An Automated Video Surveillance System Using Viewpoint Feature Histogram and CUDA-enabled GPUs

Saurabh Jha
School of Computing Sciences and Engineering
VIT University
Chennai, India
saurabh.jha2010@vit.ac.in

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

***Abstract*— This paper presents an automated video surveillance system which deals with content monitoring and activity change in the environment. We use Viewpoint Feature Histogram, an image descriptor for object recognition and pose estimation for purpose of monitoring in the surveillance system. In order to enhance the performance of the system, we exploit the GPU architecture to perform data intensive task of surveillance system and implement it on CUDA-enabled devices. The experimental evaluation on the static data sets and live scenes captured from Microsoft Kinect show that Viewpoint Feature Histogram can be successfully used as an image descriptor in surveillance systems. We also test the performance of the Viewpoint Feature Histogram generation for different data sets on GPU and CPU to conclude that GPU clearly outperforms CPU for larger datasets.**

Keywords—Video surveillance; viewpoint feature histogram;computer vison; CUDA;Microsoft Kinect Application

I. INTRODUCTION

Automated Video surveillance is one of fastest growing industries along with the growth of technological advances in image and video processing, whereby forming an important research area. With the onset of the cheap depth sensing cameras, surveillance cameras are being used in wide range of applications. There is a pressing need to monitor through surveillance cameras in order to detect events and persons-of-interest. The key aspect lies in the fact that human monitoring requires a large number of personnel, leading to high ongoing costs and susceptible reliability due to the diminishing utility of humans while performing such mundane task s. Therefore the advances in technology have led to robot based surveillance systems to monitor all video feeds. In the surveillance of the Art Galleries, it is enough to detect the change in the static scene through which we can raise the alarm to convey the breach of security. Such problem requires very high speed computations which can be achieved by utilizing the GPU architecture.

Robert Collins et al also proposed a VSAM [1] which works through cooperative video sensors to provide continuous coverage of people and vehicles in a cluttered environment. There are been a plethora of work going on visual surveillance through though IP and unique Ethernet based hybrid video surveillance system. Datong Chen et al [2] propose an AI approach to recognize aggressive behaviors from video records using local binary motion descriptors. A recent work by Lozano and Otsuka [3] proposed a particle filtering based method for 3D tracking of faces using GPUs.

In various image processing applications, image description forms a key aspect of defining the key point features of the image. Among the few widely applied 3D point feature extraction approaches include: spherical harmonic invariants [4], spin images [5] and Point Feature Histograms (PFH) [6]. Rusu et al. in their work [6] gave a revised set of features to form Fast Point Feature Histograms (FPFH) that can be computed online and which have a computational complexity of O (k) (as opposed to O (k^2) for PFH.

In this paper, we design and develop a two phase automated surveillance system using Viewpoint Feature Histogram [7] and its implementation on Compute Unified Device Architecture (CUDA) [8], a C-based programming model from NVIDIA. We evaluate our proposed system on static images database of Point Cloud Library as well as using live frames captured from the depth sensing camera, Microsoft Kinect. It requires real-time object recognition and poses estimation of objects from the datasets. We utilize the Viewpoint Feature Histogram of Point Cloud Library [9], which records the relative angular directions of Surface Normal with respect to one another along with viewpoint, which forms an integral parameter in consideration to the security surveillance systems.

The rest of the paper is organized as follows. The next section describes the background and motivation of the proposed system and as to why there is pressing need to perform data intensive tasks using GPGPUs. It describes the Viewpoint Feature Histogram and its usage in the security system. Section III elaborates upon the proposed system design and architecture along with the algorithm. Section IV portrays the Experimental results and analysis. Section V mentions the conclusion and future works of the proposed system.

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. Numerous attempts have been made to utilize GPUs, not only for manipulating computer graphics, but also for different purposes, leading to a new research field, called "General-purpose computation on GPU (GPGPU)". The surveillance systems involves feature matching and pose estimation on the large data set in the monitoring phase which is computationally intensive, whereby GPUs are used to iteratively check upon the real-time generated Viewpoint Feature Histogram.

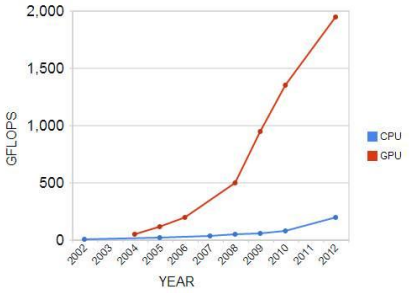

Figure 1: Evolving computational power of the CPU and the GPU in a billion floating point operations per second (GFLOPS).

The proposed security surveillance utilizes the capability of the GPUs by performing the data intensive tasks on it iteratively. The entire implementation is divided into two components where the first part involves the training of the data for the purpose of pose estimation of the sample objects and second part involves the monitoring phase which is iteratively performed on the data set for the purpose of the security surveillance system. The system takes advantage of the Viewpoint Feature Histogram's ability to accurately and robustly classify points with respect to their underlying surface, similar to Point Feature Histogram (PFH) [10]. Viewpoint Feature Histogram was built over the Fast Point Feature Histogram, which measures the angular features of the set of values between a data point <x, y, z> and its k nearest neighbor [10]. The Viewpoint Feature Histogram adds a viewpoint variance while retaining its invariance to scale. The reason why we use Viewpoint Feature Histogram is because it is a viewpoint which is calculated by collecting histograms of the angles that the viewpoint direction makes each normal at every point cloud data point,<x,y,z>. This angle is the relative angle with respect to the central viewpoint direction. Such a property forms a key parameter is determining a change in the orientation of the objects of the security system. If there exists any change in the central viewpoint, it'll be directly reflected in the Viewpoint Feature Histogram (see Figure 2), thereby enabling the system to detect the changes.

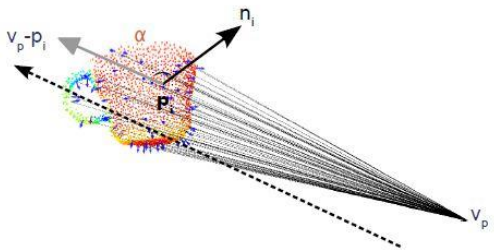

Figure 2**:** Viewpoint Feature Histogram showing the angle with respect to the central viewpoint (Courtesy Point Cloud Library)

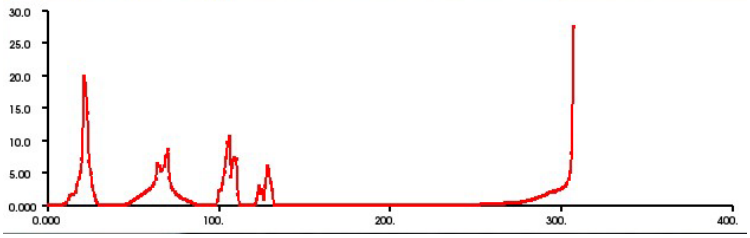

Figure 3: Viewpoint Feature Histogram for one of the objects used in the experimental system

III. SYSTEM DESIGN OVERVIEW

The proposed system works in two distinct components. The first phase is the training phase, which entails the object recognition and pose estimation model and the second phase is the monitoring phase in which the verification of the presence of the objects is performed and any changes to the environment are intimated. The proposed system depends on two important hardware components, Microsoft Kinect and NVIDIA's CUDA -enabled device.

Algorithm 1 defines the training phase of the system is trained for object recognition and pose estimation of different objects available in the 3D scene. The algorithm BuildFeatureDatabase () is used in training phase to train the system. Since Viewpoint Feature Histogram accommodates for statistics of the relative angles between the surface normal at each point to the surface normal at the centroid of the object, every object can be uniquely segmented along with its 6DOF pose estimation. The input data p_i is down sampled by the VoxelGrid filter, an implementation of Point Cloud Library [9] so that no noise in encountered while processing the data and for every processed scene $p_{i''}$, Viewpoint Feature Histogram V_i , is generated and stored in the global set V.

Algorithm 1: BuildFeatureDatabase( )

1. **foreach** object O
   a. **foreach** Pose p_i of object O
   b. Capture the raw point cloud data p_i of object O from Kinect device
   c. $p_{i'}$ = removeNaN(p_i)
   d. $p_{i''}$ = voxelGridFilter($p_{i'}$)

e. Add p_i to Global Dataset GS, GS = {GS U $p_{i''}$ }

f. Generate VFH V_i of $p_{i''}$

g. Add v_i to the set V, V = {V U v_i}

h. **Endfor**

Endfor

In monitoring phase, the system is given the task to monitor the environment for any changes in the environment and possible threats in the environment as required in any surveillance system. The system raised an alarm in case it detects any threats in environment and notifies the user of the system. Any change in the system is detected through the change in the viewpoint feature histogram. Viewpoint Feature Histogram (VFH) generation is performed on the CUDA enabled system for enhanced performance and save to the global set V. Every time a new VFH is generated, it is verified against the latest VFH of the scene to detect any changes. As noted in [2] that that object recognition and pose estimation capabilities using global dataset and VFH can be as accurate as 98.52%. Algorithm 2 elaborates over the monitoring algorithm of the surveillance system.

Algorithm 2: MonitorEnvironment( )

1. **while** (monitoring: true)
   a. Capture the raw point cloud data m_i of environment from kinect device
   b. $m_{i'}$ = removeNaN(m_i)
   c. $m_{i''}$ = voxelGridFilter($m_{i'}$)
   d. Segment $m_{i''}$ into distinct features F_i
   e. **foreach** F_i of m_i
   f. generate VFH, vm_i of F_i
   g. Alarm: 1
   h. **foreach** v_j in VS
   i. if v_j < = vm_i – 5 or v_j > = vm_i + 5
   j. Alarm: 0
   k. **if** Alarm: true
   l. Notify user of possible threat or Flag the object as missing
   m. **Endif**
   n. **Endfor**
   o. **Endfor**
2. **EndWhile**

IV. Experimental Evaulation

In order to test the veracity of the proposed security system, we perform tests over sample dataset gathered from the Microsoft Kinect in a simulated security environment and also with the static dataset [12] for VFH generation. We test the proposed system with 22 GB of memory on 33.5 EC2 Compute Units (2 x Intel Xeon X5570) and 2 x NVIDIA Tesla "Fermi" M2050 GPUs with 1690 GB of instance storage and 64 bit architecture.

Our analysis and experiments with the system shows that the difference between the two View Point Feature Histogram (VFH) signatures depends on the size of the object moved or removed from 3D scene keeping other environment features same.

Figure 4 and Figure 6 show four point clouds of the same static scene captured from Microsoft Kinect. In Figure 4.b, we removed only the small bag placed on chair as shown in Figure 4.a. In fig. 2.b, we removed only the pillow placed on chair along with the bag as shown in Fig 2.a. The difference in VFH which is calculated as sum over the difference between 308 bin values of two VFH signatures for Fig 1.a and Fig 1.b is 12.1855. The VFH difference between point clouds of Fig 2.a and 2.b is 24.4259. Also from our experiments, we note that the VFH difference of 5 or greater is significant difference in a security surveillance system. Therefore, when a difference of 5 or greater is encountered by the system an alarm is raised such as in these cases.

Figure 4 a Figure 4 b

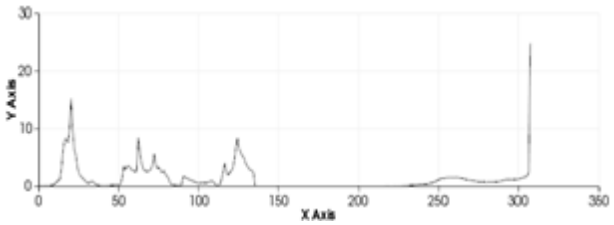

Figure 5 a

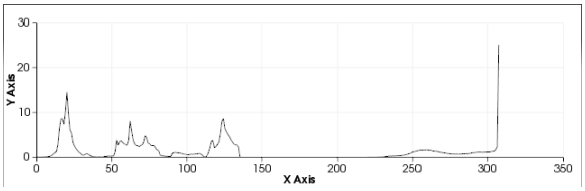

Figure 5 b

Figure 6 a

Figure 6 b

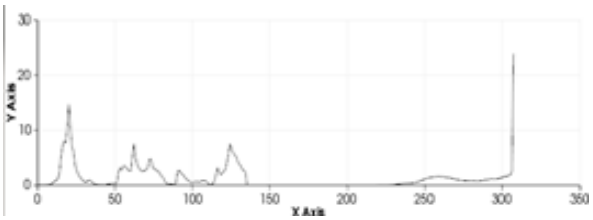

Figure 7 b

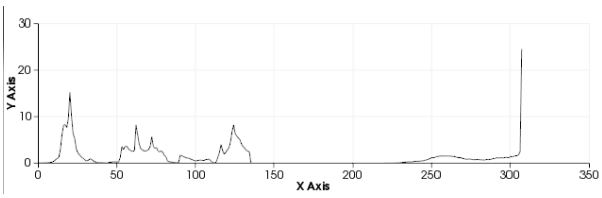

Figure 7 a

The proposed system for surveillance in Section III is heterogeneous in nature. However, the system can be executed exclusively on CPU or GPU eliminating the need for a heterogeneous system. In a closed circuit environment, where live feed is coming from three or more different 3D cameras, we prefer GPU over CPU to get the best performance with respect to time as shown in performance characteristics graph in Figure 8 and make our system near real time. It is clear from performance characteristic graph that GPU performs better than CPU for large datasets. Also, we note from our experiments that VFH generated on CPU and GPU are exactly same as shown in Figure 8 and hence can be used interchangeably used as per the security system design but the execution speed differs for large data sets.

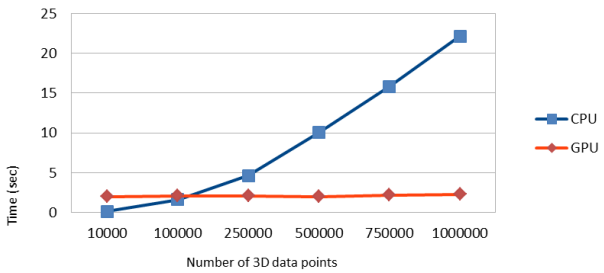

Figure 8

V. Conclusions and Future Work

In this paper we presented an automatic surveillance system using 3D image processing with Viewpoint Feature Histogram (VFH) as the key image descriptor on CUDA-enabled GPUs. The ability of VFH to detect changes in the scene captured through the depth sensing camera is experimentally evaluated on the live stream video feed as well as static data set. The experimental results show that there are significant differences in the VFH in the occurrence of the change in the scene of the environment. We can thus experimentally conclude that VFH is a robust image descriptor for object recognition and pose estimation and can be successfully used in video surveillance systems.

We also test the heterogeneity of the system and experimentally evaluated the time taken in the generation of VFH on CPU and GPU and can conclude that GPUs are highly useful for large datasets and clearly outperforms CPUs (see Figure 8).

We are currently working on making the system more robust by including other image descriptors such as spin images and contour maps which can be suitably adapted for the purpose of the surveillance systems. Also we would like to extend this to a distributed system to develop a distributed surveillance system.

References

[1] Robert T. Collins, Alan J. Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, Osamu Hasegawa, Peter Burt and Lambert Wixson, "A System for Video Surveillance and Monitoring", Robotics Institute, Carnegie Melon University.

[2] Datong Chen and Howard Wactlar and Ashok Bharucha and Ming-yu Chen and Can Gao and Alex Hauptmann, "*An AI Approach to Measuring Resident-on-Resident Physical Aggression In Nursing Homes* ", Association for the Advancement of Artificial Intelligence, 2008

[3] O. M. Lozano and K. Otsuka, "*Simultaneous and fast 3d tracking of multiple faces in video by gpu-based stream processing*," in Proc. Int. Conf. Acoustics, Speech and Signal Processing, 2008, pp. 713–716.

[4] G. Burel and H. H´enocq, "Three-dimensional invariants and their application to object recognition," Signal Process., vol. 45, no. 1, pp. 1–22, 1995.

[5] A. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," IEEE Transactions on Pattern Analysis and Machine Intelligence, May 1999.

[6] R. B. Rusu, N. Blodow, and M. Beetz, "*Fast Point Feature Histograms (FPFH) for 3D Registration,*" in ICRA, 2009.

[7] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, John Hsu, "Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram" in IROS2010

[8] NVIDIA, "Nvidia compute unified device architecture," http://www.nvidia.com/object/cuda_home_new.html, 2013.

[9] "Point Cloud Library, is a standalone, large scale, open project for 2D/3D image and point cloud processing". http://pointclouds.org , 2013

[10] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in VISAPP, 2009.