

# FAST VISUAL HULL AND STEREO MATCHING ON CUDA

Mykyta Fastovets, Jean-Yves Guillemaut, Adrian Hilton

CVSSP, University of Surrey, UK {mykyta.fastovets, j.guillemaut, a.hilton}@surrey.ac.uk

## Abstract

Stereo matching and visual hull are techniques that are often used in 3D reconstruction. This paper presents and evaluates implementations of these algorithms on the GPU using the CUDA architecture. Experimental results show that both, visual hull and stereo matching, have much to gain in terms of speed from the data parallel execution model.

**Keywords:** CUDA, GPGPU, stereo matching, visual hull.

## 1 Introduction

Current computational hardware development trends show that newer generations of hardware tend to focus on using multiple processor cores on a chip, rather than improving the performance of a single core. This trend is likely to continue and processors with hundreds of cores are already in existence (GPUs)[1]. Traditional implementations of vision algorithms tend to be created and optimised to run on a single core, which is no longer a satisfactory approach. This paper explores the adaptability of the visual hull and stereo score computation algorithms to the data parallel programming model and execution on manycore GPUs using the CUDA architecture.

## 2 Visual Hull and Stereo Matching on CUDA

The data parallel programming model assumes that a task can be segmented into many similar components (threads) that can be executed in parallel on the GPU.

### 2.1 Visual Hull

In the case of the visual hull reconstruction from  $n$  images, a thread can be defined as the set of operations required for determining whether the binary voxel  $v_{i,j,k} \in V$  is within the object, where  $V$  designates the voxel space. Thus,

$$v_{i,j,k} = I(P_0\nu_{i,j,k}) \cap I(P_1\nu_{i,j,k}) \dots \cap I(P_n\nu_{i,j,k}) \quad (1)$$

where  $P_n$  is the projection matrix associated with the  $n^{th}$  image,  $I(p)$  is the intensity value at pixel  $p$ , and  $\nu$  gives the real world coordinates of the voxel  $v$ .

### 2.2 Stereo Matching

Stereo score computation methods implemented were the Normalised Cross Correlation and the Sum of Square Difference. These methods are formally defined in [2]. A thread is defined as the set of operations required to produce

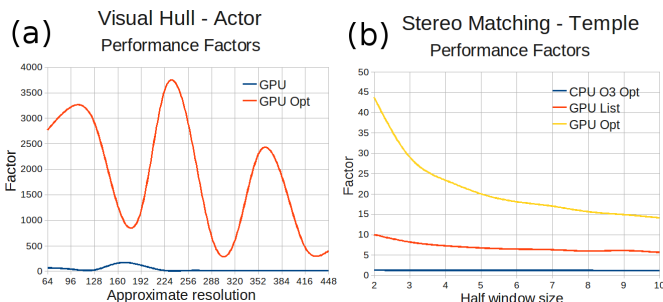


Figure 1: Performance comparison with CPU algorithms. (a) Visual hull; (b) Stereo matching

a score for a pair of pixels  $p_l$  and  $p_r$  on the left and the right images respectively. The result produced by this algorithm is a 3D matrix containing a score for each pair of pixels. An alternative implementation of both algorithms that uses foreground information to limit the search space was also tested.

## 3 Results and Conclusions

Experiments were conducted on two data sets, one generated at CVSSP and the other being the Middlebury Temple image set. Figure 1(a) shows the improvement factor in terms of speed of the GPU implementations over a CPU one, for the visual hull algorithm. *GPU* represents a naive implementation, while *GPU Opt* exploits more advanced capabilities of the architecture. These results suggest that visual hull is a perfect candidate for parallel computation, with the algorithm completing execution from 480 to 3700 times faster than on the CPU, putting it well within the real time range. The results for stereo matching shown in Figure 1(b), while not as impressive, also show considerable improvement, especially for smaller window sizes. Stereo score computations benefit from considerable speed up when taking advantage of parallelism provided by the CUDA architecture. These implementations can be further optimised. Future work will focus on the development of real time implementations of more complex algorithms and techniques.

## References

- [1] NVIDIA Corporation. *NVIDIA CUDA Programming Guide version 2.3*, July 2009.
- [2] Emanuele Trucco and Alessandro Verri. *Introductory techniques for 3-d computer vision*. 1998.