# Parallel 3D Local Descriptor on NVIDIA GPU

Linjia Hu  Saied Noshabadi
Department of Computer Science and Center of Computer System Research (CCSR)
Michigan Technological University, MI, USA

## Motivation

The compact local feature descriptor of 3D object is a key in Surface matching which is core techniques for 3D object recognition in computer vision. Signature of histogram of orientations (SHOT) is a novel 3D object local descriptor that achieves a good balance between descriptiveness and robustness. However, the computation workload of this descriptor is much higher than the other 3D local descriptors. Fortunately, graphics processing unit (GPU) is finding its way beyond graphics processing into general purpose computing. This p investigates the suitability of GPU for high density and large scale 3D object local descriptors computing. We employ GPU to accelerate a local descriptor SHOT, and show that efficient parallelization of this descriptor on GPU can speedup it by a factor of 5 to 40.

## Mathematical Model of SHOT

The strength of SHOT descriptor is based on two features. First, it conceives on a repeatable robust local reference frame, is unique and unambiguous and demonstrates better descriptive power and robustness. The details of computing of repeatable local reference frame can be found in [1][2]. Second, it combines the merits of signature and histogram categories of descriptors to create a more effective descriptor. For each key point, the SHOT technique uses an isotropic spherical grid partitioned along the radial, azimuth and elevation axes, as illustrated in the Figure 1. Experimentations show that 32 is a proper number of the spatial volumes, resulting in eight azimuth, two radial and two elevation divisions. Each segment within sphere in Figure 1 encodes a descriptive entity represented by its local histogram.

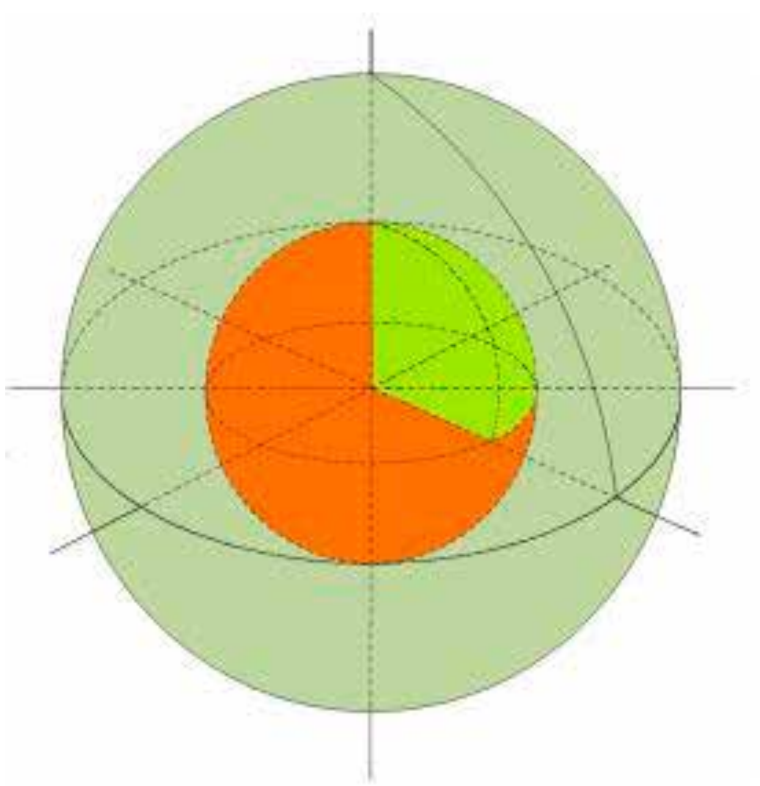

Figure 1

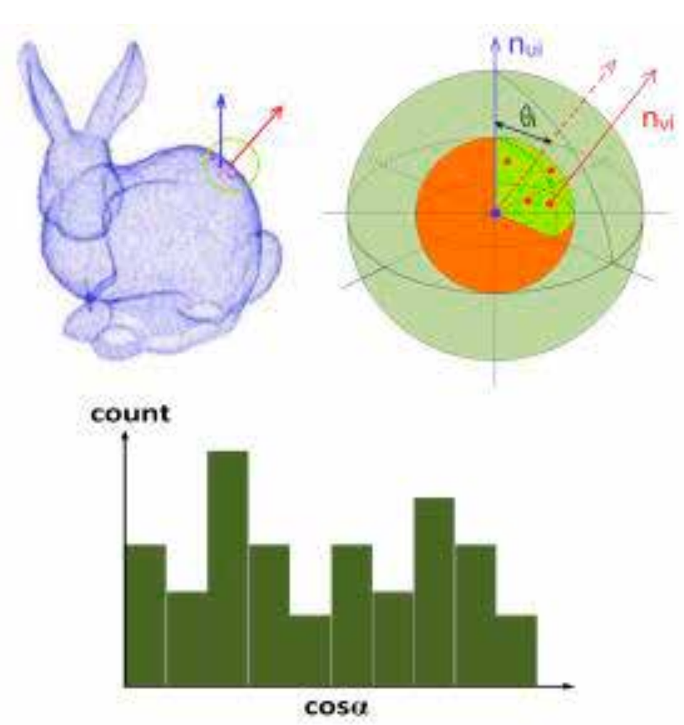


Figure 2

The formation of the local histogram is shown in Figure 2. One key point in the point cloud of Stanford Bunny is selected and one neighbor sphere around it is drew. Here, we take one segment marked in light green in the sphere into account. For the local histograms of this segment, we accumulate points counts into bins according to function $\cos\theta_i$, with $\theta_i$, the angle between the normal at each point $p_i$ within the spherical grid segment ($n_{vi}$), and the normal at the key point ($n_{ui}$).

## Parallel Implementation on GPU

The computing of SHOT descriptor consists of two key parts: one to calculate the unique LRF for each key point, and the other to compute the descriptor histogram. The data profiling results of SHOT descriptor on the CPU show that these two algorithm parts consume more than 90% of the total computing time. Since the computing of all key points are independent to each other, so both time consuming parts can be performed in parallel on GPU. The process of parallel key point descriptors computing are carried out in six phases including two CUDA kernels, two memory copings and two host computing as show in Algorithm 1.

**Algorithm 1** The GPU Accelerated SHOT Algorithm

```
Input:   Key points and surface points Coordinates
         and Neighbour Points of Each Key-point
Output: SHOT descriptor histogram
procedure G-SHOT
    [Host] Compute the Nearest Neighbour Point (NNP)
        for each key-point, preprocessing the data.
    [Host→Device] Copy surface points and
        coordinates, distances and coordinates
        information of NNP from host to device.
    [Kernel I] This kernel is to compute the Local
        Reference Frame (LRF) for all the key points.
    [Host] Remove irregular key points.
    [Kernel II] This kernel is to compute the descriptor
        for all the key points.
    [Device→Host] Copy the all the key-points' SHOT
        descriptors to host.
end procedure
```

Before the launch of two kernels, for the computation of LRF (Kernel I) and histogram (Kernel II) on GPU, we need to copy point data from the host (CPU) to device (GPU). After the completion of the local descriptor computation, the results are transferred from the device to host. The computing of histogram is dependent on the calculating of LRF, so we design two separate kernels to handle the corresponding computing. Prior to lunching the second kernel, the irregular key points must be removed. Two separate kernel launching here indeed implements coarse level threads synchronization.

## Experiment Results

Our test platform uses a 3.20 GHz Intel Core i7 quad-core CPU, a GeForce GTX 570 graphic card with 2.5 GB video memory, CUDA 4.0 and the Ubuntu 12 operating system. The following table shows the timing of serial CPU SHOT and parallel GPU SHOT, as well as speedup (the ratio of CPU serial SHOT descriptor computation time to that of the parallel GPU accelerated SHOT descriptor computing time) for some selected 3D point cloud models and scenes. Most of those models and scenes are broadly used computer graphic and computer vision research areas. All of them can be download from the PCL official website. The speedup of all these models and scenes ranges from 4 to 40. For the largest surface point and key point set in our experiments, the speedup can reach up to factor of 40. The main purpose of this comparation is to demonstrate that G-SHOT algorithm can achieve great speedup performance for the common models and scenes, instead of comparing the speedup of individual model or scene.

| *Data Sets** | $N_{Surf}$ | $N_{key}$ | $T_{cpu}$ $(s)$ | $T_{gpu}$ $(s)$ | *Speed up* |
|---|---|---|---|---|---|
| *Model* 1 | 13704 | 13704 | 1.05 | 0.24 | 4.38 |
| *Model* 2 | 18815 | 18715 | 3.33 | 0.37 | 9.00 |
| *Model* 3 | 204800 | 40251 | 6.51 | 0.83 | 7.84 |
| *Model* 4 | 135142 | 135142 | 12.42 | 1.09 | 11.39 |
| *Model* 5 | 313260 | 121550 | 15.75 | 1.06 | 14.86 |
| *Model* 6 | 614560 | 487951 | 97.27 | 2.36 | 40.70 |
| *Scene* 1 | 145511 | 145505 | 13.67 | 1.23 | 11.12 |
| *Scene* 2 | 307200 | 66053 | 12.53 | 0.91 | 13.76 |
| *Scene* 3 | 307198 | 241407 | 41.46 | 1.71 | 24.25 |

*Models from the first to the sixth are **Milk Box Model**, **Office Chair Model**, **Stanford Bunny Model**, **Stanford Dragon Model** and **Happy Buddha Model reactively**; and scenes from the first to the third are **Office Scene**, **Table Scene** and **Five People Scene**.

## Reference

[1] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in 11th European Conference on Computer Vision (ECCV), Hersonissos, Greece, September 5-11 2010.

[2] F. Tombari, S. Salti, and L. Di Stefano, "A combined texture-shape descriptor for enhanced 3d feature matching," in IEEE International Conference on Image Processing (ICIP), Brussels, Belgium, September 11-14, 2011.