MC-RANSAC: A Pre-processing Model for RANSAC using Monte Carlo method implemented on a GPU

Priyank Trivedi
School of Computing Sciences and Engineering
VIT University
Chennai, India
priyank.trivedi2010@vit.ac.in

Tejaswi Agarwal
School of Computing Sciences and Engineering
VIT University
Chennai, India
tejaswi.agarwal2010@vit.ac.in

K. Muthunagai
School of Advanced Sciences
VIT University
Chennai, India
muthunagai@vit.ac.in

***Abstract*— RANSAC is a repeating hypothesize-and-verify procedure for parameter estimation and filtering of noise or outlier data. In the traditional approach, this algorithm is evaluated without any prior information on the set of data points which leads to an increase in the number of iterations and compute time. In this paper, we present a GPU based RANSAC algorithm with pre-processing of the assumed sample set of hypothetical inliers by Monte Carlo method. Based on our implementation and results using the Point Cloud Library and NVIDIA CUDA framework for data intensive tasks we obtain significant improvement in the performance of plane segmentation algorithm over the randomly sampled subset of hypothetical inliers. The final consensus set is formed with less number of iterations using our pre-processing model. We can conclude that a pre-processed sample set of hypothetical inliers results in a faster determination of the consensus set consisting of maximum inliers**

Keywords— Monte-Carlo method, RANSAC, GPU, Plane Segmentation

I. INTRODUCTION

RANdom **SA**mple **C**onsensus (RANSAC) [1] is a powerful technique to estimate the parameters of a geometric model. RANSAC estimates a subset of data that fits the model, while iteratively reject the outliers, (data which do not fit the model) and accepts the inliers, (data which are consistent with the relation). RANSAC functions in a hypothesize-and-verify framework where a minimal subset S of the input data is randomly selected and sample parameters are estimated from this subset for the verification of the model. This assumed model is then evaluated on the complete dataset and its consistency is determined iteratively until the probability of determining a model with better inliers outrun the current best model falls below a predefined threshold (typically 1%-5%.).

RANSAC works on an assumption that a "good" assumed sample set which is free from outlier data would lead to an enhanced model with maximum support. Therefore, selection of this assumed minimal subset S of the data forms the key evaluation point of this iterative algorithm. Although, numerous such minimal subsets can be assumed from the dataset, it can be shown that for a threshold value of 95% confidence limit, enough successful models with maximum support can be drawn on an average [1].

As stated above, RANSAC involves a two-step process where the first step deals with optimizing the sampling technique and the second step deals with the verification model process. There have been numerous efforts to optimize the two step process in order to enhance the performance of the system. Such efforts have utilized robust statistics [2, 3] in the pre-evaluation stage to filter the outliers. The comparative analysis of such techniques has been elaborated in [4]. Although the traditional RANSAC algorithm is sequential, a parallelized version was proposed in which multiple unique minimal subsets have been tested to form the final consensus set of inlier data [5]. Nister [6] explained a preemptive RANSAC framework, where a fixed number of hypotheses are evaluated in parallel. This approach has been successfully applied in various domains such as image and video processing for plane segmentation, registering 3D point cloud data sets and so on.

In this paper, we implement the pre-processing of the input data through the Monte Carlo Method to assume a better sample set for the RANSAC algorithm on the GPU. The algorithm is applied for the Plane Segmentation using Point Cloud Library [6].We test the performance of the algorithm on the NVIDIA CUDA framework and compare it with the RANSAC algorithm with random sampling against the proposed Monte Carlo based sampling method.

The rest of the paper is organized as follows. Section 2 describes the background and motivation of the problem statement. Section 3 explains the proposed model with a detailed algorithm. Section 4 depicts the experimental results and analysis for a 3-D input data captured by a depth-sensing camera. Section 5 elaborates upon the conclusion and future works.

II. MOTIVATION AND BACKGROUND

Over the last few years GPUs have transformed from being just a graphic rendering device to being an integral part of a computing system providing high throughput for a parallelizable problem. The GPU architecture benefits from a massive fine-grained parallelization, as they are able to execute as many as thousands of threads concurrently. Attempts to utilize GPUs not only for manipulating computer graphics, but also for different purposes, lead to a new research field, called "General-purpose computation on GPU (GPGPU)". Due to an extremely large data set while feature matching with 3-D

imaging, RANSAC is computationally intensive. Making use of the inherent parallelism in RANSAC and the high computational benefits of the GPU, for our implementation, we use RANSAC with the Monte Carlo method for pre-processing. Figure 1 shows the evolving computational power of the CPU and the GPU in a billion floating point operations per second (GFLOPS).

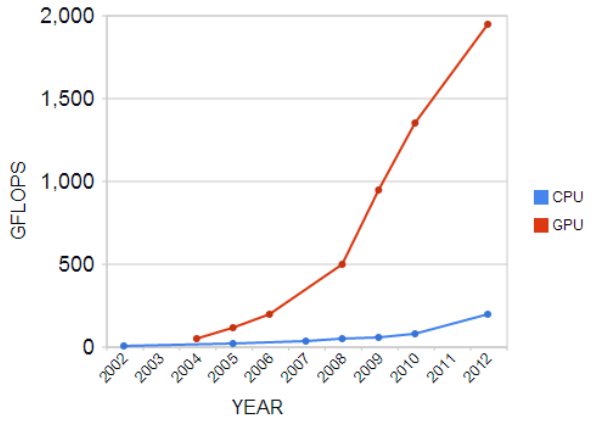

Figure 1:Evolving Computational power of CPU and GPU

The main motivation behind our work was to use another proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

pre-processing model for RANSAC than the conventional pre-processing such that the running time of RANSAC computation could be reduced. Numerous extensions and improvements in RANSAC have been proposed over the years. T. Sattler et al, in their SCRAMSAC model, use a special consistency filter to improve its efficiency [8]. OChum and J. Matas proposed an optimal randomized RANSAC using Walds theory of sequential decision based on the sequential probability ratio test (SPRT) [9].

Sunglok Choi et al, in their work, Performance evaluation of RANSAC family, gives a detailed overview on the performance of RANSAC and its extensions [10]. With major extensions and improvements, geometric verifications prove to be a major cost considerable task in RANSAC.

Xiaoyan Wang et al[11], in their work propose a reliable RANSAC using pre-processing model based on a bucketing model and verified it on the CPU. In this paper, we use Monte Carlo to pre-process the data input to RANSAC such that a better sample can be chosen over a random sample to reduce the number of iterations to arrive at the solution. The GPU implementation and verification of pre-processing model helps us extend it to a host of new applications, requiring large scale real time video or image processing. and "outliers", that is, the data inconsistent with model parameters. The traditional RANSAC randomly assumes a sample minimal subset S of some preset size from the initial set in order to hypothesize a correspondence rejection geometric model. To verify this subset S against the remaining correspondences and the amount of inliers', an iteration process is followed by iteratively selecting a random subset from the original data, i.e. hypothetical inliers, until a consensus set is determined.

In order to illustrate the RANSAC algorithm, consider the confidence limit p, number of times the sampling is done being N, the equation is :

$$N = \frac{log(1-p)}{log(1-\psi^s)} \quad (1)$$

,where s is mean of the minimal size of the sampling subset to hypothesize the geometric model, and ψ represents the probability of a point being an inlier, or (1- ψ) being the probability of a point being an outlier.

In our proposed framework, we utilize the stochastic nature of the random selection of data and use the Monte Carlo method to sample the assumed minimal subset S. In the Monte Carlo method, the probability ψ defined above is guided by a probability mass function $f_X(x)$, x being discrete data points, which is greater than zero on a set of values X. Therefore the expected value of a function $g(x)$ is:

$$E(g(x)) = \sum_{x \epsilon X}\big(g(x).f_X(x)\big) \quad (2)$$

Considering an n-sample of X's, ($x_1, \ldots . x_n$), and mean of $g(x)$over X, with Monte Carlo estimate,

$$g'(x) = \frac{1}{n}\sum_{i=1}^{n} g(x_i) \quad (3)$$

of $E\big(g(x)\big)$.

We consider a 3D image with millions of data point to sample a minimal subset S exploiting the core capability of the GPU based architecture to enhance the performance of the RANSAC algorithm. Figure 2 demonstrates the Monte Carlo approximations to distributions to approximate the whole probability distribution by $P(X_t \mid X_0 = x_0)$ by Monte Carlo method.

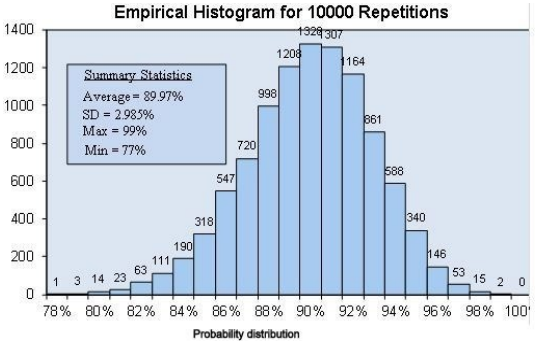

Figure 2: Empirical Histogram for 1000 Repetitions

RANSAC, being a stochastic optimization method, its efficiency can be improved by a GPU based Monte Carlo sampling method. The basic idea behind a processing model is to form improved final consensus set for better correspondence matching. We deploy the Monte Carlo sampling method on GPU to exploit the data intensive iterative process. The CUDA

enabled GPU performs the sampling iteratively to produce faster sampling techniques, thereby enhancing the performance of the complete RANSAC algorithm. Such a GPU based preprocessing model converges to form better consensus set with an enhanced performance.

A. Algorithm

The algorithm below demonstrates the working of the GPU based RANSAC algorithm with sampling by Monte Carlo method where n is the data contaminated with outliers with N being the number of iterations.

Algorithm 1 GPU based RANSAC with sampling Monte Carlo Method

1. **Hypothesis generation:**
   - **a.** maxConsensusSet ← ∅
   - **b.** N ← $\binom{n}{s}$
   - **c.** Iterate N times:
     - i. Sample a minimal subset E' of size s by Monte Carlo method.
     - ii. **for** each data point x_i in X match for correspondence .
       1. **if** satisfies the threshold condition, $N = \frac{log(1-p)}{log(1-\psi^s)}$ **then**
       2. maxConsensusSet ←S, potential set of inliers I'
2. **Verification Process:**
   - **a.** **for** each data point x_i in I'
     - i. Executing standard RANSAC on set I'
   - **b.** **Return** solution and set of inliers I

III. EXPERIMENTAL ANALYSIS AND RESULTS

In order to demonstrate the efficiency in execution time and enhanced performance, this section presents the experimental results over the algorithm of plane segmentation in 3D image processing using Point Cloud Library [7]. The input to the RANSAC algorithm is a set of observed values of 3D data points, a parameterized model which can be fitted to the observations along with some confidence parameters.

We perform the experiments using a depth-sensing camera, Microsoft Kinect to capture 3D data points. After capturing the data, we perform the filtering process to remove the noisy values. Figure 3.a represents the 3D point cloud data captured through the camera and Figure 3.b shows the 3D point cloud scene with plane segmentation. Also Figure 4.a represents the 3D data from a different angle with Figure 4.b showing the scene with plane segmentation.

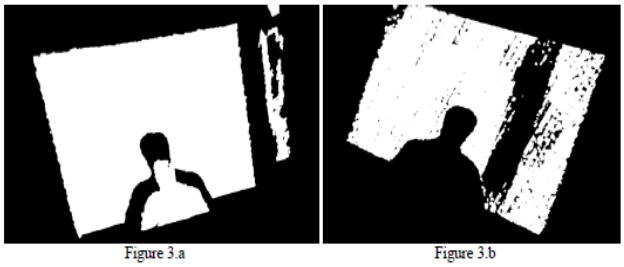

Figure 3.a Figure 3.b

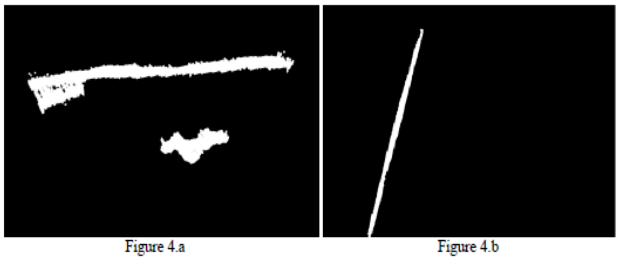

Figure 4.a Figure 4.b

Sample test case of 3d data of MC-RANSAC

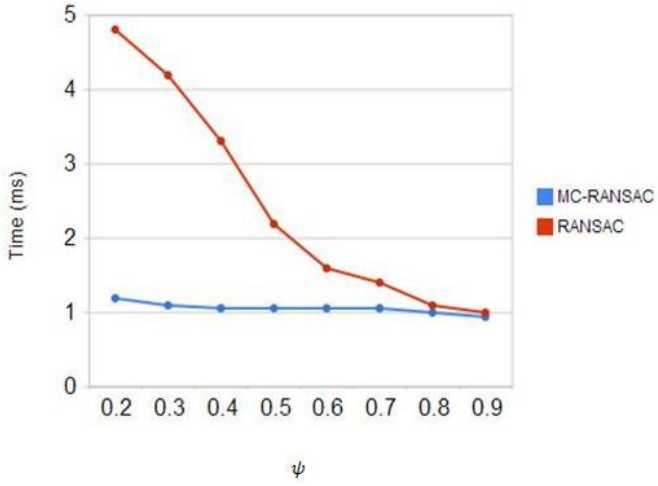

Figure 5: Performance of MC-RANSAC vs RANSAC

Figure 5 presents the performance of a GPU based pre-processed Monte Carlo RANSAC, MC-RANSAC, algorithm against the random sampling based RANSAC algorithm. The algorithm is tested on the plane segmentation of 3D point cloud data with the threshold parameter equal to 0.6 and confidence parameter as 95%. From the graph, we can conclude that ψ, which is guided by,

$$\Psi = \sqrt[s]{1 - \sqrt[N]{1-p}}$$

takes less number of iterations for $\psi \leq 0.7$ with MC-RANSAC and is almost the same for $0.8 \leq \psi \leq 0.9$.Therefore, we can conclude that a pre-processed MC-RANSAC sampling method sufficiently enhances the performance of the RANSAC algorithm and thereby increasing the usability of the algorithm in various applications such correspondence matching in

geometric model, plane segmentation in 3D image processing and numerous other computer vision applications.

IV. CONCLUSION

In this paper, we proposed MC-RANSAC, a pre-processing model for RANSAC using the Monte Carlo method on the GPU for video processing applications. The results of MC-RANSAC on GPU were compared with the traditional RANSAC and a significant change in the number of iterations was observed. With significant advances in accelerators, our model performed better than the existing RANSAC extensions proposed on the CPU. In order to scale the proposed algorithm, better hardware capabilities would sufficiently enhance the performance. Also, enhanced sampling techniques can further increase the determination of the consensus set with minimum number of iterations.

REFERENCES

[1] Fisher, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM 24(6), 381–395 (1981).

[2] Matas, J., Chum, O.: Randomized RANSAC with Td,d test. Image and Vision Computing 22(10), 837–842 (2004)

[3] Matas, J., Chum, O.: Randomized RANSAC with Sequential Probability Ratio Test. In: Proc. ICCV, pp. 1727–1732 (2005)

[4] Rahul Raguram1, Jan-Michael Frahm1, and Marc Pollefeys1,2; A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus.In: ECCV 2008, Part II, LNCS 5303, pp. 500–513, 2008. Springer-Verlag Berlin Heidelberg 2008

[5] Fijany, A. Diotalevi, F, A cooperative search algorithm for highly parallel implementation of RANSAC for model estimation on Tilera MIMD architecture, IEEE Aerospace Conference, 2012.

[6] Nister, D.: Preemptive RANSAC for live structure and motion estimation. In: Proc. ICCV, vol. 1, pp. 199–206 (2003).

[7] Point Cloud Library, www.pointclouds.org

[8] Torsten Sattler, Bastian Leibe and Leif Kobbelt, 'SCRAMSAC: Improving RANSAC's Efficiency with a Spatial Consistency Filter'

[9] Matas J, Randomized RANSAC using Sequential Probability Ratio Test, In: ICCV 2005. Tenth IEEE International Conference on Computer Vision, 2005.

[10] Sunglok Choi, Taemin Kim, and Wonpil Yu (2009). "Performance Evaluation of RANSAC Family". In: Proceedings of the British Machine Vision Conference (BMVC).

[11] Xiaoyan Wang,Hui Zhang and Sheng Liu,*Reliable RANSAC Using a Novel Preprocessing Model,* Computational and Mathematical Methods in Medicine Volume 2013.