3-Point RANSAC for Fast Vision based Rotation Estimation using GPU Technology

Danial Kamran¹, Mohammad T. Manzuri¹, Ali Marjovi² and Mahdi Karimian¹

Abstract—In many sensor fusion algorithms, the vision based RANDom Sample Consensus (RANSAC) method is used for estimating motion parameters for autonomous robots. Usually such algorithms estimate both translation and rotation parameters together which makes them inefficient solutions for merely rotation estimation purposes. This paper presents a novel 3-point RANSAC algorithm for estimating only the rotation parameters between two camera frames which can be utilized as a high rate source of information for a camera-IMU sensor fusion system. The main advantage of our proposed approach is that it performs less computations and requires fewer iterations for achieving the best result. Despite many previous works that validate each hypothesis for all of data points and count the number of inliers for it, we use a voting based scheme for selecting the best rotation among all primary answers. This methodology is much more faster than the traditional inlier based approach and is more efficient for parallel implementation of RANSAC iterations. We also investigate parallel implementation of the proposed 3-point RANSAC using CUDA technology which leads to a great improvement in the processing time of estimation algorithm. We have utilized real datasets for evaluation of our algorithm and also compared it with the well-known 8-point algorithm in terms of accuracy and speed. The results show that the proposed approach improves the speed of estimation algorithm up to 150 times faster than the 8-point algorithm with similar accuracy.

I. INTRODUCTION

Angular velocity of a robot is crucial information used as input in the Inertial Navigation System (INS) for having accurate estimation of the robot’s location in real world. Gyroscope is one of the main sensors used for the navigation of robots in the INS process which provides high rate angular velocity information; however, its measurements are accurate enough only for a short period of time [1] and after that period the measurements are mixed with some sensor errors such as biases and scale factor errors that are not necessarily constant over time. Moreover, other inertial sensors such as accelerometer or compass have their own drawbacks in specific situations.

Due to the drawbacks of gyroscopes and also other inertial sensors for having an accurate estimation of angular velocity, nowadays researchers have focused on the utilization of camera images as a new type of information for improving the accuracy of motion estimations for the robots. In several works researchers have mixed visual and inertial sources in a sensor fusion system in order to have an accurate estimation of robot’s rotation (e.g., [1]–[6]). However, the visual part of rotation estimation algorithm still needs some improvements. In some articles the rotation estimation algorithm is just applicable in indoor environments with enough number of vanishing points (e.g., [4]–[6]) which are not applicable for outdoor or unknown environments. On the other hand, feature based estimation methods [1]–[3] usually utilize geometric techniques that estimate both translation and rotation parameter jointly with extra processing. Therefore, even feature based methods are not efficient for the purpose of fast and accurate rotation estimation. These types of methods usually perform some inefficient computations because of estimating extra parameters rather than only rotation of the robot. These methods try to estimate only 3 rotation parameters (roll, pitch and yaw); however, most of them require more than 3 points (5, 6 or 8 points) which results in their inefficiency for merely rotation estimation. This situation leads to some extra execution time and power consumption of the overall system.

Due to the fact that most of the feature based estimation methods need to apply RANSAC technique [7] in order to remove outliers and estimate the final answer by considering all inlier correspondences, the number of minimum data points for estimating a primary hypothesis has a great impact on the number of required iterations for finding the best answer. This relationship can be seen in Fig. 1 that shows the number of required iterations for different numbers of minimum points having a primary answer and for different proportions of outliers in the correspondences. According to this figure, processing extra points in the RANSAC algorithm will also increase the number of iterations required for finding the best answer among different hypotheses which results in more execution time and waste of processing resources in crucial applications.

The main contribution of this paper is proposing a 3-point RANSAC algorithm for a fast and also efficient rotation estimation algorithm which can be used as a high rate source of information for sensor fusion applications. This 3-point algorithm requires fewer number of RANSAC iterations comparing with other similar methods due to separation between rotation and translation effects and concentrating merely on the rotation estimation. Moreover, we present a novel parallel implementation for the proposed RANSAC algorithm based on a voting approach for best rotation selection and parallel estimation of different hypotheses using CUDA cores. This methodology helps to improve the speed of overall RANSAC algorithm. The efficiency of the proposed algorithm is evaluated using several real datasets.

¹Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
²School of Architecture, Civil and Environmental Engineering, Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland
from Karlsruhe vision benchmark [9], [10] and is compared with the well-known 8 point algorithm [11] in both terms of accuracy and speed.

The outline of the paper is the following. In section II, we review previous related works. In section III, we introduce our 3-point RANSAC and also explain its advantages for a fast parallel implementation using GPU technology. In section IV, we express our dataset and the hardware platform we used in our experiments. We also present our experimental results the section IV. Finally, section V draws the conclusions of this article.

II. RELATED WORKS

The most well-known minimal solutions for the purpose of rotation estimation between camera images are the 5-point method by Nister [12], Stewenius et al. [13] and Kukelova et al. [14]. The 8-point algorithm is also among famous algorithms that was proposed by Longuet-Higgins [11]. The main problem for these approaches is mixing rotation and translation parameters jointly by considering fundamental or essential matrix in their solutions [15].

There are other rotation estimation algorithms which try to decouple the effect of rotation and translation from each other in order to concentrate merely on the rotation estimation part. The 6-point algorithm proposed by Kneip et al. [15] tries to estimate the rotation between two frames of the camera independently from its translation. The proposed algorithm shows a great accuracy for the synthetic data; however, it has not been validated for real datasets. Moreover, this method produces up to 20 candidates for each rotation estimation and therefore needs some extra assumptions in order to find the best solution among others. Also an optimization algorithm for generating accurate rotation between two frames has been proposed in [16]. The main drawback for this work is that it needs an initial value as a primary estimation of the rotation. According to our experiments, this algorithms is very sensitive to the initial value and without having good estimation cannot produce accurate results.

All of the algorithms expressed previously require more than 3 points for estimating the rotation between two frames. Hence, they lead to more RANSAC iterations, extra processing and finally increasing the execution time. On the other hand, a few previous works have tried to reduce the number of minimum required points for motion estimation in order to decrease the number of RANSAC iterations. In [17], Scaramuzza et al. proposed 1-point RANSAC for estimating the rotation and translation of ground vehicles by restricting the motion model and parameterizing the motion with only 1 feature correspondence. This method is just applicable for on-road vehicles due to the assumptions that it has in the motion model of the vehicle. Also other works have proposed 2-point [18] and 3-point [19] estimation algorithms which use the information from other sensors such as IMU or accelerometer in order to estimate some of motion parameters. The main defect for these methods is that they are dependent to accurate information from an inertial sensor.

III. PROPOSED METHOD

The method that we propose in this article for the purpose of fast rotation estimation between two camera frames is basically derived from a rigid body transformation estimation algorithm developed by Arun et al. [20]. Arun’s algorithm decouples the effect of translation on 3D point locations by subtracting the average point of each correspondent set from all its points. The main drawback of this technique is that it is applicable for 3D correspondences; however, we will show that by using some assumptions in the relationship between 2D points, this algorithm is even applicable for 2D correspondences without knowing 3D location of matched points. In the remaining parts of this section, at first we explain our assumptions about the relationship between 2D correspondences which allows us to utilize Arun’s decoupling algorithm for estimating only the rotation between them. After that we introduce our 3-point RANSAC approach for fast rotation estimation with minimum number of required iterations and also the methodology for parallel implementation of RANSAC iterations.

A. Decoupling Rotation from Translation

According to Arun et al. in [20], if we define the mean points for two 3D point clouds as the average of all 3D locations in each point cloud, the effect of translation can be omitted from 3D correspondences by subtracting the mean point of each 3D set from all its points. This is done using below equations for two point clouds $p_i$ and $p'_i$:

$$q_i = p_i - \frac{1}{N} \sum_{k=1}^{N} p_k$$

$$q'_i = p'_i - \frac{1}{N} \sum_{k=1}^{N} p'_k$$

Where $p_i$ and $p'_i$ are the 3D correspondent points in each point cloud and $N$ is the size of each point cloud. Using
the above equations, we can assume that just the rotation transformation exists between the two 3D point clouds:

\[ \mathbf{q}_t' = R \mathbf{q}_t \]  

(3)

Now in order to estimate the rotation matrix (R) between two sets, Arun’s algorithm defines H as the covariance matrix between \( \mathbf{q}_i \) and \( \mathbf{q}_t' \) points (as following). Then it derives the rotation matrix from computing the Singular Value Decomposition (SVD) of H:

\[ H = \sum_{i=1}^{N} \mathbf{q}_i \mathbf{q}_t'^{\mathsf{T}} \]  

(4)

\[ H = U \Lambda V^{\mathsf{T}} \]  

(5)

\[ R = V \Lambda^{\mathsf{T}} U^{\mathsf{T}} \]  

(6)

As we expressed previously, the only limitation on Arun’s method which prevents us to apply it in our 2D-2D correspondences is that it is developed for 3D points. However, if we compute bearing vectors of 2D correspondences and assume that all of these bearing vectors point to unknown 3D locations with fixed depth from the camera center, then we can apply Arun’s algorithm in order to decouple the effect of translation and estimate the rotation matrix. The main reason which allows us to use such assumption is that the rotation matrix in Arun’s algorithm is computed from SVD of H matrix and is independent of the depth of real 3D points.

In order to prove our hypothesis, at first we compute bearing vectors from 2D correspondences [21]:

\[ b_i = \frac{1}{||K^{-1}(u_i,v_i,1)||} K^{-1}(u_i,v_i,1)^t \]  

(7)

Where \( K \) is calibration matrix for the camera and \( u_i \) and \( v_i \) are x-y coordinates for each 2D point. Now in order to apply Arun’s algorithm on the bearing vectors, we assume a fixed but unknown depth for all bearing vectors pointing to their real 3D points:

\[ \mathbf{p}_i = \lambda * \mathbf{b}_i \]  

(8)

The \( \lambda \) is unknown for us due to the fact that we are unaware of real 3D locations; however, we will see that the \( \lambda \) value does not have any effect on the computation of rotation matrix:

\[ H = \sum_{i=1}^{N} (\lambda \mathbf{b}_i) (\lambda \mathbf{b}_i')^t \]  

(9)

\[ H = \lambda^2 \sum_{i=1}^{N} b_i b_i'^t \]  

(10)

The rotation matrix based on the Arun’s algorithm is derived from the SVD of H matrix:

\[ H = U \lambda^2 \Lambda \Lambda^t \]  

(11)

\[ R = V \Lambda^t U^t \]  

(12)

The above equation proves that Arun’s algorithm is applicable even for 2D-2D correspondences but when they are all in a same plane parallel to the image plane. It means that all of the 2D matched points should have a same depth from camera center; however, usually we cannot assume such assumption for all of matched points between two camera frames due to the fact that they are related to different objects in images with different depths from camera center. In the next part of this section, we will introduce a methodology for finding 2D points with the same actual depth by applying RANSAC scheme on all matched points.

### B. 3-point RANSAC

1) Feature Detection and Matching: One of the important steps in all feature based motion estimation algorithms is finding the location of key points in two camera frames and matching them together. Actually several types of feature detectors and descriptors have been proposed for this purpose; however, the combination of FAST detector [22] and BRIEF descriptor [23] has been approved as an efficient procedure of feature detection and matching in previous works [24]. Hence, we also decided to use such combination in order to perform feature detection and matching for the purpose of producing 2D-2D correspondences as the input of our 3-point RANSAC estimator.

2) Decoupled Rotation Estimation for Matched Points with Different Depths: In order to solve the problem of depth variation for matched bearing vectors which prevents us to assume a fixed depth for them, we propose a 3-point RANSAC scheme for estimating the rotation for different triples of random bearing vectors and finally choosing the best rotation with maximum consensus. Each random triple, which creates a plane in the 3D coordinates, produces a true rotation if its related plane is parallel to the image plane (the three points have similar depths). Therefore, in the proposed RANSAC at first we generate several random triples in order to cover different combinations of matched points with various actual depths. Then, in the motion selection step, we estimate the rotation for all of these random triples. We can guarantee that each rotation matrix estimated for each random set is a true rotation matrix, if all three bearing vectors satisfy our assumption of having similar depth from camera center.

3) Selecting the Best Rotation using Histogram Voting: The main drawback for the traditional method of outlier rejection is that each hypothesis should become validated for all matched points. The execution time of this process has a direct relationship with the number of iterations and also number of matched points. In order to address this challenge, we use a histogram voting methodology for finding the best rotation with maximum number of votes among all estimations. A similar approach has been also used in the 1-point RANSAC method [17] which requires much fewer iterations. We followed a similar methodology for building the histogram on the yaw rotations between images. However, in our method, we need to investigate much more iterations in order to cover a large proportion of triples in the matched points.
The proposed 3-point RANSAC methodology for estimating the rotation between 2D correspondences has these advantages compared to other rotation estimation algorithms:

- **Fewer points for a primary estimation**: The minimum number of required points for estimating the rotation is 3, which is much lower than 5, 6, 7, or 8-point algorithms [11]–[13], [15].

- **Fewer RANSAC iterations**: Requiring only 3 points to have a primary estimation, the RANSAC algorithm needs much lower iterations in order to cover a majority of combinations of 3 points from all correspondences.

- **Voting based selection**: Histogram voting approach has a great impact on the improvement of execution time due to the fact that there is no need to calculate number of inliers for each hypothesis. We will also see that this method has a great impact on the independent and parallel implementation of RANSAC iterations.

C. GPU Implementation

As we expressed previously, one of the main challenges for the RANSAC algorithm is serial execution of its iterations which leads to a huge processing time for the rotation estimation process. In this paper, we introduce a novel parallel implementation of RANSAC using CUDA technology which performs all of RANSAC iterations separately in each CUDA thread. This methodology has been also used previously in [8] for the 8-point RANSAC algorithm without any concentration on the rotation estimation. Moreover, selecting the best rotation of all iterations using the voting scheme, in the methodology which is proposed in this paper there is no need to calculate number of inliers in each CUDA core which causes all of parallel threads to execute completely independent from each other.

IV. EXPERIMENTAL RESULTS

In this section, at first we introduce datasets and hardware platform that we have used in our experiments. Then we explain the procedure of detecting key points and matching them that is needed for the purpose of generating 2D correspondences between two camera views. Finally, we will report the result of our experiments for applying our method on one of datasets and compare its accuracy and speed with 8-point algorithm.

A. Datasets and Hardware

We have used Karlsruhe [9], [10] datasets in our experiments which is a famous benchmark for evaluating several vision based motion estimation algorithms e.g., [25]. Fig. 2 shows a sample frame from the Karlsruhe dataset that gives a general imagination about the location of camera which is attached on the top of a ground vehicle. Angular rotation speed from a high quality IMU is assumed as the ground truth for this dataset and is used for the evaluation of our algorithm. Moreover, due to the fact that the ground vehicle only has significant changes in the yaw degree rotation, we merely investigate the yaw rotation in order to compare our method with ground truth data and also with the 8-point algorithm.

The system on which the proposed RANSAC has been evaluated is equipped with a Dual-Core Pentium CPU running at frequency of 2.8 GHz and 4 gigabytes of RAM. Moreover, the CUDA device which we have utilized for GPU implementation is a NVIDIA GeForce GT 730 with maximum frequency of 902 MHz and 2 Multiprocessors containing 384 CUDA cores. The GPU code has been developed using CUDA version 7.5 [26].

B. Accuracy of 3-point RANSAC

Now in order to evaluate the accuracy of our 3-point RANSAC, we first compare it with the well-known 8-point algorithm which estimates both translation and rotation between two camera frames using the Fundamental matrix method [11]. For this purpose, we have compared the yaw rotation of proposed method with the 8-point algorithm for two datasets of Karlsruhe benchmark. The result for one of these datasets is shown in Fig. 3 which depicts estimated yaw degree using our 3-point RANSAC and also 8-point RANSAC for different numbers of RANSAC iteration. According to these diagrams, the proposed 3-point RANSAC does not have a good accuracy compared to the 8-point method for small number of iterations (N=300). The main reason is that we have used the voting scheme for selecting the best rotation, which requires a relatively large number of iterations in order to have a comprehensive selection. However, for higher iterations (N=15000) the proposed algorithm has a comparable accuracy due to selecting more samples. Therefore, we can conclude that the proposed 3-point RANSAC algorithm requires more iterations because of using voting scheme for selecting the best rotation. Although more iterations can lead to longer execution time (which is considered as a drawback in previous works), we will show that by using the GPU implementation and also because of independent execution of iterations due to the voting scheme, we can achieve a great improvement in the performance of 3-point RANSAC.

C. Parallel Implementation using CUDA

One of the main advantages of our method comparing with other RANSAC based estimation algorithms is using voting scheme for selecting the best rotation among several hypotheses. This approach prevents parallel threads to compute the number of inliers for their estimated solution and leads to a much faster implementation. This fact is depicted in Fig. 4 which shows a great improvement in
the execution time of our 3-point RANSAC. According to this figure, the execution time of 8-point algorithm has a direct relationship with the number of RANSAC iterations; however, the GPU implementation of our 3-point approach does not have any significant relationship to this parameter. This is because of independent implementation of parallel threads in our method. We have also computed the achieved speed-up in the execution time for the GPU implementation comparing with serial implementation for the 8-point and our 3-point RANSAC algorithms which are depicted in Fig. 5 and Fig. 6 respectively. Comparing these two diagrams (note the difference in the scale of vertical axis for these two diagrams), we conclude that GPU implementation has a better improvement in the speed of 3-point algorithm due to the fact that it has much less computations and also uses the voting scheme for selecting the best rotation.

D. The impact of Number of Features

One of the main parameters which has a great impact on the speed of motion estimation algorithm is the number of detected features in the camera images. Although using more feature points can increase the accuracy of estimated values, it also can lead to more processing and finally longer execution time for both feature matching and RANSAC steps. In the RANSAC algorithm, inlier selection is the main part which its execution time has a great relationship with the number of matched points. However, using the voting approach for finding the best rotation, the performance of our proposed 3-point RANSAC does not get any adverse effect from the number of detected features. This fact is shown in Fig. 7 which plots the execution time of overall motion estimation for serial 8-point, parallel 8-point and parallel 3-point algorithms using different numbers of matched points. Comparing these diagrams, we conclude that by increasing the number of matched points, the execution time for both serial and parallel 8-point algorithms will increase; however, the execution time of our 3-point algorithm does not have any dependency on the number of points. On the other hand,
the processing time for feature matching step has a great dependency with the number of matched points which we do not have any control on it. Therefore, we conclude that the overall speed of estimation algorithm is restricted by feature extraction and matching process, as the parallel 3-point RANSAC has a relatively constant execution time for different numbers of key points.

V. CONCLUSIONS

In this paper, we showed that by decoupling the effect of translation and rotation from each other, we can utilize a 3-point RANSAC algorithm for the purpose of merely rotation estimation between camera views. We also used a voting based approach for choosing the best rotation among several hypotheses. Moreover, we explained the methodology of parallel implementation for the proposed 3-point algorithm using the GPU technology. One of the main advantages for the proposed 3-point method is that it allows parallel threads on the GPU to execute completely independent from each other. We evaluated the accuracy and speed of our approach with the 8-point algorithm and showed that it improves the performance of the rotation estimation algorithm up to 150 times faster than parallel implementation of 8-point algorithm. Still, the proposed approach has a similar accuracy with the 8-point algorithm.

REFERENCES