Understanding 3D Shapes Being in Motion

Received 10th October 2012; accepted 22nd November 2012.

Janusz Bedkowski

Abstract:
This paper concerns a classification problem of 3D shapes being in motion. The goal is to develop the system with real-time capabilities to distinguish basic shapes (corners, planes, cones, spheres etc.) that are moving in front of RGB-D sensor. It is introduced an improvement of SoA algorithms (normal vector computation using PCA Principal Component Analysis and SVD Singular Value Decomposition, PFH – Point Feature Histogram) based on GPGPU (General Purpose Graphic Processor Unit) computation. This approach guarantee on-line computation of normal vectors, unfortunately computation time of the PFH for each normal vector is still a challenge to obtain on-line capabilities, therefore in this paper it is shown how to find a region of movement and to perform the classification process assuming the decreased amount of data. Proposed approach can be a starting point for further developments of the systems able to recognize the objects in the dynamic environments.

Keywords: *RGB-D camera, point cloud, normal vector estimation, point feature histogram, parallel programming*

1. Introduction

Understanding shapes being in motion is still an open problem in mobile robotics especially if we would like to perform this task on-line. Understanding changes in dynamic environment is still a problem not only because of the limitation of the available sensors but also because of the computational complexity of algorithms performing this task. Currently there are available sensors (3D laser VELODYNE or RGB-D camera like KINECT) that can provide accurate 3D data for INDOOR and OUTDOOR environments. This sensors provide data on-line, therefore it could be possible to detect changes in the environment in short intervals of time. Unfortunately there is no existing approach to perform computation on such data, therefore it is proposed to improve State of The Art by the usage of NVIDIA GPGPU with CUDA (Compute Unified Device Architecture). Using GPGPU is a promising choice because it can run thousand of kernels (functions performed on GPGPU) using decomposed 3D data set. The problem of understanding 3D shapes being in motion can be decomposed, therefore we can perform computation for each single 3D point in parallel. Unfortunately it is possible only for normal vector computation and motion-detection, the Point Feature Histogram (PFH) is still demanding problem because of its computational complexity.

2. Related Work

Currently we can observe numerous research related with modern RGB-D KINECT like sensors. In [1] Kurt3D, data from KINECT sensor is shown, which gives an impression that State of the Art (SoA) already offers efficient mobile platforms equipped with advanced sensors for observing dynamic environments. Semantic objects identification [2] is well known research direction, unfortunately currently related with the static environments [3], usually extracted from 3D laser data [4]. In [5] a model of an indoor scene is implemented as a semantic net, this approach is also used in [6]. In [7] the location of features is extracted by using a probabilistic technique (RANSAC: RANdom SAmple Consensus) [8]. It was shown in [9] that the region growing approach, extended from [10] using k-nearest neighbor (KNN) search, is able to process unorganized point clouds.

The research presented in this paper is inspired by the work [11], where authors were able to recognize several shapes such as corners, cylinders, edges, planes, torus etc. They proposed Point Feature Histogram(PFH) technique to distinguish different classes. Unfortunately the performance of proposed Point Feature Histogram algorithm was not satisfactory, therefore authors proposed faster approach – Fast Point Feature Histogram (FPFH) [12].

Based on The State of the ART it is claimed that the problem of understanding shapes being in motion is not yet discussed, therefore it was decided to develop system that can be a starting point to solve mention problem.

3. Normal Vector Estimation

Estimating the surface normal is performed by PCA – Principal Component Analysis (Figure 1) of a covariance matrix C created from the nearest neighbors of the query point. The main contribution was to develop the PCA solver based on Singular Value Decomposition SVD method that can be performed in parallel for each query point at one single step. In last step of the algorithm it is checked if normal vectors are consistently oriented towards the viewpoint and flip otherwise.

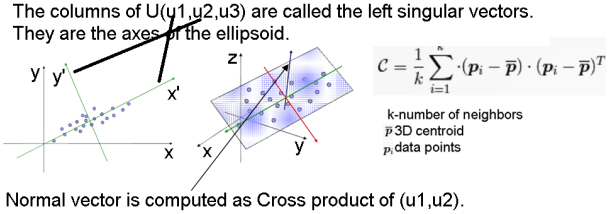

Fig. 1. Estimating the surface normal is performed by PCA – Principal Component Analysis of a covariance matrix C created from the nearest neighbors of the query point.

3.1. Parallel Implementation

Figure 2 demonstrates the parallel implementation of normal vector estimation in NVIDIA CUDA framework. The idea was to perform normal vector estimation in two steps assuming performing computation for each query point in parallel. First step is to compute covariance matrices and store the result in the GPGPU's shared memory. In second step normal vector estimation is performed using SVD (Singular Value Decomposition) method for each query point in parallel. The main contribution was to implement the CUDA kernels for covariance matrix computation and SVD solver. It is important to emphasize the NNS (Nearest Neighborhood Search) method in this approach. It is used the fixed organization of RGB-D data (640x480 or 320x240 data points), therefore NNS (Nearest Neighborhood Search) is performed for neighbors assigned by neighboring indexes to the index of query point. Additional threshold determines the radius of search space.

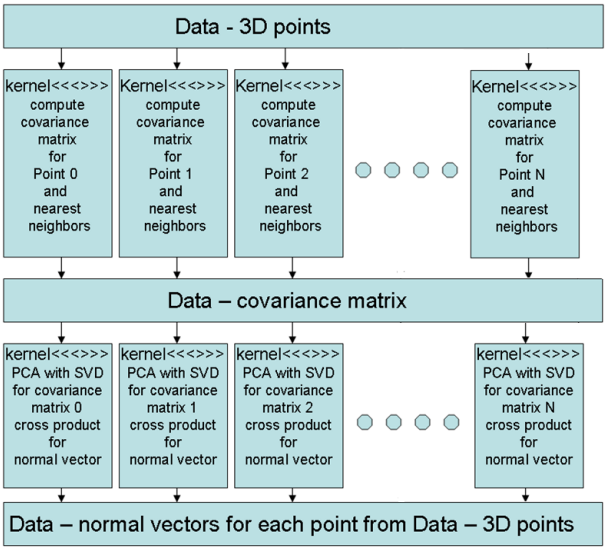

Fig. 2. Parallel implementation of normal vector estimation in NVIDIA CUDA framework.

4. Point Feature Histogram

A histogram of values encodes the local neighborhood's geometrical properties by generalizing the mean curvature at a point p. This method provides an overall density and pose invariant multi-value feature [11]. Figure 3 demonstrates the PFH – Point Feature Histogram for query point. This method provides a possibility to distinguish several types of shapes such as plane, cylinder, corner, sphere etc.

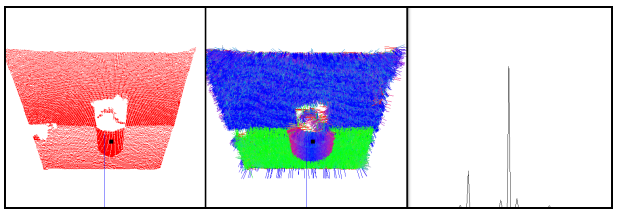

Fig. 3. Normal vector estimation visualization and Point Feature Histogram for query point.

4.1. Parallel Implementation

To improve the performance of Point Feature Histogram (PFH) computation CUDA based parallel computation is used. Current implementation computes 64 histograms at single step. Figure 4 shows an idea. For the quantitative comparison with State of the Art (PCL – Point Cloud Library) the PFH is composed of three features, therefore the dimension is 125 (5x5x5).

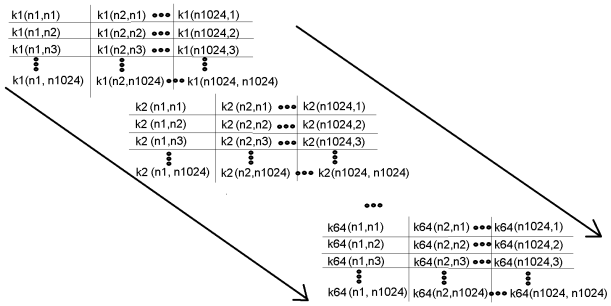

Fig. 4. The idea of parallel implementation of 64 PFH's computation at single step. The maximum amount of nearest neighbors to a query point is 1024. It has to be performed the computation for each pair of normal vectors in neighborhood ([11]), therefore kernels are organized into 1024x1024x64 data structure.

4.2. PFH classification

The method used for classification is K-Means clustering, where an iterative search for the optimal k clusters is performed. Histograms that are belonging to the same class tend to be grouped together in the same cluster. The cluster label can be verified by looking at its proximity to the mean histogram of the proper shape. This approach guarantee finite amount of computation steps to obtain candidate labels. The disadvantage is the low classification result (60% – 70%), but it is not considered as a problem because in general we deal with on-line classification process.

5. Motion Detection

In the presented approach it is assumed that RGB-D sensor is static during motion detection. Motion detection is performed by comparing current RGB-D frame to the previous one. If there is a large difference in range for corresponding query points or large angle between corresponding normal vectors it is obtained the region of motion. This procedure is extremely fast and does not affect the classification procedure in the sense of computational time. Figure 5 demonstrates the ball being in motion and corresponding computed region of motion (red color).

6. Experiments

6.1. Classifier Training and Testing

To train the system we have to provide training data set composed of labeled histograms (supervised learning). The method used for classification was introduced in section 4.2. The method is using an iterative search for the optimal k cluster, therefore histograms that are belonging to the same class tend to be grouped together are are marked by the same label. During the training of the system user

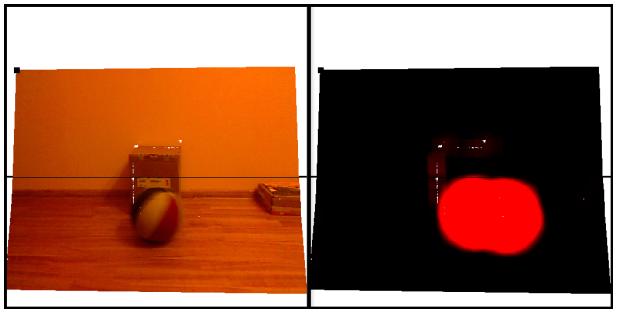

Fig. 5. Motion detection, red color corresponds to the region of motion.

is marking similar histograms by the same label. Figure 6 demonstrates the idea of collecting training data with the help of developed HMI (Human Machine Interface). The HMI is designed especially for supervised learning purpose. Example of histograms taken to the classifier training procedures are marked by pink color in Figure 6. Bottom picture shows classification result as different colors/labels for different histograms.

7. Quantitative Comparison with State of The Art

The goal of this paper is to compare quantitatively the performance of open source library PCL (Point Cloud Library) and proposed parallel implementation called cuPCL. The main problem for CUDA computation is a bottleneck related to the copy data from/to host to/from device. The host is related with CPU, device is related with GPGPU. The implementation is dedicated for NVIDIA FERMI architecture with an advantage of double floating point precision capability. Figure 7 shows the comparison between PCL and cuPCL of normal vector estimation of depth image 640x480 vertexes. The performance is measured for different radius of NNS (Nearest Neighborhood Search) procedure, it is obvious that with more radius we should expect more time for computation. The reasonable radius is 3 cm and for this value the cuPCL speed up over PCL is over 16 (GPGPU GF540M).

Based on the previous observation it is proposed to decrease amount of data from 640x480 to 320x240 of processed points. Figure 8 shows the comparison between PCL and cuPCL of normal vector estimation of depth image 320x240 vertexes. The speed up is over 10 for radius 3 cm. It is important to emphasized the decreased variance of computational time for cuPCL what is an optimistic observation to build real time systems.

Last experiment concerns the qualitative comparison of PFH (Point Feature Histogram) computation. We can observe satisfactory speed up (over 40 for radius 3 cm) for cuPCL what can give an impression that it is possible to build on-line system. Unfortunately this figure demonstrates the performance for computing 64 histograms in a single step. To perform more PFH computation we need more GPGPUs, what can be considered as a limitation for modern mobile platforms.

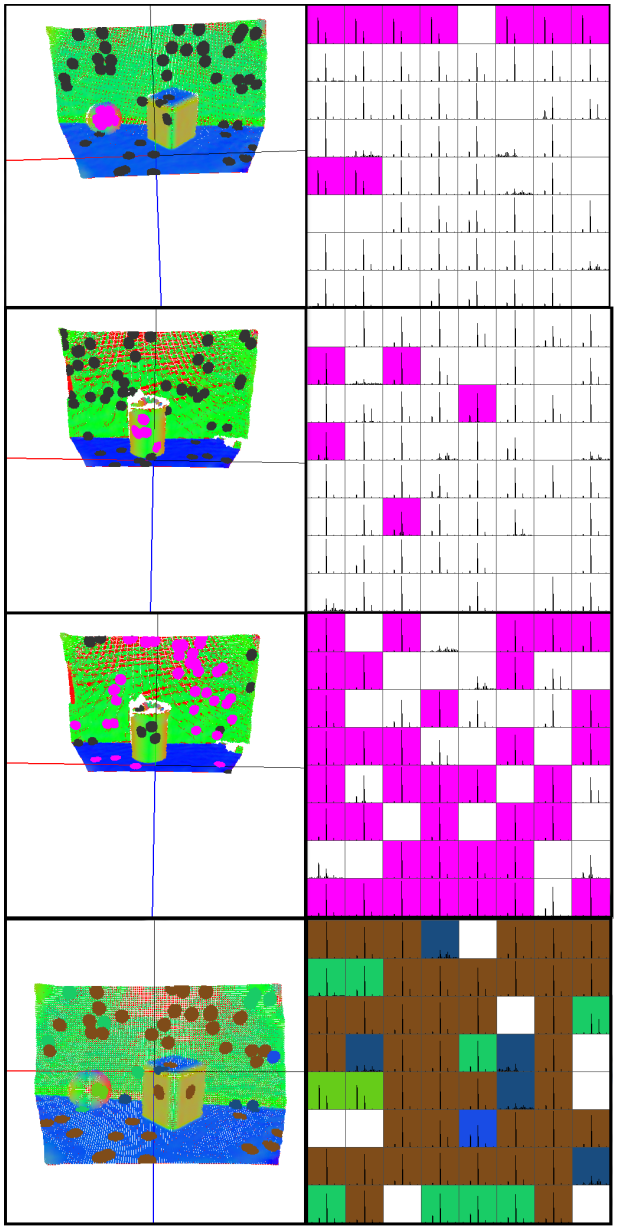

Fig. 6. Three training and one testing data sets. Histograms taken to the classifier training procedures are marked by pink color. Bottom picture shows classification result as different colors/labels for histograms of different classes (for example brown color corresponds to the wall).

8. Conclusion

In this paper new implementation capable to classify shapes being in motion is proposed. The system is able to detect regions of motion and to perform classification of 64 Point Feature Histograms on-line (between 2 and 10 frames per second, depends on GPGPU). The improvement is based on NVIDIA CUDA implementation of the algorithms that are available in State of The Art PCL (Point Cloud Library) library. The main contribution of this paper are new method for detecting motion, new parallel implementation of SVD (Singular Value Decomposition) solver for PCA (Principal Component Analysis) analysis related with normal vector estimation. The system can classify up to 64 shapes being in motion on-line with the 60% – 70% of classification result. It can process more histograms but

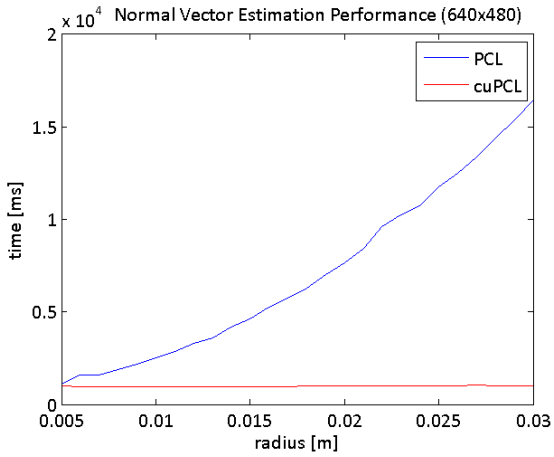

Fig. 7. The performance of normal vector estimation of depth image 640x480 vertexes computed via CPU(PCL) and GPU(cuPCL).

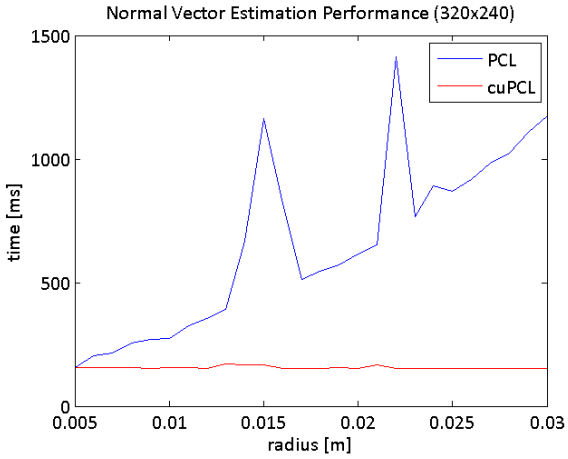

Fig. 8. The performance of normal vector estimation of depth image 320x240 vertexes computed via CPU(PCL) and GPU(cuPCL).

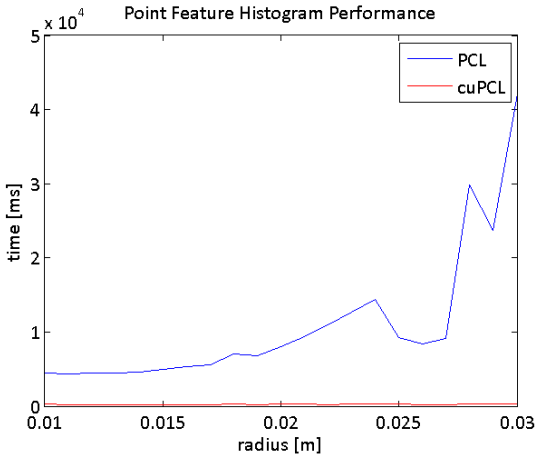

Fig. 9. The performance of 64 Point Feature Histograms computed via CPU(PCL) and GPU(cuPCL).

it will determine the use of more GPGPUs assuming the same performance.

AUTHOR

Janusz Bedkowski – Intitute of Mathematical Machines, Institute of Automation and Robotics, Warsaw, Poland, e-mail: januszbedkowski@gmail.com.

Acknowledgements

This work is performed under the funding of Polish National Center of Science, project "Methodology of semantic models building based on mobile robot's observations(2012-2015)" grant agreement UMO-2011/03/D/ST6/03175.

References

[1] J. Elseberg, D. Borrmann, A. Nüchter, "Efficient processing of large 3d point clouds". In: *Proceedings of the XXIII International Symposium on Information, Communication and Automation Technologies (ICAT11)*, Sarajevo, Bosnia, 2011.

[2] A. Nüchter, J. Hertzberg, "Towards semantic maps for mobile robots", *Robot. Auton. Syst.*, vol. 56, 2008, pp. 915–926. doi:10.1016/j.robot.2008.08.001. URL http://dl.acm.org/citation.cfm?id=1453261.1453481

[3] M. Asada, Y. Shirai, "Building a world model for a mobile robot using dynamic semantic constraints". In: *Proc. 11th International Joint Conference on Artificial Intelligence*, 1989, pp. 1629–1634.

[4] A. Nüchter, O. Wulf, K. Lingemann, J. Hertzberg, B. Wagner, H. Surmann, "3d mapping with semantic knowledge". In: *RoboCup International Symposium*, 2005, pp. 335–346.

[5] O. Grau, "A scene analysis system for the generation of 3-d models". In: *NRC '97: Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, IEEE Computer Society, Washington, DC, USA, 1997, p. 221.

[6] A. Nüchter, H. Surmann, K. Lingemann, J. Hertzberg, "Semantic scene analysis of scanned 3d indoor environments". In: *Proceedings of the 8th International Fall Workshop on Vision, Modeling, and Visualization (VMV 03)*, 2003, pp. 665–673.

[7] H. Cantzler, R. B. Fisher, M. Devy, "Quality enhancement of reconstructed 3d models using coplanarity and constraints". In: *Proceedings of the 24th DAGM Symposium on Pattern Recognition*, Springer-Verlag, London 2002, pp. 34–41. URL http://dl.acm.org/citation.cfm?id=648287.756379

[8] M. A. Fischler, R. Bolles, "Random sample consensus. A paradigm for model fitting with applications to image analysis and automated cartography". In: *Proc. 1980 Image Understanding Workshop (College Park, Md., Apr 1980), L. S. Baumann (ed.), Science Applications, Arlington, Va.*, 1980, pp. 71–88.

[9] M. Eich, M. Dabrowska, F. Kirchner, "Semantic labeling: Classification of 3d entities based on spatial

feature descriptors". In: *IEEE International Conference on Robotics and Automation (ICRA2010)*, Anchorage, Alaska, May 3, 2010.

[10] N. Vaskevicius, A. Birk, K. Pathak, J. Poppinga, "Fast detection of polygons in 3d point clouds from noise-prone range sensors". In: *IEEE International Workshop on Safety, Security and Rescue Robotics, SSRR*, IEEE, Rome, 2007, pp. 1–6.

[11] R. B. Rusu, Z. C. Marton, N. Blodow, M. Beetz, "Learning informative point classes for the acquisition of object model maps", In: *Proc. 10th International Conference on Control, Automation, Robotics and Vision ICARCV*, 2008, pp. 643–650.

[12] R. B. Rusu, N. Blodow, M. Beetz, "Fast point feature histograms (fpfh) for 3d registration". In: *Proc. IEEE International Conference on Robotics and Automation ICRA09*, 2009, pp. 3212–3217.