f_{t,2,DFP}^j, \cdots, f_{t,m,DFP}^j\}$, respectively. Each feature set includes several features.

B. Adaptive Visual Measurement Modeling

After identifying the category of the visual feature, we propose to give them different uncertainty. During the integration of visual and IMU, the uncertainty of visual measurement is encoded in the information matrix $\mathrm{P}_l^{c_j}$ in (20). In [3], the information matrix is correlated with the specification of the camera as follows:

$$\mathrm{P}_l^{c_j} = \begin{bmatrix} \frac{\mathrm{F_c}}{1.5} & 0 \\ 0 & \frac{\mathrm{F_c}}{1.5} \end{bmatrix} \tag{22}$$

where the $\mathrm{F_c}$ is the focal length of the camera. For an SFP, we propose to calculate the corresponding information as follows:

$$\mathrm{P}_{l,SFP}^{c_j} = \begin{bmatrix} \frac{\mathrm{F_c}}{1.5} & 0 \\ 0 & \frac{\mathrm{F_c}}{1.5} \end{bmatrix} N_{f,t,l} S_{scaling} \tag{23}$$

where the $S_{scaling}$ is an experimentally determined scaling factor. Therefore, the uncertainty for the static feature is correlated with the focal length of the camera and the feature tracking quality ($N_{f,t,l}$). For a DFP, we propose to calculate the corresponding information as follows:

$$\mathrm{P}_{l,DFP}^{c_j} = \begin{bmatrix} \frac{\mathrm{F_c}}{1.5} & 0 \\ 0 & \frac{\mathrm{F_c}}{1.5} \end{bmatrix} N_{f,t,l} S_{scaling} W_D \tag{24}$$

where the W_D is a weighting which is smaller than 1. Therefore, the uncertainty for the DFP is correlated with the focal length of the camera, the feature tracking quality ($N_{f,t,l}$) and the category it belongs to.

C. Optimization with Self-tuning Sensor Models

Based on the derived adaptive information matrix in Section IV, improved optimization can be derived as follows:

$$\min_{\chi}\left\{\left\|\mathrm{r}_p-\mathrm{H}_p\chi\right\|^2+\sum_{k\epsilon\mathcal{B}}\left\|\mathrm{r}_{\mathcal{B}}\left(\hat{Z}_{b_{k+1}}^{b_k},\chi\right)\right\|_{\mathrm{P}_{b_{k+1}}^{b_k}}^2+\sum_{(l,j)\epsilon C}\left\|\mathrm{r}_C\left(\hat{Z}_{l,SFP}^{c_j},\chi\right)\right\|_{\mathrm{P}_{l,SFP}^{c_j}}^2+\sum_{(l,j)\epsilon C}\left\|\mathrm{r}_C\left(\hat{Z}_{l,DFP}^{c_j},\chi\right)\right\|_{\mathrm{P}_{l,DFP}^{c_j}}^2\right\} \tag{25}$$

where the $\hat{Z}_{l,SFP}^{c_j}$ and $\hat{Z}_{l,DFP}^{c_j}$ denote the SFP and DFP observation measurements, respectively. In this case, both the SFP and the DFP are employed during the optimization.

V. EXPERIMENT EVALUATION

A. Experiment Setup

The proposed method is verified through real road tests in the deep urban canyon of Hong Kong. An Xsens Mti 10 IMU is employed to collect raw measurements at a frequency of 200 Hz. A monocular camera (BFLY-U3-23S6C-C) is employed to collect colored images. In addition, the NovAtel SPAN-CPT, a GNSS (GPS, GLONASS, and Beidou) RTK/INS (with fiber-optic gyroscopes) integrated navigation system, is used to provide the ground truth of positioning. The gyro bias in-run stability of the FOG is 1 degree per hour and its random walk is 0.067 degree per hour. The baseline between the rover and GNSS base station is about 7 km. All the data were collected and synchronized based on the time stamp provided by the robot operation system (ROS) [42]. The coordinate systems between all the sensors were calibrated before the experiment. The sensor setup is shown in Fig. 5. Fig. 5 (a) and (b) show the snapshots with numerous moving objects, which is challenging for VINS.

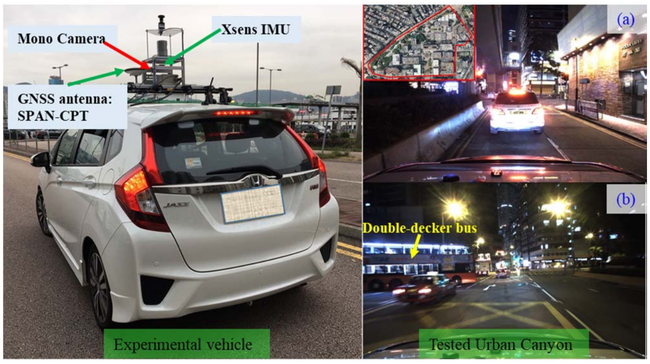


Fig. 5. Sensor setup and the examples of tested scenario.

To verify the performance of the proposed method, several methods are compared.

(1) **VINS** [3]: positioning from VINS.

(2) **VINS-R**: positioning from VINS with DFP removal.

(3) **VINS-M**: positioning from VINS with DFP remodeling.

The performance of the VINS is evaluated by the relative positioning error (RPE) using the EVO toolkit [43], which is extensively used in the SLAM research community. The parameters used during the experiment are shown in Table I.

TABLE I. PARAMETER VALUES USED IN THIS PAPER

Window Size (n)	10
$\mathrm{F_c}$	400
W_D	0.5
$S_{scaling}$	0.02

B. Performance Evaluation of the Proposed Method in the Urban Canyon

The positioning results for the listed three methods are shown in Table II. The mean error of VINS is 0.79 meters with the maximum error reaching 5.58 meters, due to the challenging dynamic environmental conditions. The mean error of VINS-R decreases from 0.79 meters to 0.68 meters with the maximum error reducing to 4.45 meters. The improvement is mainly caused by the removal of the detected DFP. The results show that the object removal can slightly mitigate the effects of dynamic objects in the tested scenario. However, the improvement is still limited. The mean error decreases to 0.52 meters with the help of the proposed DFP remodeling method with the maximum error reducing to 4.21 meters, in which the positioning accuracy is improved by 34.18%.

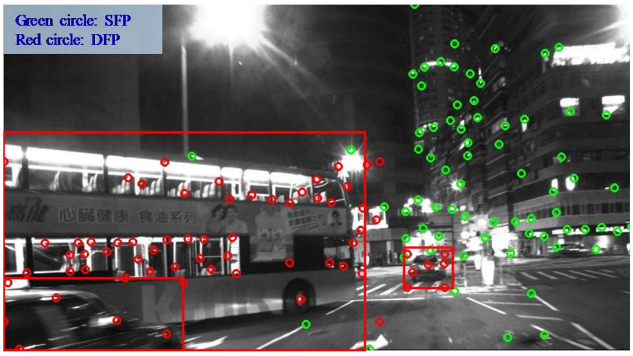


Fig. 6. Example of a case (Epoch 163) with numerous dynamic objects making up the majority of the image.

TABLE II. POSITIONING PERFORMANCE COMPARISON BETWEEN THE LISTED THREE METHODS

All data	VINS	VINS-R	VINS-M
Mean error	0.79 m	0.68 m	0.52 m
Std	0.96m	0.79 m	0.67 m
Max error	5.58 m	4.45 m	4.21 m
Improvement		13.92%	34.18%

TABLE III. POSITIONING PERFORMANCE OF THE LISTED THREE METHODS AT EPOCH 163

Epoch 163	VINS	VINS-R	VINS-M

RPE	0.45 m	0.42 m	0.14 m
Improvement		6.67%	68.89%

Interestingly, Fig. 6 shows a case where the features belonging to dynamic vehicles make up the left part of the image. After removing all the detected DFP, only the features lies on the right side of Fig. 6 are employed to estimate the state of ego-vehicle. The positioning results are shown in Table III. The performance of VINS-R (0.42 meters) is almost the same as the VINS (0.45 meters). However, the proposed method, the VINS-M obtains significantly better performance (0.14 meters). This is mainly caused by the distortion of the geometry of feature distribution. Therefore, excessive dynamic feature removal is not preferable in this kind of case. In fact, the double-decker bus shown in Fig. 6 is quite common in the urban canyons of Hong Kong. Instead of directly removing all the DFP, this paper makes use of both the static and dynamic feature in VINS. The improvement shown in Table III shows the feasibility of this proposed argument.

The trajectories of the listed three methods and the reference are shown in Fig 7. Overall, the trajectory from VINS-M (blue curve) is the one closest to the reference trajectory (black curve). The relative positioning error throughout the test is shown in Fig. 8. Although the accuracy of the proposed method is improved with the help of DFP remodeling, the maximum error still can reach more than 4 meters. The remaining error is mainly caused by other factors, such as the unstable illumination conditions, the failure feature tracking. In other words, this paper only contributes to mitigate the effects of dynamic objects. Effectively modeling the potential error caused by unstable environment conditions is still an open question.

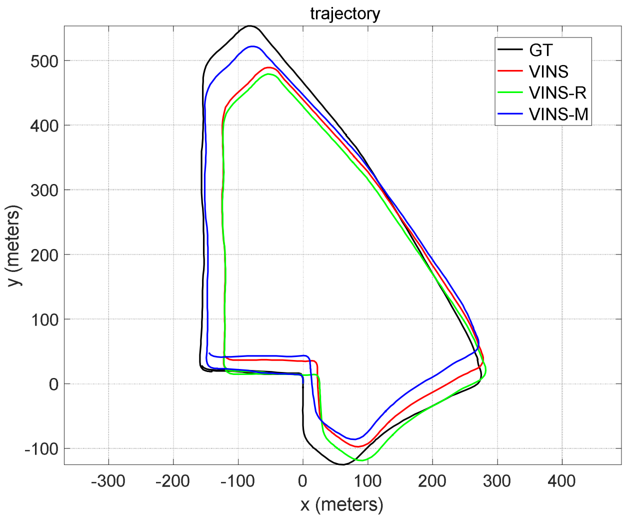


Fig. 7. The trajectories of the VINS, VINS-R, VINS-M and reference in the tested scenarios. The black curve denotes the reference from SPAN-CPT. The red, green and blue represent the VINS, VINS-R and VINS-M, respectively.

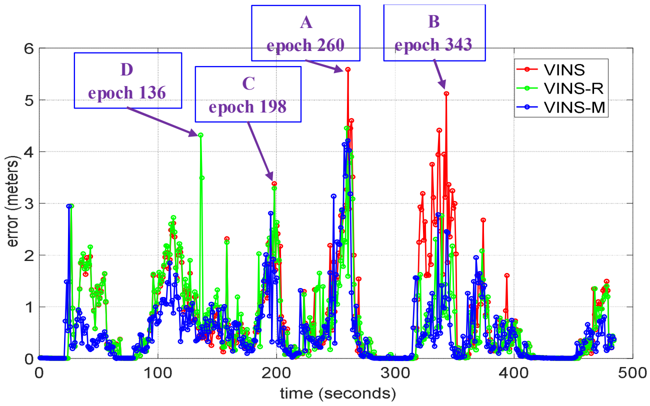


Fig. 8. Relative positioning errors of the tested three methods. The red curves denote the VINS. The green and blue curves denote the VINS-R and VINS-M, respectively.

C. Discussions

To show the detail of the improvement caused by the proposed method, four selected snapshots are shown in Fig. 9. The corresponding positioning errors are also shown in Fig. 8.

We can see from Fig. 8 that the error of VINS reaches the maximum value 5.58 meters at the epoch 260 (A). The error of VINS-R decreases to 4.00 meters, which shows that the DFP removal can slightly mitigate the effects of dynamic objects. The error of VINS-M decreases to 3.18 meters, which shows that DFP remodeling can acquire the better improved result than VINS-R. Similar condition appears on the epoch 343 (B) and epoch 198 (C). Compared the VINS-R, the proposed method can outperform in positioning accuracy.

Interestingly, we found that the VINS-R leads to larger positioning error at epoch 136 (D), which can reach 4.32 meters. The error of VINS and VINS-M are 0.42 meters and 0.53 meters, respectively. In other words, our proposed method still has shortcoming in the challenging environment. We will further study how to improve the positioning accuracy in the future.

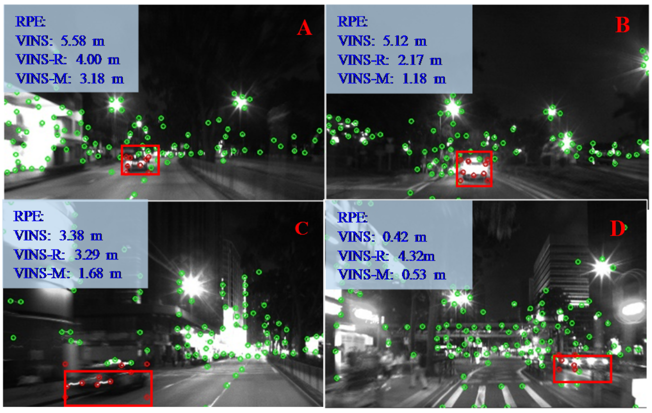


Fig. 9. The images of the tested scenarios in the four selected epochs with respect to Fig. 8. The red box represents the dynamic object.

VI. Conclusions and Future Work

Remodeling the DFP is significant for positioning in VINS, especially in the challenging scenario with numerous dynamic objects, which can degrade the feature quality. In this paper, instead of directly removing the DFP, we propose to remodel the DFP by giving the DFP smaller weighting than the SVF and the improved result is obtained. Compared with the proposed method VINS-M, the VINS-R is not recommended for the VINS positioning, since the excessive DFP removal can distort the geometry distribution of features in the environment with numerous moving objects. However, the proposed method still has limitations in that the weighting is fixed during the experiment. To enhance the robustness of VINS-M, we will further study how to acquire a better quality of features in future work.

ACKNOWLEDGMENT

The authors acknowledge the support of the Hong Kong PolyU internal grant on the project ZVKZ, “Navigation for Autonomous Driving Vehicle using Sensor Integration”.

REFERENCES

[1] S. Gupte, P. I. T. Mohandas, and J. M. Conrad, "A survey of quadrotor unmanned aerial vehicles," in *2012 Proceedings of IEEE Southeastcon*, 2012, pp. 1-6: IEEE.

[2] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354-3361: IEEE.

[3] T. Qin, P. Li, and S. J. I. T. o. R. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," vol. 34, no. 4, pp. 1004-1020, 2018.

[4] R. Mur-Artal, J. M. M. Montiel, and J. D. J. I. t. o. r. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," vol. 31, no. 5, pp. 1147-1163, 2015.

[5] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. J. T. I. J. o. R. R. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," vol. 34, no. 3, pp. 314-334, 2015.

[6] M. Li and A. I. J. T. I. J. o. R. R. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," vol. 32, no. 6, pp. 690-711, 2013.

[7] W. Wen, L.-T. Hsu, and G. Zhang, "Performance analysis of NDT-based graph SLAM for autonomous vehicle in diverse typical driving scenarios of Hong Kong," *Sensors,* vol. 18, no. 11, p. 3928, 2018.

[8] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Robotics: Science and Systems*, 2014, vol. 2, p. 9.

[9] X. Bai, W. Wen, and L.-T. Hsu, "Performance Analysis of Visual/Inertial Integrated Positioning in Typical Urban Scenarios of Hong Kong," in *Proceedings of 2019 Asian-Pacific Conference on Aerospace Technology and Science* Taiwan, 2019.

[10] W. Wen *et al.*, "UrbanLoco: A Full Sensor Suite Dataset for Mapping and Localization in Urban Scenes," 2019.

[11] G. J. a. p. a. Huang, "Visual-inertial navigation: A concise review," 2019.

[12] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual SLAM and structure from motion in dynamic environments: A survey," *ACM Computing Surveys (CSUR),* vol. 51, no. 2, p. 37, 2018.

[13] C. Yu *et al.*, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1168-1174: IEEE.

[14] Y. Sun, M. Liu, and M. Q.-H. Meng, "Motion removal for reliable RGB-D SLAM in dynamic environments," *Robotics and Autonomous Systems,* vol. 108, pp. 115-128, 2018.

[15] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust monocular SLAM in dynamic environments," in *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013, pp. 209-218: IEEE.

[16] V. Mohanty, S. Agrawal, S. Datta, A. Ghosh, V. D. Sharma, and D. Chakravarty, "Deepvo: A deep learning approach for monocular visual odometry," *arXiv preprint arXiv:1611.06069,* 2016.

[17] F. Zhong, S. Wang, Z. Zhang, and Y. Wang, "Detect-SLAM: Making object detection and slam mutually beneficial," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1001-1010: IEEE.

[18] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving RGB-D SLAM in dynamic environments: A motion removal approach," *Robotics and Autonomous Systems,* vol. 89, pp. 110-122, 2017.

[19] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *2010 ieee intelligent vehicles symposium*, 2010, pp. 486-492: IEEE.

[20] S. Wangsiripitak and D. W. Murray, "Avoiding moving outliers in visual SLAM by tracking moving objects," in *2009 IEEE international conference on robotics and automation*, 2009, pp. 375-380: IEEE.

[21] X. Bai, W. Wen, and L.-T. Hsu, "Performance Analysis of Visual/Inertial Integrated Positioning in Diverse Typical Urban Scenarios of Hong Kong," 2019.

[22] W. Wen, G. Zhang, and L.-T. Hsu, "Correcting GNSS NLOS by 3D LiDAR and Building Height," presented at the ION GNSS+, 2018, Miami, Florida, USA., 2018.

[23] W. Wen, G. Zhang, and L.-T. Hsu, "GNSS NLOS Exclusion Based on Dynamic Object Detection Using LiDAR Point Cloud," *IEEE Transactions on Intelligent Transportation Systems,* 2019.

[24] W. Wen, G. Zhang, and L. T. Hsu, "Correcting NLOS by 3D LiDAR and building height to improve GNSS single point positioning," *Navigation,* vol. 66, no. 4, pp. 705-718, 2019.

[25] X. Bai, W. Wen, G. Zhang, and L.-T. Hsu, "Real-time GNSS NLOS Detection and Correction Aided by Sky-Pointing Camera and 3D LiDAR," presented at the Proceedings of ION Pacific PNT 2019, Honolulu, HA, USA, 2019.

[26] W. Wen, G. Zhang, and L.-T. Hsu, "Object Detection Aided GNSS and Its Integration with LiDAR in Highly Urbanized Areas,," *IEEE Intelligent Transportation Systems Magazine (accepted),* p. Accepted, 2019.

[27] X. B. Weisong Wen, Yin Chiu Kan, and Li-Ta Hsu, "Tightly Coupled GNSS/INS Integration Via Factor Graph and Aided by Fish-eye Camera (Accepted)," *Ieee Transactions on Vehicular Technology,* 2019.

[28] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.

[29] A. Antonini, "Pre-integrated dynamics factors and a dynamical agile visual-inertial dataset for UAV perception," Massachusetts Institute of Technology, 2018.

[30] S. Agarwal and K. Mierle, "Ceres solver: Tutorial & reference," *Google Inc,* vol. 2, p. 72, 2012.

[31] C. Forster, L. Carlone, F. Dellaert, and D. J. I. T. o. R. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual--Inertial Odometry," vol. 33, no. 1, pp. 1-21, 2016.

[32] F. Dellaert, M. J. F. Kaess, and T. i. Robotics, "Factor graphs for robot perception," vol. 6, no. 1-2, pp. 1-139, 2017.

[33] P. D. Groves, *Principles of GNSS, inertial, and multisensor integrated navigation systems*. Artech house, 2013.

[34] S. J. A. M. Thrun, "Probabilistic algorithms in robotics," vol. 21, no. 4, pp. 93-93, 2000.

[35] W. Wen, X. Bai, Y.-C. Kan, and L.-T. Hsu, "Tightly Coupled GNSS/INS Integration Via Factor Graph and Aided by Fish-eye Camera," *IEEE Transactions on Vehicular Technology,* 2019.

[36] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual--Inertial Odometry," *IEEE Transactions on Robotics,* vol. 33, no. 1, pp. 1-21, 2016.

[37] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "Supplementary material to: IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," Georgia Institute of Technology2015.

[38] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics,* vol. 34, no. 4, pp. 1004-1020, 2018.

[39] J. Shi, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, 1994, pp. 593-600: IEEE.

[40] T. Senst, V. Eiselein, and T. Sikora, "II-LK–a real-time implementation for sparse optical flow," in *International Conference Image Analysis and Recognition*, 2010, pp. 240-249: Springer.

[41] F. Zhang, *The Schur complement and its applications*. Springer Science & Business Media, 2006.

[42] M. Quigley *et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, 2009, vol. 3, no. 3.2, p. 5: Kobe, Japan.

[43] M. Grupp, "evo: Python package for the evaluation of odometry and slam," ed, 2017.