A New Framework of Moving Object Tracking based on Object Detection-Tracking with Removal of Moving Features

Ly Quoc Ngoc[1], Nguyen Thanh Tin[2], Le Bao Tuan[3]
Department of Computer Vision and Cognitive Cybernetics
VNUHCM–University of Science, Ho Chi Minh city, Vietnam

***Abstract*—Object Tracking (OT) on a Moving Camera so-called Moving Object Tracking (MOT) is extremely vital in Computer Vision. While other conventional tracking methods based on fixed camera can only track the objects in its range, a moving camera can tackle this issue by following the objects. Moreover, single tracker is used widely to track object but it is not effective due to the moving camera because the challenges such as sudden movements, blurring, pose variation. The paper proposes a method inherited by tracking by detection approach. It integrates a single tracker with object detection method. The proposed tracking system can track object efficiency and effectively because object detection method can be used to find the tracked object again if the single tracker loses track. Three main contributions are presented in the paper as follow. First, the proposed Unified Visual based-MOT system can do the tasks such as Localization, 3D Environment Reconstruction and Tracking based on Stereo Camera and Inertial Measurement Unit (IMU). Second, it takes into account camera motion and the moving objects to improve the precision rate in localization and tracking. Third, proposed tracking system based on integration of single tracker as Deep Particle Filter and Object Detection as Yolov3. The overall system is tested on the dataset KITTI 2012, and it has achieved a good accuracy rate in real time.**

***Keywords*—Moving object tracking; object detection; camera localization; 3D environment reconstruction; tracking by detection**

I. Introduction

In Object Tracking, it is necessary to predict the position of object being tracked in the current frame and match them with the previous ones to achieve its precise position in the current frame. Many significant works have dealt with appearance changes overtime such as color histogram [1], HoG feature [2], SIFT or SURF feature [3], or texture features like LBP [4].The single tracker based on the popular filters such as Correlation Filter, Kalman Filter or Particle Filter. Correlation filter [5] [6] [7] is also used and it acquired high speed and accuracy. Other filters such as Kalman Filter or Particle Filter are used because they could predict the position of objects and then match the predicted position with the previous one. Kalman Filter [8]-[10] could not deal with non-linearity in the measurements because the filter tries to linearize it using approximation method. Particle filter [11], [12] are used as it could solve the drawback of Kalman filter. Recently, deep neural networks have been applied in tracking problems. S. Chen and W. Liang [13] used a CNN to distinguish the background from objects and then track the objects according to their position. CNN is integrated with correlation filter [14] or with particle filter [15], [16]. But these approaches do not take into account the challenges of moving camera. J. S. Lim and W. H. Kim [17], Y. Chen et al. [18] tried to calculate translation vector between two consecutive frames (or two frames from stereo camera).

Based on data acquired by IMU and stereo camera, the paper proposed a solution by integration of a single tracker as Deep Particle Filter and an object detection method as YOLOv3 [19], however, the object would be tracked by its three-dimensional center. In traditional object tracking from static camera, two-dimensional position of tracked object is enough but in MOT, three-dimensional position of tracked object must be considered. The challenges must be taken into account as the vibration of the camera and the movement of the object. YOLOv3 is the right solution because it can detect objects very quickly and then this result can be used to support the single tracker be more robust, and the most important thing is that it is suitable for real-time applications. In addition, in the localization and three-dimensional environment reconstruction, the removal of moving objects is considered to increase accuracy rate. To do that, the paper does not rely on estimating 6 degrees of freedom to find out the robot position, but inspired from [20], the paper splits it into two separate transformations including a rotation transformation and a translation transformation. Rotation transformation is calculated based on IMU and the translation transformation is estimated from the stereo camera. Robot can locate by itself based on these two transformations in real environments. Data which is observed from stereo camera-based environments includes two kinds of object: moving objects and static objects. If the feature points of moving objects are used to estimate the robot position and 3D point cloud of environment, the estimated error will increase over time. Therefore, the paper considers to eliminate feature points of moving objects to increase accuracy rate. This is an improvement of the paper to increase the accuracy rate of localization and 3D environment reconstruction. Most of the solutions be published have not yet considered the feature points of moving objects. But in experimental results of the paper, the accuracy rate with removal of moving features has yielded better results than the opposite. To remove moving objects, the paper uses the background subtraction method with camera motion compensation to detect moving objects proposed in [21], [22], the advantage is fast and accurate detection of moving objects. Meanwhile in the research [23] it is assumed that moving objects are identified as belonging to movable categories which are likely to move currently or in the time coming, such as people, dog, cat, and car. For instance, once a person is detected, no matter walking or standing, it is considered as a potentially moving object and remove the features belonging to the region in the image where the person was detected. The limitation of the proposed method in [23] is that it is impossible to distinguish moving objects or static objects.

In the MOT problem, the paper tries to use stereo camera and IMU without GPS for the following reasons: The paper would like to test the power of visual information acquired from stereo camera in estimating the position of the robot. The IMU data will provide rotation transformation of the robot motion. Stereo camera integrated with IMU can work better than GPS in many environments such as indoors, radio interference, noisy GPS and in the cases that the input is only visual information of tracked object.

In Section II, the paper reviews the previous work in visual tracking on a fixed camera as well as moving camera. Section III describes the proposed methods such as object localization, 3D environment reconstruction and tracking algorithm based on stereo camera and IMU. Section IV shows experimental results of localization and tracking. The paper discusses about the pros and cons of the proposed methods in Section V. Conclusion and future works will be presented in Section VI.

II. Related Works

A. Camera Localization

Robot localization is crucial for many high-level tasks such as object tracking, obstacle detection and avoidance, motion planning, autonomous navigation, local path planning and a waypoint follower, etc. Over the years, many researchers have been working on the problem of robot localization and made certain contributions. David Nistér et al. [24] proposed a system for real-time ego-motion estimation of a single camera or stereo camera. Bernd Kitt et al. [25] proposed another visual odometry algorithm based on RANSAC outlier rejection technique. Shaojie Shen et al. [20] used the feature points from stereo images and IMU information to estimate robot position. S.Prabu and G. Hu [12] proposed a vision based on localization algorithm which combinesthe partial depth estimation and particle filter techniques. Yanqing Liu et al. [26] present a robust stereo visual odometry using an improved RANSAC based method (PASAC) that makes the procedure of motion estimation much faster and more accurate than standard RANSAC. Yuquan Xu et al. [27] propose a novel algorithm for the problem of three-dimensional point cloud map based on localization using a stereo camera. S. Hong et al. [28] proposed the real-time autonomous navigation system using only a stereo camera and a low-cost GPS. All the aforementioned research works have provided the fundamental background knowledge to solve the localization problem in this paper. Here, the paper proposes a novel method to localize a robot using a stereo camera and IMU sensor, especially it takes into account moving objects to increase accuracy rate.

B. Moving Object Tracking

The moving camera could solve the disadvantages of fixed camera. The fixed camera can only track objects within their range, if the objects come out of field of view (FOV) of camera, it cannot monitor the objects and for realize this matter, it should be mounted on the moving framework such as robot, drone or an autonomous-driving car.

Y. Chen et al. [18] used features such as SIFT, SURF to match the features between two consecutive frames to find out the translation vector of camera and uses it to predict the position of objects in the frame. J. S. Lim and W. H. Kim [17] estimated motion by distinguishing 16x16 patches between the two frames. Each patch has a vector that is the main motion in this area and after traverse all the 16x16 patches in two consecutive frames, the vector with the highest frequency is selected as camera motion vector. These frameworks partly alleviate the effects of fast moving, rotation, vibration of the cameras.

There are also several ways to matching objects between two images. Q. Zhao et al. [1] matched objects by comparing color histograms but this is easy to fail in case there are regions which have the same colors with the objects. C. Ma et al. [14] applied CNN to extract features and compared objects by a correlation filter. R. J. Mozhdehi and H. Medeiros [15], T. Zhang et al. [16] inherited the previous framework and integrated it with Particle Filter.

The tracking part will inherit Particle filter to track objects and improve the performance in its prediction and measurement steps. Firstly, the paper will find out a translation vector by using feature matching algorithm, and then the position of the tracked object will be solved by applying deep neural network in conjunction with correlation filter.

Moreover, the paper inherits a deep CNN-based object detection algorithm named YOLOv3 [19] which is very fast and quite accurate to detect objects. By combining these methods, the tracking part has developed an algorithm called Tracking by Detection.

However, to track the object in the context of moving camera and moving object, the tracking part has to track object in 3D environment (by IMU and stereo cameras) so that the tracking system is realistic.

III. Method

The paper proposes a Unified Visual Based-MOT system can do the tasks such as Camera Localization, 3D Environment Reconstruction and Object Tracking.

A. Camera Localization

Inspired from the method of [20], the significant improvement is proposed in feature detection stage with removal of moving feature points. In addition, there are some differences between [20] and the paper. Specifically, instead using a built-in system [20] to get the camera position as ground truth, the paper used the ground truth GPS of KITTI dataset.

To locate the position of the camera, the paper estimates the camera motion at time t, consists of the translation transformation and the rotation transformation of camera coordinate system between two consecutive frames based on the stereo camera and IMU sensor. The IMU data provides the rotation matrix for rotation transformation. The feature points of the image used to estimate the translation, these features include moving and static features. In this case, moving features are noise. Therefore, the paper removes the moving feature points to reduce error rate in estimating camera position. It is a new point in improving the robot localization process. The camera location estimation steps are shown in Fig. 1.

1) Camera model, feature detection and feature tracking: Both cameras in the system are calibrated using the Camera Calibration Toolbox. Both cameras are divided into two systems and play different roles:

- Stereo Camera System (right and left cameras): They are used to estimate 3D positions of the features in the world coordinate system (WCS), initialize the local map at the start and update local maps when the local map accumulated errors large enough (see Fig. 1 and 7).
- Monocular Camera System (left camera): It is used to estimate robot locations, initialize and update local maps.

In the model, at each moment, the paper gets two images from the stereo camera (see Fig. 2). These two images are used for feature detection and reproduce the 3D positions of the features in WCS. However, feature detection and 3D position reconstruction will not be performed consecutively in pairs of successive images, but they are performed in a given cycle, which corresponds to 25 consecutive frames (depending on the device) (see Fig. 2). This means that at the beginning the feature detection is made from the two images of stereo camera and reconstructed the 3D position of the features in the world coordinate system, and after 25 consecutive frames of cycle (including frames used for feature detection), the above calculation process will be performed again. For 25 consecutive frames of cycle, the feature detection process and estimate the 3D positions are not performed, instead the features will be kept track on the successive image frames until a new cycle be done. The purpose of this solution is to reduce computational time but still retain the required accuracy rate.

In feature detection stage, the image features play an important role in locating robot positions. The SURF feature (Speeded Up Robust Features) [29] is extracted from pairs of images of the left and right cameras. FLANN matching algorithm (Fast Library for Approximate Nearest Neighbor) [30] is used for matching the features of two images. In order to remove outliers, Lowe outlier rejection method [31] is used. This outlier removal supports significant improvement the accuracy of localization.

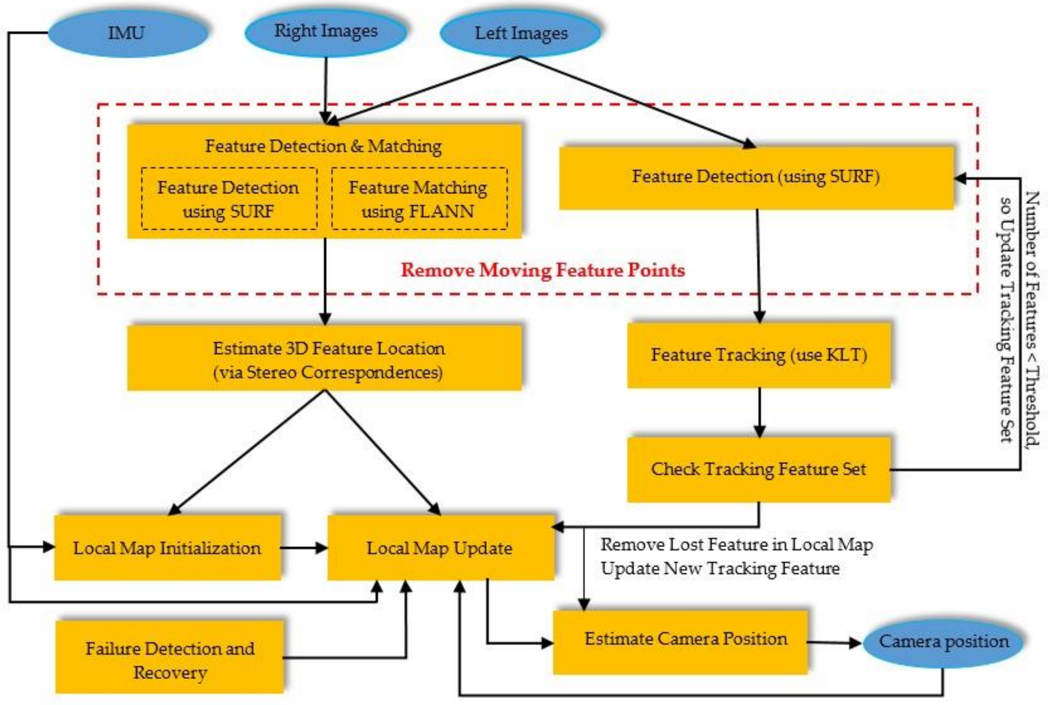
IMU
Right Images
Left Images
Feature Detection & Matching
Feature Detection using SURF
Feature Matching using FLANN
Feature Detection (using SURF)
Remove Moving Feature Points
Number of Features < Threshold, so Update Tracking Feature Set
Estimate 3D Feature Location (via Stereo Correspondences)
Feature Tracking (use KLT)
Check Tracking Feature Set
Local Map Initialization
Local Map Update
Remove Lost Feature in Local Map
Update New Tracking Feature
Failure Detection and Recovery
Estimate Camera Position
Camera position

Fig. 1. Diagram of Camera Position Estimation. Inspired from [20].
(Suppose the Robot Position is Considered as the Camera Position).

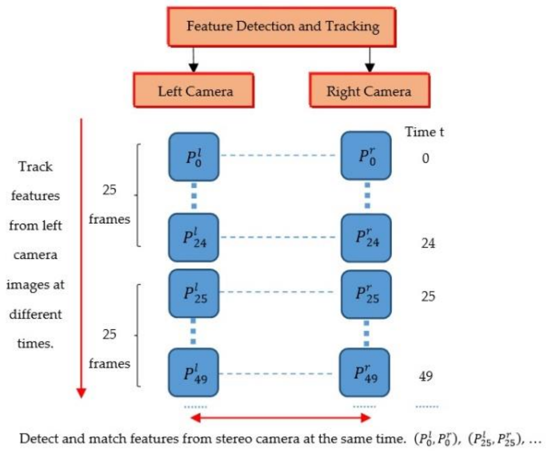
Feature Detection and Tracking
Left Camera
Right Camera
Time t
Track features from left camera images at different times.
25 frames
25 frames
0
24
25
49
Detect and match features from stereo camera at the same time. (P_0^l, P_0^r), (P_{25}^l, P_{25}^r), ...

Fig. 2. Feature Detection and Tracking Diagram of Stereo Camera at different Times. P_t^l and P_t^r are the Feature sets of Left and Right Cameras at Time t.

In the feature tracking stage, the KLT algorithm [32] is used to track features through the consecutive image frames. This feature tracking is only performed in the left camera (Monocular Camera System). Tracked features are features that are detected and estimated at the 3D position at the stereo camera system at the start of each new cycle. In the tracking process, the moving features are detected and removed in the image (see Fig. 3).

In traditional methods, to estimate camera position, the moving and static features are all used. In order to increase the accuracy rate of camera position, the paper proposes removal of moving features. Because these moving features will have a position that changes over time, so using the features to calculate the camera position, the error of the predicted position will increase over time. The process of detecting and removing moving features is performed before estimating the camera position. Here, the paper proposes using the background subtraction method for two consecutive images to detect and remove moving objects as suggested in [21]. This method will find a transformation matrix (called a homography matrix) between two consecutive images and then use this homography matrix to transform two consecutive images into the same coordinate system. Then background subtraction method for these two images is performed to find the regions of moving objects on the image. Finally, the removal of features is performed in moving regions.

Outline of the steps of the removal of moving features process (see Fig. 3). At time t, there are two consecutive images from the left camera at the time t-1 and t, are called I_{t-1} and I_t. Besides, at this time, in the feature tracking step between successive frames, two feature sets of tracking P_{t-1} and P_t are obtained, respectively, for two images I_{t-1} and I_t. Assuming that $P_{t-1} = [\mathrm{p}_{t-1}^1, \dots, \mathrm{p}_{t-1}^N]$ be the set of N key points found at time t – 1 and $P_t = [\mathrm{p}_t^1, \dots, \mathrm{p}_t^N]$ be the set of the tracked points at time t. Here, $\mathrm{p}_t^i = [x_t^i, y_t^i]$ while x_t^i and y_t^i represent its 2D position in the image. Two sets of P_{t-1} and P_t are used to find the coordinate transformation between the two images. Then convert two images I_{t-1} and I_t to the same coordinate system and perform background subtraction to find the moving regions. Finally, remove the features located in the moving regions. Steps to remove moving features at time t:

Step 1: Image registration: To find the transformation between two frames at the time t-1 and t, homography transformation is used. The relationships between these frames are shown as follow.

$$P_{t-1} = H \cdot P_t \tag{1}$$

where the transform matrix H (homography matrix) is a 3-by-3 matrix which describes the spatial relationship between two consecutive image frames.

As in [33], H can be solved by least square criteria with:

$$H = P_{t-1} P_t^T (P_t P_t^T)^{-1} \tag{2}$$

where $(\cdot)^T$ represents the matrix transpose and$(\cdot)^{-1}$represents the matrix inverse.

By multiplying the estimated transform matrix H on the pixel positions on current image, they are warped onto the image plane at the previous time instance and the same background scenes in consecutive frames can approximately overlap with each other, it performs camera motion compensation. Therefore, a new image will get at time t, $I_t^{(T)}$, this image has the same coordinate system as the image at time t-1, I_{t-1}.

Step 2: Background subtraction: Perform background subtraction between image $I_t^{(T)}$ and I_{t-1} with a certain threshold and the moving regions be detected. In addition, using Morphology operators to refine the result image to increase the accuracy of moving regions.

Step 3: Removal of moving features: The feature points of moving regions are moving features, so they are excluded (The feature points are converted to the same coordinate system with the result image).

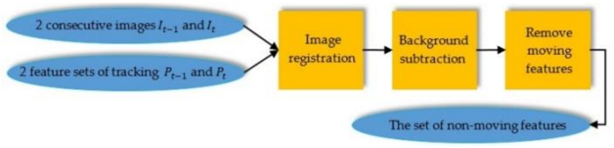
2 consecutive images I_{t-1} and I_t
2 feature sets of tracking P_{t-1} and P_t
Image registration
Background subtraction
Remove moving features
The set of non-moving features

Fig. 3. Diagram to Detect and Remove Moving Objects from Two Consecutive Images of Monocular Camera.

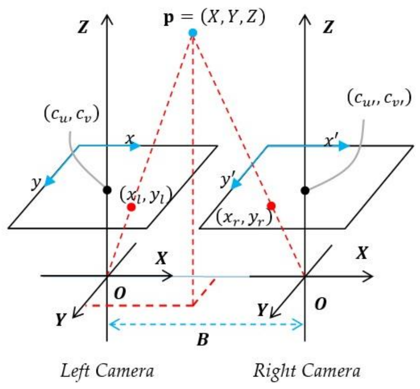

Fig. 4. 3D Environment Reconstruction from Stereo Camera Model.

2) 3D Feature location via stereo correspondences: The 3D pose of corresponding feature points is estimated by stereo correspondences [34] (see Fig. 4). The 3D camera coordinates of the feature points are obtained based on the following equations:

$$X = (x_l - c_u)\frac{B}{d} \tag{3}$$

$$Y = (y_l - c_v)\frac{B}{d} \tag{4}$$

$$Z = \frac{Bf}{d} \tag{5}$$

$$d = \sqrt{(x_l - x_r)^2 + (y_l - y_r)^2} \tag{6}$$

where f is the focal length of the stereo camera. B represents the baseline between the stereo cameras. c_u, c_v represents x and y coordinate of the principal point. d is the disparity between the feature points in the left and right images. (x_l, y_l) and $(x_r, y_r) \in \mathbb{R}^2$ are the coordinates in the left and right images of the feature point, respectively.

Thus, (X, Y, Z) is 3D camera coordinates of the feature points and Z represents the depth of the feature point.

3) Estimate camera position via 2D-3D correspondences: Inspired from the camera location estimation method presented in [20], the paper improves precision rate of localization by removal of moving features. Assume that the 3D local feature map is known. Details of local map initialization and maintenance will be presented in the following sections. The robot position is assumed that the 3D position of the left camera in the WCS. Given observations of a local map consisting of known 3D features at the time $t-1$ and the observation vector of features at the present time t, the 3D position of the camera can be estimated by minimizing the sum-of-square reprojection error of the observed features:

$$\mathbf{r}_t^* = \underset{\mathbf{r}_t}{\operatorname{argmin}} \sum_{i\in\mathfrak{I}} \left\| \frac{\mathbf{r}_t - \mathbf{p}_i}{\|\mathbf{r}_t - \mathbf{p}_i\|} \times \mathbf{k}_{it} \right\|^2 \tag{7}$$

where, as shown in Fig. 6, $\mathbf{r}_t$ is the 3D position of the camera at time t in WCS, $\mathbf{k}_{it}$ is the observation vector of the i^{th} feature point at time t in WCS (see Fig. 5), $\mathbf{g}_{it} = \frac{\mathbf{r}_t - \mathbf{p}_i}{\|\mathbf{r}_t - \mathbf{p}_i\|}$ is the unit truth vector when there is an exact position of $\mathbf{r}_t$, this vector has a direction from position $\mathbf{r}_t$ to 3D position of the $\mathbf{p}_i$ feature point, $\mathfrak{I}$ represents the set of features observed in the image at time t, $\mathbf{p}_i$ is the 3D position of the i^{th} feature in WCS.

Calculation of observation vector $\mathbf{k}_{it}$ of the i^{th} feature point at time t in WCS (see Fig. 5): The unit feature observation vectors is crucial in the problem of estimating camera position. Each feature will provide directional information from the camera position to the feature position through observation at the image plane at different times. That information is used to find camera locations in real environments.

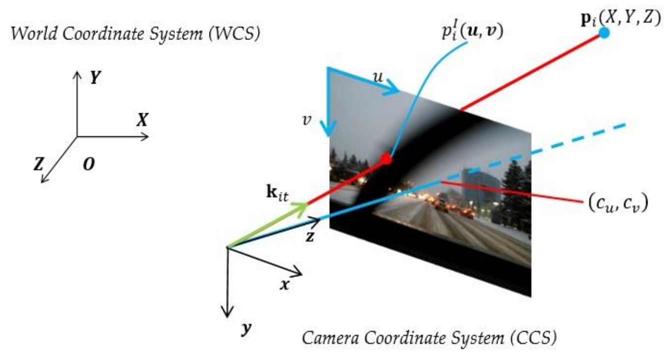
World Coordinate System (WCS)
Y
X
Z
O
$p_i^I(\boldsymbol{u},\boldsymbol{v})$
$\mathbf{p}_i(X,Y,Z)$
u
v
$\mathbf{k}_{it}$
(c_u, c_v)
z
x
y
Camera Coordinate System (CCS)

Fig. 5. The Observation of the p_iFeature Point at Time t in the Image Plane of the Left Camera. Vector k_{it} (Green Color) is Observation Vector of p_i at t in the WCS.

The observation of the i^{th} feature point from the homogeneous image coordinate system (ICS) $p_i^I(u_i, v_i, 1)$ will be transformed into an observation vector in the camera coordinate system (CCS) $\mathbf{k}_{it}^{\mathbf{C}}$ at time t and is denoted as,

$$\mathbf{k}_{it}^{\mathbf{C}} = K^{-1} p_i^I \tag{8}$$

where K^{-1} is the inverse matrix of K, this matrix will convert a p_i^I point on the image plane into a directional vector $\mathbf{k}_{it}^{\mathbf{C}}$, starting from the camera position to $\mathbf{p}_i$ (or from the feature point to $\mathbf{p}_i$), in the CCS. K is a matrix transform from the CCS to the ICS.

Then, the observation vector $\mathbf{k}_{it}^{\mathbf{C}}$ is normalized to unit vector $\frac{\mathbf{k}_{it}^{\mathbf{C}}}{\|\mathbf{k}_{it}^{\mathbf{C}}\|}$.

Transforming the $\mathbf{k}_{it}^{\mathbf{C}}$ observation vectors from the CCS to the WCS, $\mathbf{k}_{it}$ and is given by,

$$\mathbf{k}_{it} = R_I^W R_C^I \mathbf{k}_{it}^{\mathbf{C}} \tag{9}$$

where R_I^W is rotation matrix from IMU coordinate system (IMUCS) to WCS, R_C^I is rotation matrix from CCS to IMUCS, R_C^I is obtained from offline camera calibration.R_I^W is obtained from IMU data at each time t.

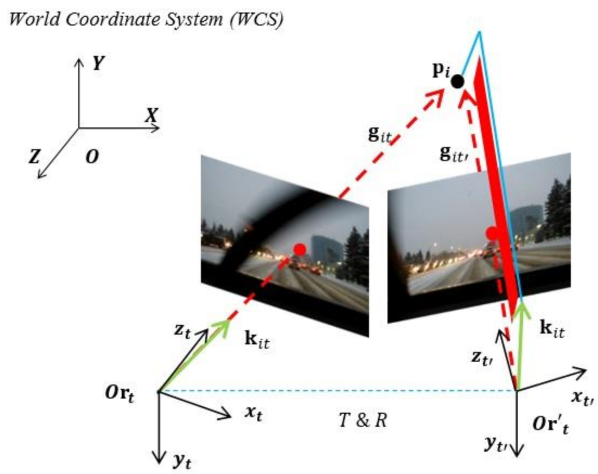

Fig. 6. Illustrate the Positions of the Left Camera at Time t and the Error between g_{it} and the Observation Vector k_{it}. The Position with the Smallest Error (Error is Total Area of Red Parallelogram) will be Robot Position at Time t. Here, the Position r_t has the Smallest Error.

Assume that the camera motion between two consecutive images is small, formula (7) can be approximated as:

$$\mathbf{r}_t^* = \underset{\mathbf{r}_t}{\operatorname{argmin}} \sum_{i\in\mathfrak{I}} \left\| \frac{\mathbf{r}_t-\mathbf{p}_i}{d_i} \times \mathbf{k}_{it} \right\|^2 \tag{10}$$

where $d_i = \|\mathbf{r}_t - \mathbf{p}_i\| \approx \|\mathbf{r}_{t-1} - \mathbf{p}_i\|$ are known quantities. By taking the derivative of formula (10) and setting it to zero, a linear system are obtained with the optimal camera position $\mathbf{r}_t$ is the unknown:

$$\left(\sum_{i\in\mathfrak{I}} \frac{\mathbb{I}_3-\mathbf{k}_{it}\mathbf{k}_{it}^T}{d_i}\right)\mathbf{r}_t = \sum_{i\in\mathfrak{I}} \frac{\mathbb{I}_3-\mathbf{k}_{it}\mathbf{k}_{it}^T}{d_i}\mathbf{p}_i \tag{11}$$

where $\mathbf{r}_t$ is the 3D position of the camera at time t in WCS, $\mathbf{k}_{it}$ is the observation vector of the i^{th} feature point at time t in WCS, $\mathbf{p}_i$ is the 3D position of the i^{th} feature in WCS, $\mathfrak{I}$ represents the set of features observed in the image at time t, $d_i = \|\mathbf{r}_{t-1} - \mathbf{p}_i\|$ is the known value, $(\cdot)^T$ represents the matrix transpose. $\mathbb{I}_3$is a 3×3 matrix unit.

Equation (11) consisted of three equations corresponding to three unknowns which are the 3D position of the camera in WCS, these three equations will not change regardless of the number of observed features. Therefore, the camera position estimation can be solved efficiently in constant time. The observed features used to calculate camera position are features that are not in moving regions. Suppose, if using the features of the moving regions, the error of the estimated camera position will increase. Since the camera position is estimated based on 3D feature points at time t-1 and the corresponding feature observation vectors at time t, if you consider a feature of moving regions, the 3D position of features in the environment will be different at the time t-1 and t in the same WCS, and the feature observation vector at time t will not match the truth vector of the 3D feature point at time t-1 (i.e. the observation vector $\mathbf{k}_{it}$ will not match the truth vector $\mathbf{g}_{it}$) will increase the error for camera location estimation. In this case, if it is a static feature, the error level will be 0 or very small.

Equation (11) can use at least two features to calculate camera position $\mathbf{r}_t$. As such, an efficient 2-point RANSAC (Random Sample Consensus) can be applied for outlier rejection. Using this algorithm will help reduce computational time compared to the traditional 3-point algorithm [35] and 5-point algorithm [36].

As mentioned above, equation (11) is solved by the 2-point RANSAC algorithm. This algorithm includes the following steps: Firstly, determine the number of iterations for the algorithm. Secondly, at each iteration, define a random sample set of two elements which are two random points from the 3D features point set. Then, the estimation results based on this sample will be evaluated by an error function. The steps above are done several times to find the best robot position. Finally, after all iterations, RANSAC will converge at a good robot position, but not sure if this is the best position. This RANSAC algorithm ensures fast processing time, the ability to estimate a good enough model and eliminate noise in the data set.

B. 3D Environment Reconstruction

In this section, 3D environment reconstruction task is presented (see Fig. 7). The environmental map is a local map. The local map is defined as the set of currently tracked 3D features. The 3D points are calculated from two different methods, one from the stereo camera, the other from the monocular camera. These 3D points will be transferred from the CCS to the WCS of robots at the start and is added to the local map. The 3D features added to the local map are static feature points, because these will be used to estimate robot positions at different times. For moving feature points, it will cause an error when estimating the robot position.

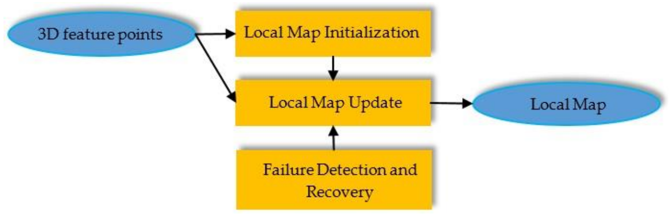

Fig. 7. Diagram of Initializing and Updating Local Maps.

At the initial time $t = 0$, the robot position is initialized. The 3D positions of the feature points will be estimated in the world coordinate from stereo camera. The 3D points are used to initialize the local map.

At the time $t(t \neq 0)$, with given robot position, the local map is updated according to the following two systems:

1) Stereo camera system: After a given period of time, the system will be restarted to update the local map. The 3D points are calculated by stereo camera.

2) Monocular camera system: During the feature tracking process, some features will be lost and lost features will be removed from the map. New features are added to the local map if the current number of features is smaller than the minimum allowable feature count. The 3D feature location $\mathbf{p}_i$ of the new feature point is estimated based on a set of τ observation of the i^{th} feature at different camera positions and is given by:

$$\mathbf{p}_i^* = \underset{\mathbf{p}_i}{\operatorname{argmin}} \sum_{t\in\tau} \|(\mathbf{p}_i - \mathbf{r}_t) \times \mathbf{k}_{it}\|^2 \tag{12}$$

where, $\mathbf{r}_t$ is the 3D position of the camera at time t in WCS, $\mathbf{k}_{it}$ is the observation vector of the i^{th} feature point at time t, $\mathbf{p}_i$ is the 3D position of the i^{th} feature in WCS.

And solve equation (12) via the following linear system:

$$\left(\sum_{t\in\tau}\left(\mathbb{I}_3 - \mathbf{k}_{it}\mathbf{k}_{it}^T\right)\right)\mathbf{p}_i = \sum_{t\in\tau}\left(\mathbb{I}_3 - \mathbf{k}_{it}\mathbf{k}_{it}^T\right)\mathbf{r}_t \tag{13}$$

where $\mathbf{r}_t$ is the 3D position of the camera at time t in WCS, $\mathbf{k}_{it}$ is the observation vector of the i^{th} feature point at time t in WCS, $\mathbf{p}_i$ is the 3D position of the i^{th} feature in WCS, τ is the set of times t that the i^{th}feature is observed on the left camera, $(\cdot)^T$ represents the matrix transpose, $\mathbb{I}_3$ is a 3×3 matrix unit.

Equation (13) is solved by basic matrix algebra, τ is defined as 2 consecutive times t-1 and t.

The 3D positions of the feature points are calculated from monocular camera will be updated in the following two cases:

- The feature points have been restored the 3D position from stereo camera system: Add the 3D position from monocular camera system to the local map.
- The feature points have not had the 3D position from stereo camera system: The feature points are added to the local map if the current number of feature points in local map is smaller than the minimum allowable feature points count.

Failure Detection and Recovery

Because the 3D positions of the feature points in local map are calculated from two systems: feature tracking by monocular camera, feature detection and matching by stereo camera, therefore, it will accumulate errors. The error of the local map at time t is calculated as:

$$\gamma = \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} \frac{\|\mathbf{p}_k^m - \mathbf{r}_t\|}{\|\mathbf{p}_k^s - \mathbf{r}_t\|} \tag{14}$$

where, $\mathbf{p}_k^s$ is the location of feature k obtained by stereo correspondence, $\mathbf{p}_k^m$ is the location of feature k obtained by monocular camera, $\mathcal{K}$is the set of 3D points to calculate the errors, $\mathbf{r}_t$ is the camera position at time t.

The system works well when $\gamma \cong 1$ and otherwise it is error. When the system has error, the system will remove all features from the monocular camera system and restart the local map based on the 3D positions of feature points from stereo camera $\mathbf{p}_k^s$.

C. *Tracking by Detection*

The goal of the paper's tracking algorithm is to solve the challenges of a moving camera which are vibration of the camera and motion of tracked object. The basic essence of the paper's tracking algorithm is the integration of single tracker and object detection to become a method called tracking by detection. The single tracker will give the state of the tracked object at every time step, moreover, its 3D position by reconstructing the environment can provide the necessary information for tracking process. A single tracker is integrated to object detection as YOLOv3 because YOLOv3 can detect objects very quickly and accurately, it can support the single tracker to find the tracked object more quickly and when the single tracker fails to track down the tracked object, YOLOv3 is a useful helper to find the object being tracked. And Tracking by Detection with 3D Environment Reconstruction can estimate the three-dimensional position of the tracked object.

This section describes the workflow of the paper's tracking algorithm, it includes four steps: Initialization, Prediction, Matching and Resampling.

Firstly, an object could be selected in a frame to track, however, the system might be given an image of object and it would be found out before tracking.

After having the bounding box of the tracked object at time t, in prediction step at time (t+1), the particle filter will generate some particles and each particle will be guided by a motion vector of the tracked object. An object detector will come in handy in this step, it will detect objects that is the same kind with the tracking object and then, some of detected objects will be chosen to add into the particle set. Object detection based on deep learning can be robust to the vibration of camera.

In the matching step, YOLOv3 will detect a number of objects of the same type as the tracked object and then select a detected object with highest matching rate with the tracked object as the current tracked object. If matching is failed, each particle will be matched with the previous object by an observation model. The correlation filter will be integrated to a deep neural net to match objects. After that, resampling could be performed if it is necessary.

The tracking pipeline of the paper is as Fig. 8:

1) Initialization: In the first frame, an object will be chosen and it is expressed as four parameters which are location and size.

$$p(t) = [px(t),\ py(t),\ pw(t), ph(t)] \tag{15}$$

$px(t),\ py(t)$ is the position of the object at time (t).

$pw(t), ph(t)$ is the width and height of the object at time (t).

$F(t)$ and $F(t-1)$ are sets of feature (SURF) points of the object at time t and time t – 1 from left or right camera.

$$F(t) = (f_t^0, f_t^1, \dots, f_t^n) \tag{16}$$

$$F(t-1) = (f_{t-1}^0, f_{t-1}^1, \dots, f_{t-1}^n) \tag{17}$$

Using a matching feature algorithm, feature points which are not matched are eliminated and the remaining K pairs of feature points are treated as a set of K motion vectors.

$$f_v(t) = (f_t^0, f_t^1, \dots, f_t^K) \tag{18}$$

With each f_t^i, i ∈ K, is a camera's motion vector of a pair of feature points.

This set is used to compute camera's motion vector of the object at time t.

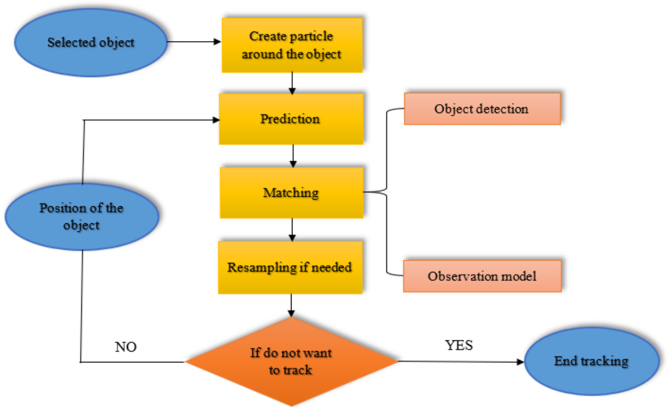

Fig. 8. The Tracking Pipeline.

2) Prediction: In this step, the state of the object will be predicted in the next frame.

The particle filter generates some particles around the previous object and each object has a weight which is the importance of a particle.

For example, in Fig. 9, the motorbike is being tracked:

The yellow box indicates the object is being tracked. The green box indicates a particle around the object.

The particle is guided by a camera's motion vector which is obtained by matching SURF features in two consecutive frames got from the left or right camera. Stereo camera is used to get two current frames instead of using two consecutive frames as [38]. After matching features,

The predicted state of tracked object is $\hat{p}(t)$:

$$\hat{p}(t) = p(t-1) + f_v(t) + q(t) \tag{19}$$

$p(t-1)$ is the state of the tracked object in previous frame (t-1).

$f_v(t)$ is a camera's motion vector at time t computed by feature matching.

$q(t)$ is Gaussian noise added.

After that, a number of particles will be generated around $\hat{p}(t)$ and each particle is called $\hat{p}_i(t)$.

With,

$$\hat{p}_i(t) = \left[\widehat{px}(t), \widehat{py}(t), \widehat{pw}(t), \widehat{ph}(t)\right] \tag{20}$$

$$\text{SCALE\_CONSTANT} = 0.2 \tag{21}$$

And,

$$\widehat{px}_t^i = \frac{\widehat{px}(t)}{2} + random(0,1) * \frac{\widehat{pw}(t)}{2} \tag{22}$$

$$\widehat{py}_t^i = \frac{\widehat{py}(t)}{2} + random(0,1) * \frac{\widehat{ph}(t)}{2} \tag{23}$$

$$\widehat{pw}_t^i = \widehat{pw}(t) + random(0,1) * \text{SCALE\_CONSTANT} \tag{24}$$

$$\widehat{ph}_t^i = \widehat{ph}(t) + random(0,1) * \text{SCALE\_CONSTANT} \tag{25}$$

$$f_v(t) = \frac{1}{K}\sum_i^K f_t^i \tag{26}$$

YOLOv3 [19] is used to detect objects in this step, after detecting, some objects will be selected by their IoU with the tracked object, if this value is higher than a threshold, this object will be kept and added into the particle set.

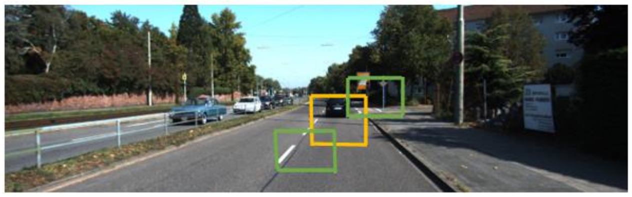

Fig. 9. KITTI Tracking: 0003 Dataset [37].

3) Matching: Each particle will be compared with the previous object by an observation (matching) model, after comparing, each particle will have a weight.

The correlation filter is presented in [5] [6] [7] will be used in observation model. This filter is obtained at time (t-1), next, it will traverse the frame at time (t), this means the filter will convolve with each region of the image from left to right, up to down. This result of a value which determine the correlation between the object at time (t-1) and the region at time (t). The higher the value is, the higher the region will be the state of the tracked object at time (t).

The correlation filter is integrated with particle filter because it does not need to traverse the whole frame, it only needs to convolve with each particle. This will reduce computational cost.

But, to use this correlation filter, it has been learned in frame (t-1). It is called as cr_f and the region of the tracked object at time (t-1) is r and M, N is width and height of this region, and $r_{m,n}$ with $m,n \in \{0,1,\dots,M-1\} \times \{0,1,\dots,N-1\}$ is the region of the tracked object at time (t-1) after translating to the right m pixel and to below n pixel.

Each $r_{m,n}$ is corresponding with a value called $g_{m,n} = e^{\frac{-(m-M/2)^2+(n-N/2)^2}{2\sigma^2}}$. This value is a Gaussian value because Gaussian value expresses how far a pixel from its center.

The correlation filter is then learned through this expression:

$$cr_f^* = agrmin_{cr_f} \sum_{m,n} \left|\left|cr_f r_{m,n} - g_{m,n}\right|\right|^2 + \lambda\left\|cr_f\right\|_2^2 \tag{27}$$

After find out cr_f^*, the weight also known as the correlation of each particle calculated by consoling cr_f^* with the region of a particle.

In this step, a deep neural network called VGG-19 is applied [14] [16]. This network is trained and has a optimal parameters. $r_{m,n}$ and the region of each particle in this step have to go through this network to obtain three feature maps called conv-3, conv-4, conv-5. And with each convolutional map, a corresponding cr_f^* will be learned. Three weights of a particle called c_1, c_2, c_3 and then the final weight of a particle is:

$$w_t^i = c_1 + c_2 + c_3 \tag{28}$$

Finally, the weight of each particle will be normalized.

$$w_t^i = \frac{w_t^i}{\sum_{i=0}^N w_t^i} \tag{29}$$

Fig. 10 is an example when a particle is passed through a deep network and feature maps are obtained. After that, each feature maps are convolved with a correlation filter to get weight values. Finally, the weight of the particle is the sum of three weight values.

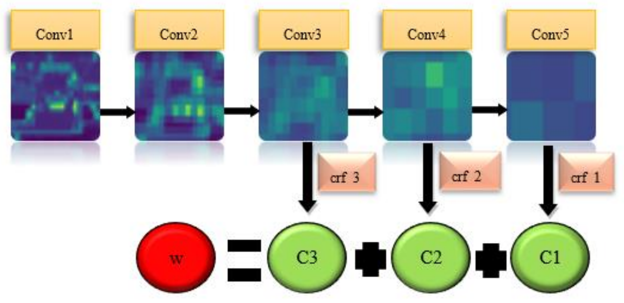

Fig. 10. Inspired from [15].

4) Resampling: After matching step, the effective sample size is computed and it is compared with a threshold to perform resampling.

$$\widehat{N} = \frac{1}{\sum_{i=1}^{N} (w_i^t)^2} \quad (30)$$

In this step, YOLOv3 [19] will be used to detect objects and choose the candidate which has the highest IoU value with the tracking object to become the one which is tracked.

Object Detector is used to overcome the degeneracy problem of particle filter.

If Object Detector does not find any objects, the Roullette table will be used to resample.

The Roullet table is described as Fig. 11, all the weights of particles will be put on a Roullet table:

The wheel will be spinned N times (N is the number of particles), and N particles will be got with different weights.

5) Estimation: After the C.3 step, the estimated state of tracked object is calculated by this expression:

$$p_t = \sum_{i=0}^{N} w_t^i \hat{p}_t^i \quad (31)$$

Recently, there are various of object detection algorithms based on neural network. The paper explored several popular object detection DNN architecture:

Faster-RCNN [39], RetinaNet [40] achieved the high accuracy rate but they are not suitable to apply in real-time applications because their computational time are still slow.

SSD [41] achieved the accuracy rate lower than Faster-RCNN and RetinaNet and it is not robust to detect small objects.

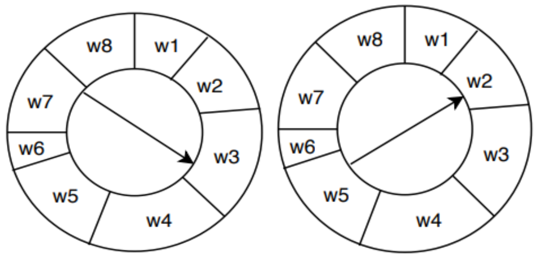

Fig. 11. Describe the Process of Spinning the Roullette Table. Inspired from [22].

TABLE I. SPEED/ACCURACY TRADEOFF ON THE AP [19]

Method	mAP	Inference time (s)
SSD321	28.0	61
SSD513	31.2	125
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198
Yolov3-320	**28.2**	**22**
Yolov3-416	**31.0**	**29**
Yolov3-608	**33.0**	**51**

YOLOv3 [19] is a suitable solution to detect objects in real-time applications, it achieves the accuracy rate ranked between the groups listed below (see Table I) but its processing speed is unsurpassed.

YOLOv3 is trained on many datasets such as: PASCAL VOC 2007 [42] and Microsoft COCO [43].

IV. EXPERIMENTAL RESULTS

The proposed method will be evaluated on the accuracy rate of camera location, tracked object location.

KITTI data set is used to evaluate the accuracy of the proposed method. KITTI dataset has many different data sets, two data sets "Raw Data" and "Object Tracking Evaluation 2012" are selected to experiment. The accuracy of the camera position estimation algorithm (camera position) is evaluated on "Raw Data" dataset. Datasets 0091, 0060, 0095, 0113, 0106 and 0005 are selected in dataset "Raw Data". Tracked object position estimation is evaluated based on the proposed tracking by detection method on "Object Tracking Evaluation 2012" dataset. These following datasets 0000, 0004, 0005, 0010, 0011, 0020 are used because they have specific objects to tracking.

The experiments are performed based on Python programming language (version 3.7.0) on computers with the following specifications: Intel Corei5 5200U 2.7GHz/ RAM 4GB / Windows 10 operating system (for camera localization); For the tasks such as tracking and 3D reconstruction, 2080 Titan GPU, Ubuntu 16.04 are used.

A. Accuracy of Camera Location Estimation

In this experiment, the data set KITTI is selected. The camera location is estimated with and without removal of moving objects.

The solution to remove moving objects in the image is described in the section 3.A.1 and Fig. 3. Distance error between estimated camera position and ground-truth camera position is estimated by formula:

$$\text{error-distance} = \frac{1}{N}\sum_{t=1}^{N}\left[(\mathbf{r}_{tx} - \text{GPS}_{tx})^2 + \left(\mathbf{r}_{ty} - \text{GPS}_{ty}\right)^2 + (\mathbf{r}_{tz} - \text{GPS}_{tz})^2\right]^{1/2} \quad (32)$$

where, $\mathbf{r}_t$ is the estimated camera location at the time t; $\mathbf{r}_{tx}, \mathbf{r}_{ty}$, and $\mathbf{r}_{tz}$ are the coordinates x, y and z of camera location $\mathbf{r}_t$, GPS_t is camera location from GPS at time t; GPS_{tx}, GPS_{ty}, and GPS_{tz} are the coordinates x, y and z of camera location from GPS_t, N is the number of frames.

From the experiment results (see Table II), estimated camera position is quite good and if the removal of moving object is considered, the better results can be achieved compared to the opposite case.

B. *Accuracy of the Tracked Object Center in 3D*

Object center is estimated as the average of all 3D points of the object being considered.

The following distances are used to evaluate errors (see Table III) between tracked object center and its ground truth object center:

$$\text{error-center}= \frac{1}{N}\sum_{t=1}^{N}\left[\left(\mathbf{c}_{est,x}(t)-\mathbf{c}_{gt,x}(t)\right)^2+\left(\mathbf{c}_{est,y}(t)-\mathbf{c}_{gt,y}(t)\right)^2+\left(\mathbf{c}_{est,z}(t)-\mathbf{c}_{gt,z}(t)\right)^2\right]^{1/2} \quad (33)$$

where $\mathbf{c}_{est}(t)$ is the center of the estimated 3D bounding box at the time t; $\mathbf{c}_{est,x}(t)$, $\mathbf{c}_{est,y}(t)$ and $\mathbf{c}_{est,z}(t)$ are the coordinates x, y and z of center $\mathbf{c}_{est}(t)$; $\mathbf{c}_{gt}(t)$ is the center of 3D bounding box from ground-truth data at time t; $\mathbf{c}_{gt,x}(t)$, $\mathbf{c}_{gt,y}(t)$ and $\mathbf{c}_{gt,z}(t)$ are the coordinates x, y and z of center $\mathbf{c}_{gt}(t)$. N is number of frames.

In this Section 4.B, a different dataset will be used to evaluate the accuracy of the tracked object center because it has ground truth data about the object's center.

From Table III, the center error in case of using YOLO has achieved better results than the opposite case.

TABLE II. Error of the Estimated Camera Position Compared to GPS between 'without Removal moving objects' and 'Removal moving Objects'

Data	Frames	Distance (m)	Without removal moving object (m)	Removal moving object (m)
0091	150	97.5	0.8449	0.8245
0060	70	0	0.0197	0.0193
0095	150	137.86	1.1654	1.0322
0113	80	16.27	0.5542	0.5440
0106	174	83.61	0.5199	0.5105
0005	150	66.78	1.5397	0.6656
Average			0.7740	0.5993

TABLE III. Evaluating the Errors between the Tracked Object center and its Ground Truth Center in 3D

Data	error-center with YOLOv3 (m)	error-center without YOLOv3 (m)
0005	0.5421	0.8608
0010	0.5470	1.7793
0011	0.4271	2.104
Average	0.5054	1.5868

C. *Accuracy of the Tracking Object Position based on Object Center and IoU in 2D*

The IoU metric is used to calculate the IoU between predicted boxes with its ground truth boxes in 2D.

In Section 4.C, the following datasets are used because of some objects exist to track, others do not have a consistent object to follow.

In Table IV, Euclide distance is used to estimate error between predicted box center and its ground truth box center. The table has shown that the IoU metric of proposed tracking method in case of using YOLOv3 is better than the opposite case.

In Table V, Euclidean distance is used to estimate the errors between the ground truth centers and the predicted centers. The table has shown that when combines YOLOv3, the Euclid distances of proposed tracking method in case of using YOLOv3 is smaller than the opposite case.

D. *Speed of the Tracking Algorithm*

Number of frames are estimated per second (FPS).

In Table VI, the paper measured the time taken to process a frame and then invert this time to get the given FPS. The table has shown that FPS of proposed tracking method in case of using YOLOv3 is about more three times faster than the opposite case.

TABLE IV. Compare the Accuracy of Estimated Object Position based on Tracking by Detection and Tracking without Detection (using metric IoU, Object Detection MethodYOLOv3)

Method Data	Average IoU 2D with object detection (YOLOv3) (%)	Average IoU without object detection (YOLOv3) (%)
0000	0.66	0.29
0004	0.41	0.11
0005	0.74	0.6
0010	0.82	0.64
0011	0.78	0.48
0020	0.65	0.6
Average	0.68	0.45
Outliers		
0018	0.08	0.03

TABLE V. Compare the Accuracy of Estimated Object Position based on Tracking by Detection and Tracking without Detection (using Euclide Distance, Object Detection Method YOLOv3)

Method Data	Average errors of centers with YOLOv3 (pixel)	Average errors of centers without YOLOv3 (pixel)
0000	16.14	234.3
0004	10.74	61.64
0005	3.66	1.47
0010	3.62	4.11
0011	5.57	14.06
0020	6.22	13.13
Average	7.66	54.79
Outliers		
0018	25.56	315.52

TABLE VI. EVALUTATE THE SPEED OF THE PROPOSEDTRACKING ALGORITHM WITH AND WITHOUT YOLO V3.

Method Data	Average FPS with object detection (YOLOv3) (Hz)	Average FPS without object detection (YOLOv3) (Hz)
0000	11	4
0004	9	3
0005	12	3
0010	12	3
0011	12	3
0020	11	4
Average	11	3
Outliers		
0018	6	3

V. DISCUSSION

The experimental results have shown that camera position is estimated quite well because the moving features are removed when estimating camera position. However, the moving features removal algorithm is still limited and should be improved in the future, though it has shown an improvement in the accuracy of the camera position when the moving features are removed. Though the error between the estimated camera location and ground truth camera location is still remaining, but it is useful in the environments such as in-doors, noisy GPS and in the cases where input of tracked object is image. The experimental results has shown the important role of visual information in MOT.

The experimental results have also shown that the processing speed is suitable for real-time applications. The speed of tracking algorithm is increased when Particle filter is integrated to YOLOv3, this is because the tracking algorithm could use either of these methods to track the object. And when it uses YOLOv3, the algorithm can run very fast. Because of this, the speed increases significantly, it achieves more three times faster than the conventional method. In comparison to accuracy, the tracking by detection method is also more effective than the single tracker method based on metric of IoU in 2D or the tracked object center.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, a unified system consisted of robot localization, environment reconstruction and object tracking based on stereo camera and IMU has been proposed.

In localization, the paper's contribution is a solution to estimate camera position and tracked object position based on stereo camera and IMU with the removal of moving features. It has shown that the accuracy rate of locatization is improved and the computational time is suitable to real-time applications.

In tracking, the paper's contributions are: (1) particles guided by a motion vectors are calculated using pairs of SURF feature points, so the direction the object is pointing to is gathered; (2) an observation (matching) model contains correlation filter and a deep neural network (VGG-19), it can deal with changes of translation of the object; (3) Tracking by detection with an object detection algorithm (YOLOv3) can support the single tracker become more accurate because it can detect more participants for particle filter, besides, it also can detect objects very fast and accurately.

Although the framework works well on the KITTI dataset, it is necessary to improve both localization and tracking algorithms.

In the localization algorithm, the algorithm should be tested on a real robot. The localization algorithm can also pave the way for the dynamic obstacle avoidance. And combination lidar with stereo camera is a novel way to explore.

In the tracking algorithm, the most time-consuming step is the matching step of the Particle Filter. In the matching step, Correlation Filter trained at each frame so the matching has increased the computational time. In the future, the Correlation Filter should be replaced by a pre-trained model such as Siamese net that can compare the features of the target at the consecutive times in real time.

ACKNOWLEDGMENT

This research is funded by Viet Nam National University Ho Chi Minh City (VNUHCM) under grant no. B2018-18-01.

Thank to Director of AIOZ Pte Ltd Company Erman Tjiputra, CTO Quang D. Tran for the valuable support on internship cooperation.

REFERENCES

[1] Q. Zhao, Z. Yang and H. Tao, "Differential earth mover's distance with its applications to visual tracking," in Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 2, pp. 274–287, 2010.

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 886-893, 2005.

[3] D. Ta, W. Chen, N. Gelfand and K. Pulli, "Surftrac: Efficient tracking and continuous object recognition using local feature descriptors," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2937-2944, 2009.

[4] D. A. Ross, J. Lim, R.-S. Lin and M.-H. Yang, "Incremental learning for robust visual tracking," in International Journal of Computer Vision, vol. 77, no. 1-3, pp. 125–141, 2008.

[5] D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, "Visual Object Tracking using Adaptive Correlation Filters," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2544-2550, 2010.

[6] H. K. Galoogahi, T. Sim and S. Lucey, "Multi-Channel Correlation Filters," in Proceedings of the IEEE International Conference on Computer Vision, pp. 3072-3079, 2013.

[7] J. F. Henriques, R. Caseiro, P. Martins and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters," in Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 3, pp. 583-596, 2014.

[8] X. Li, K. Wang, W. Wang and Yang Li, "A multiple object tracking method using Kalman filter," in Proceedings of the 2010 IEEE International Conference on Information and Automation, pp. 1862-1866, 2010.

[9] P. Kalane, "Target Tracking Using Kalman Filter," in International Journal of Science & Technology (IJST), vol. 2, no. 2, Article ID IJST/0412/03, 2012.

[10] H. A. Patel and D. G. Thakore, "Moving Object Tracking Using Kalman Filter", in International Journal of Computer Science and Mobile Computing, vol. 2, no. 4, pp. 326-332, 2013.

[11] K. Nummiaro, E. Koller-Meier and L. V. Gool, “Object Tracking with an Adaptive Color-Based Particle Filter,” in DAGM 2002: Pattern Recognition, Lecture Notes in Computer Science (LNCS, vol. 2449), pp. 353-360, 2002.

[12] S. Prabu and G. Hu, “Stereo Vision based Localization of a Robot using Partial Depth Estimation and Particle Filter,” in Proceedings of the 19th World Congress, The International Federation of Automatic Control, vol. 47, no. 3, pp. 7272-7277, 2014.

[13] S. Chen and W. Liang, “Visual Tracking by Combining Deep Learned Image Representation with Particle Filter,” in ICIC Express Letters, Part B: Applications, vol. 3, no. 1, pp. 1-6, 2012.

[14] C. Ma, J. B. Huang, X. Yang and M. H. Yang, “Robust Visual Tracking via Hierarchical Convolutional Features,” in Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence (Early Access), pp. 1-1, 2018.

[15] R. J. Mozhdehi and H. Medeiros, “Deep Convolutional Particle Filter for Visual Tracking,” in Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 3650-3654, 2017.

[16] T. Zhang, C. Xu and M. H. Yang, “Multi-task Correlation Particle Filter for Robust Object Tracking,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4819-4827, 2017.

[17] J. S. Lim and W. H. Kim, “Detection and Tracking Multiple Pedestrians from a Moving Camera,” in ISVC 2005: Advances in Visual Computing, LNCS 3804, pp. 527-532, 2005.

[18] Y. Chen, R. H. Zhang, L. Shang and E. Hu, “Object detection and tracking with active camera on motion vectors of feature points and particle filter,” in The Review of Scientific Instruments, vol. 84, no. 6, 2013.

[19] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” in University of Washington, 2018.

[20] S. Shen, Y. Mulgaonkar, N. Michael and V. Kumar, “Vision-Based State Estimation for Autonomous Rotorcraft MAVs in Complex Environments,” in Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1758-1764, 2013.

[21] L. Gong, M. Yu and T. Gordon, “Online codebook modeling based background subtraction with a moving camera,” in 2017 3rd International Conference on Frontiers of Signal Processing (ICFSP), pp. 136-140, 2017.

[22] S. Minaeian, J. Liu and Y. J. Son, “Effective and Efficient Detection of Moving Targets from a UAV’s Camera,” in IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 2, pp. 497 – 506, 2018.

[23] F. Zhong, S. Wang, Z. Zhang, C. Zhou and Y. Wang, “Detect-SLAM: Making Object Detection and SLAM Mutually Beneficial,” in IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1001-1010, 2018.

[24] D. Nistér, O. Naroditsky and J. Bergen, “Visual Odometry,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), vol. 1, pp. I-I, 2004.

[25] 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, pp. 486-492, 2010.

[26] Y. Liu, Y. Gu, J. Li and X. Zhang, “Robust Stereo Visual Odometry Using Improved RANSAC-Based Methods for Mobile Robot Localization,” in Sensors 2017, vol. 17, no. 10, 2017.

[27] Y. Xu, V. John, S. Mita et al., “3D Point Cloud Map Based Vehicle Localization Using Stereo Camera,” in 2017 IEEE Intelligent Vehicles Symposium (IV), pp. 487-492, 2017.

[28] S. Hong, M. Li, M. Liao and P. v. Beek, “Real-time mobile robot navigation based on stereo vision and low-cost GPS,” in Intelligent Robotics and Industrial Applications using Computer Vision 2017, pp. 10-15(6), 2017.

[29] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, “Speeded-Up Robust Features (SURF),” in Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346-359, 2008.

[30] M. Muja and D. G. Lowe, “Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration,” in VISAPP International Conference on Computer Vision Theory and Applications, vol. 1, 2009.

[31] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” in International journal of computer vision, vol. 60, no. 2, pp. 91-110, 2004.

[32] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81), pp. 674-679, 1981.

[33] S. Kim, K. Yun, K. Yi, S. Kim and J. Choi, “Detection of moving objects with a moving camera using non-panoramic background model,” in Machine Vision and Applications, vol. 24, no. 5, pp. 1015– 1028, 2013.

[34] R. I. Hartley and A. Zisserman, “Multiple View Geometry in Computer Vision,” Cambridge University Press, 2004.

[35] R. Haralick, C. Lee, K. Ottenberg and M. Nolle, “Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem,” in International Journal of Computer Vision, vol. 13, no. 3, pp. 331-356, 1994.

[36] D. Nister, “An efficient solution to the five-point relative pose problem,” in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. II-195, 2003.

[37] A. Geiger, P. Lenz and R. Urtasun, “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite,” in Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[38] S. Minaeian, J. Liu and Y. J. Son, “Effective and Efficient Detection of Moving Targets from a UAV’s Camera,” in IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 2, pp. 497 – 506, 2018.

[39] S. Ren, K. He, R. Girshick and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2015.

[40] T. Y. Lin, P. Goyal, R. Girshick, K. He and P Dollar, “Focal Loss for Dense Object Detection,” in 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2999-3007, 2017.

[41] W. Liu, D. Anguelov, D. Erhan et al., “SSD: Single Shot MultiBox Detector,” in Computer Vision – ECCV 2016, pp. 21-37, 2016.

[42] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results,” 2007.

[43] C. C. Lin, “Detecting and Tracking Moving Objects from a Moving Platform,” Georgia Institute of Technology, 2012.