



# SPEEDUP IMAGE SPATIO TEMPORAL FEATURE EXTRACTION USING GPGPU

By

**Ahmed Mahmoud Ahmed Mehrez**

A Thesis Submitted to the

Faculty of Engineering at Cairo University

In Partial Fulfillment of the

Requirements for the Degree of

**MASTER OF SCIENCE**

**In**

**COMPUTER ENGINEERING**

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2018

# Speedup Image Spatio Temporal Feature Extraction Using GPGPU

By

**Ahmed Mahmoud Ahmed Mehrez**

A Thesis Submitted to the

Faculty of Engineering at Cairo University

In Partial Fulfillment of the

Requirements for the Degree of

**MASTER OF SCIENCE**

**In**

**COMPUTER ENGINEERING**

**Under the Supervision of**

**Prof. Dr. Elsayed E. Hemayed**
Department of Computer Engineering
Faculty of Engineering, Cairo University

**Dr. Ahmed Abdel-Fattah Morgan**
Department of Computer Engineering
Faculty of Engineering, Cairo University

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
2018

# Speedup Image Spatio Temporal Feature Extraction Using GPGPU

By
**Ahmed Mahmoud Ahmed Mehrez**

A Thesis Submitted to the

Faculty of Engineering at Cairo University

In Partial Fulfillment of the

Requirements for the Degree of

**MASTER OF SCIENCE**

**In**

**COMPUTER ENGINEERING**

Approved by the Examining Committee

| | |
|---|---|
| **Prof. Elsayed E. Hemayed** | Thesis main Advisor |
| **Prof. Magda B. Fayek** | Internal examiner |
| **Prof. Reda Abdelwahab Ahmed** | External examiner |

- Vice Dean for educational and student affairs, Faculty of Computer and Information -Cairo University

FACULTY OF ENGINEERING CAIRO UNIVERSITY
GIZA, EGYPT
2018

**Engineer:** **Ahmed Mahmoud Ahmed Mehrez**
**Date of Birth :** **02 / 02 / 1988**
**Nationality :** **Egyptian**
**E-mail :** **ahmed.mehrez@feng.bu.edu.eg**
**Phone. :** **01111546547**
**Address :** **4 Hefni Nasef st. – Qobba Gardins**

**Registration Date :** **01 / 10 / 2011**
**Awarding Date :** **/ / 2018**
**Degree :** **Master of Science**
**Department :** **Computer Engineering**

**Supervisors :** **Prof. Dr. Elsayed E. Hemayed**
**Dr. Ahmed Abdel-Fatah Morgan**

**Examiners :**

**Prof. Dr. Elsayed E. Hemayed** (Thesis main advisor)
**Prof. Dr. Magda B. Fayek** (Internal examiner)
**Prof. Dr. Reda Abdelwahab Ahmed** (External examiner)
- Vice Dean for educational and student affairs, Faculty of Computer and Information -Cairo University

**Title of Thesis:**

## Speedup Image Spatio Temporal Feature Extraction Using GPGPU

**Key Words: (GPU - Image features - Parallel processing - Image matching)**

**Summary:**

The robust representation of image features becomes fundamental to most machine vision and image registration applications. Spatio-temporal feature extraction algorithms are favored because of their robust generated features. However, they have high computational complexity. In this thesis, we propose new parallel implementations, using GPU computing, for the two most widely used Spatio-temporal feature extraction algorithms: Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Feature (SURF).
In our implementations, we solve problems with previous parallel implementations, such as load imbalance, thread synchronization, and the use of atomic operations. We compare our presented implementations to previous CPU and GPU parallel implementations of the two algorithms. Results used in Human action recognition and achieve accuracy 96% for SIFT and 94.5% for SURF.

# ACKNOWLEDGEMENT

All praises to almighty unique Allah, who blessed me with the ability to undertake and finally complete this thesis and who brought me peace and happiness.

I am highly grateful to my supervisors, Prof. Dr. **Elsayed E. Hemayed** and **Dr. Ahmed Morgan**, for their continuous support, excellent guidance, and their advice to complete this thesis successfully.

I would like to thank Prof. **Magda Fayek** for her helpful comments, and support.

I would also like to thank Prof. **Reda Abdelwahab**, Vice Dean for educational and student affairs, Faculty of Computer and Information -Cairo University, for his helpful comments, and support.

# Dedication

***In the Name of the Unique God "Allah", the Beneficent, the Merciful***

# Table of Contents

# INDEX OF TABLES

# INDEX OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| **FPS** | **Frames Per Second** |
| **CPU** | **Central Processing Unit** |
| **GPU** | **Graphical Processing Unit** |
| **SIFT** | **Scale Invariant Feature Transform** |
| **SURF** | **Speeded-Up Robust Features** |
| **CUDA** | **Compute Unified Device Architecture** |
| **SIMD** | **Single Instruction Multiple Data** |
| **DOG** | **Difference Of Gaussians** |
| **SM** | **Streaming Multiprocessor** |
| **SP** | **Streaming Processor** |
| **DMA** | **Direct Memory Access** |
| **API** | **Application Programming Interface** |
| **CUFFT** | **CUDA Fast Fourier Transform library** |
| **CUBLAS** | **CUDA Basic Linear Algebra Subroutine** |
| **SVM** | **Support vector machine** |
| **SD** | **Standard Deviation** |
| **GPC** | **Graphics processor controller**XzCC |
| **DRAM** | **Dynamic RAM** |
| **HOG** | **Histogram of oriented gradients** |

# ABSTRACT

The robust representation of image features becomes fundamental to most machine vision and image registration applications. Local approaches, which do not require image segmentation, are robust to changes, become more and more used. Local approaches consist of two parts, detector that locates features and descriptors, which describe these features. Spatio-temporal feature extraction algorithms are widely used in many image processing and computer vision applications. They are favored because of their robust generated features. However, they have high computational complexity. Parallelizing these algorithms, in order to speed their execution up, is of great importance. In this thesis, we propose new parallel implementations, using GPU computing, for the two most widely used Spatio-temporal feature extraction algorithms: Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Feature (SURF). In our implementations, we solve problems with previous parallel implementations, such as load imbalance, thread synchronization, and the use of atomic operations. Our implementations speed up the execution by simultaneously processing all the work of each stage of the two algorithms, without dividing that stage into smaller sequential ones. The allocation of the threads in our implementations further allows them to increase the occupancy of the GPU Streaming Multiprocessors (SM's). We compare our presented implementations to previous CPU and GPU parallel implementations of the two algorithms.

Results show that the proposed implementations could do all the processing in real time with high accuracy. They further achieve higher speedup, frame rate, and SM occupancy than previous best-known parallel implementations of the two algorithms. It achieves 18X for SURF and 27X for SIFT. Processing can be done in real time and give good accuracy. Proposed solution achieved 229 frame per second for SURF and 134 frame per second for SIFT for the 320*240 images. Results used in Human action recognition and achieve accuracy 96% for SIFT and 94.5% for SURF.

# Chapter 1
# INTRODUCTION

Most computer vision tasks require a great deal of mathematical computation. For many computer vision algorithms, the analysis of a single image can take anywhere from a few seconds to several hours to process. In short, computer vision algorithms require a large number of computations as well as an equally large number of memory values. Computer vision algorithms are applied to broad range of environments from home entertainment systems to the operation of unmanned aerial and ground vehicles. Each new generation of application increases the need for more computational resources. Traditionally software developers and even scientists have strictly relied on the increase of processor clock frequency as the primary method of gaining performance for the next generation of applied algorithms. The robust representation of image features is fundamental to most machine vision and image registration applications.

Local approaches, which do not require image segmentation, are proved to be robust to changes that may occur in images, such as the change in the illumination and the view angle [1]. According to the changes in the image environment, local feature extraction approaches should overcome two main challenges. The first challenge is how to locate regions of the image that have different features and how to detect these features. The second challenge is how to describe the detected features in a unique way, which could be used to find a match with similar features in other images. Spatio-temporal features are shown to represent robust ones against the many variations in the image environment [2]. Accordingly, they become widely used in most image processing and computer vision applications.

Scale Invariant Feature Transform (SIFT) [1] and Speeded-Up Robust Features (SURF) [3]are among the most robust Spatio-temporal local feature extraction algorithms that are used in many computer vision techniques. For example, they are used in object recognition and tracking, image classification, face authentication, and video event classification. They could overcome illumination, scale, and rotation variations. SIFT and SURF extract features in the form of interest points, which represent special points in the image that could be used in image matching. A feature vector is created for each point, which describes the gradients in the region around that point.

The SIFT algorithm uses Gaussian filters with increasing □. As the filter size increases, the required computations and the execution time of the algorithm increase as well. This long computational time prevents the sequential implementation of the SIFT algorithm from being used in real-time applications. Alternatively, in order to use the SIFT algorithm in these real-time applications, some implementations use smaller Gaussian filters[4]. This unfortunately results in poor quality interest points. On another hand, the SURF algorithm is proposed to overcome the computational time problem of SIFT. The feature detection of the SURF algorithm is based on Hessian matrix and box filters approximation. Therefore, the SURF algorithm has fixed computational time throughout all points. In turn, the computation time of the SURF algorithm allows it to detect and describe interest points faster than SIFT. However, its sequential implementation is still far from being efficiently used in real-time applications.

The SIFT and SURF algorithms have been used for content based image retrieval [5-9], video event classification [10, 11], object recognition [1, 12, 13], object tracking [14, 15], image classification[16-19], building panoramas [20, 21] mobile surveillance [22, 23], and face authentication[24] , among other applications. It stands out among local feature descriptors for its invariance to scale, rotation and linear illumination, and its partial invariance to 3D viewpoint change[25] . The widespread use of SIFT may be attributed to both its success at localizing invariant interest points in position and scale, using the difference-of-Gaussians detector, and the distinctiveness of its 128-element keypoint descriptor, which is derived from a 16 x 16 pixel gradient patch centered on the image feature location.

In order to enable the use of SIFT and SURF algorithms in real-time applications, many parallel implementations of the two algorithms are presented using different hardware architectures [26-30]. These implementations managed to provide a good speedup with respect to sequential implementations of the two algorithms. Nevertheless, they still have some problems that should be handled, in order to better enhance the performance of the two algorithms. First, most of previous parallel implementations suffer from a load imbalance problem. In each stage of the two algorithms, the computations are distributed in an imbalanced fashion over the Processing Elements (PEs) of the employed hardware architecture. Consequently, the heavily loaded PEs would need more time to finish its work than the less loaded ones. Indeed, a more balanced distribution of the work would result in a higher speedup. Second, most of previous parallel implementations suffer from a thread synchronization problem. Computations of different stages of the two algorithms are split into smaller segments, which could be processed in parallel. However, each segment has to wait until the previous one finishes. With the aforementioned load imbalance problem, many PEs are left idle and the hardware occupancy deteriorates. This, in turn, prevents previous parallel implementations from achieving the maximum possible speedup. Finally, previous parallel implementations store interest points in a sequential manner, which depress the overall speedup that could be achieved by the parallel implementation. In this thesis, we present a new parallel implementation that targets these problems. In other words, our implementation better handles the load imbalance and the synchronization between threads. It also increases the GPU occupancy and stores the detected interest points in a more efficient parallel manner.

Graphical Processing Units (GPUs) are mainly used for graphics. In this thesis, we target NVIDIA GPUs; and hence, we would employ its terminology. A GPU consists of tens or hundreds of Streaming Processors (SPs) that are grouped to form Streaming Multiprocessors (SMs). Therefore, GPUs are cheap platforms that could result in a significant performance enhancement, if their parallelism is properly exploited. NVIDIA further develops CUDA, as a parallel computing platform, to facilitate the programming of their GPUs. This motivates many researchers to employ GPUs in speeding up general-purpose applications [31]. Interested readers would find dozens of such research work in [32].

In most cases, problems possessing data-level parallelism are best suited for GPU execution. Data parallelism focuses on distributing the large amounts of data across different parallel computing cores. A problem appears data-parallel if each core can perform the same identical task on different pieces of distributed data. There are distinct ranges of data-parallel problems. Small-scale image processing that includes the parallel manipulation or analysis of pixels can be achieved with multiprocessor extensions such as Single Instruction, Multiple Data (SIMD). Larger problems of data-parallel computing can be solved with large scale distributed systems consisting of