

**VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF
TECHNOLOGY**

(An Autonomous Institute Affiliated to University of Mumbai)

Department of Computer Engineering



Project Report on

Applications of CUDA Programming

Submitted in partial fulfillment of the requirements of the
degree

**BACHELOR OF ENGINEERING IN COMPUTER
ENGINEERING**

By

Vishakha Mangtani D12B/30

Ruchir Jain D12B/19

Ketan Paryani D12B/40

Project Mentor

Prof. Mrs. Sunita Suralkar

**University of Mumbai
(AY 2023-24)**

**VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF
TECHNOLOGY**

(An Autonomous Institute Affiliated to University of Mumbai)

Department of Computer Engineering



CERTIFICATE

This is to certify that the Mini Project entitled “**Applications of CUDA Programming**” is a bonafide work of **Vishakha Mangtani (30), Ruchir Jain (19), Ketan Paryani (40)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of “**Bachelor of Engineering**” in “**Computer Engineering**” .

(Prof. Mrs Sunita Suralkar)

Mentor

(Prof. Mrs. Nupur Giri)

Head of Department

(Mrs. J M Nair)

Principal

MINI PROJECT APPROVAL

This Mini Project entitled “**Applications of CUDA Programming**” by
Vishakha Mangtani (30), Ruchir Jain (19), Ketan Paryani (40) is approved
for the degree of **Bachelor of Engineering in Computer Engineering**.

Examiners

1.....

(Internal Examiner Name & Sign)

2.....

(External Examiner name & Sign)

Date:

Place:

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Vishakha Mangtani (30)

Ruchir Jain (19)

Ketan Paryani (40)

Date:

ACKNOWLEDGEMENT

We are thankful to our college Vivekanand Education Society's Institute of Technology for considering our project and extending help at all stages needed during our work of collecting information regarding the project.

It gives us immense pleasure to express our deep and sincere gratitude to Assistant Professor **Mrs. Sunita Suralkar** (Project Guide) for her kind help and valuable advice during the development of project synopsis and for her guidance and suggestions.

We are deeply indebted to Head of the Computer Department **Dr.(Mrs.) Nupur Giri** and our Principal **Dr. (Mrs.) J.M. Nair** , for giving us this valuable opportunity to do this project.

We express our hearty thanks to them for their assistance without which it would have been difficult in finishing this project synopsis and project review successfully.

We convey our deep sense of gratitude to all teaching and non-teaching staff for their constant encouragement, support and selfless help throughout the project work. It is a great pleasure to acknowledge the help and suggestion, which we received from the Department of Computer Engineering.

We wish to express our profound thanks to all those who helped us in gathering information about the project. Our families too have provided moral support and encouragement several times.

INDEX

Chapter No.	Title	Page No.
	Abstract	8
	List of figures	9
1.	Introduction 1.1 Introduction 1.2 Motivation 1.3 Problem Definition 1.4 Existing Systems 1.5 Lacuna of the existing systems 1.6 Relevance of the Project	10
2.	Literature Survey 2.1. Brief Overview of Literature Survey	15
3.	Chapter 3: Requirement Gathering for the Proposed System 3.1 Introduction to requirement gathering 3.2 Functional Requirements 3.3 Non-Functional Requirements 3.4. Hardware, Software , Technology and tools utilized 3.5 Constraints	17
4.	Proposed Design 4.1 Block diagram of the system 4.2 Modular design of the system 4.3 Detailed Design 4.4 Project Scheduling & Tracking using Timeline / Gantt Chart	23

5.	Implementation of the Proposed System 5.1. Methodology employed for development 5.2 Algorithms and flowcharts for the respective modules developed 5.3 Datasets source and utilization	25
6.	Chapter 6: Results and Discussion 6.1. Screenshots of User Interface (UI) for the respective module 6.2. Performance Evaluation measures 6.3. Input Parameters / Features considered 6.4. Graphical and statistical output 6.5. Comparison of results with existing systems	30
7.	Chapter 7: Conclusion 7.1 Limitations 7.2 Conclusion 7.3 Future Scope	33
8.	References	36
9.	Appendix	38

ABSTRACT

The Compact Muon Solenoid (CMS), a versatile detector at the Large Hadron Collider (LHC), captures particle collisions, generating numerous event images for subsequent analysis. These collisions leave distinctive color trails crucial for particle identification, traditionally analyzed manually. To streamline this process, we propose employing advanced neural network techniques like GCN and data handling tools such as ROOT, trained on collision images provided by CERN. By doing so, we aim to make particle identification more efficient and less reliant on manual intervention. Meanwhile, CUDA (Compute Unified Device Architecture), pioneered by NVIDIA, facilitates parallel computing on GPUs, enabling developers to accelerate compute-intensive tasks effectively. This parallel execution capability enhances the performance of various applications, from scientific simulations to deep learning models, by harnessing the massive parallelism inherent in GPUs.

LIST OF FIGURES

Figure Number	Figure Name
Fig. 1. 1.	Hadron Collider
Fig. 4. 1.	Block diagram of the system
Fig. 4. 2.	Modular design of the system
Fig. 4. 3.	Detailed Design
Fig. 4. 4.	Project Scheduling & Tracking using Timeline / Gantt Chart
Fig. 5. 1.	Compute distance flowchart
Fig. 5. 2.	Feature matrix flowchart
Fig. 5. 3.	Sample Dataset
Fig. 6. 1.	CPU Execution
Fig. 6. 2.	GPU Execution
Fig. 6. 3.	Graphical and statistical output
Fig. 6. 4.	Comparison of results with existing systems
Fig. 9. 1.	Project review sheet

1. INTRODUCTION

The introduction section provides brief information about the project. It also includes the motivation behind the project and the drawbacks behind the existing system. A detailed problem definition description is provided which discusses the problem statement in detail. Followed by there is a subsection on relevance of the project. Finally, last but not least, is the subsection on Methodology used.

1.1 Introduction

The CMS experiment at CERN has a broad physics program ranging from studying the Standard Model to searching for extra dimensions and particles that could make up dark matter. In the CMS experiment at CERN, Geneva, a large number of HGCal sensor modules were fabricated in advanced laboratories around the world. Various particles and their natures are detected inside the CMS large hadron collider. The Large Hadron Collider (LHC) is the world's most powerful and largest particle accelerator. The LHC consists of a 27-kilometer ring of superconducting magnets with a number of accelerating structures to boost the energy of the particles along the way. It accelerates protons to nearly the velocity of light and then collides them at four locations around its ring. To face the new challenges of the LHC, Physicists at the CMS collaboration are working on a completely new calorimeter, the High-Granularity Calorimeter which will deliver 10 times more collisions to its experiments to be installed in the endcaps of the detector. Each particle that emerges from an LHC collision is like a piece of a puzzle, with some of these pieces breaking up further as they travel away from the collision. Each leaves a trace in the detector and CMS's job is to gather up information about every one - perhaps 20, 100 or even 1000 puzzle tracks - so that physicists can put the jigsaw back together and see the full picture of what happened at the heart of the collision. The inner tracking system of CMS is designed to provide a precise and efficient measurement of the trajectories of charged particles emerging from the LHC collisions, as well as a precise reconstruction of secondary vertices. It surrounds the interaction point and has a length of 5.8m and a diameter of 2.5m. Deep learning has recently been effectively applied to image recognition. Identification of particles and their trails can be done from photos using a combination of deep learning and image processing technologies.

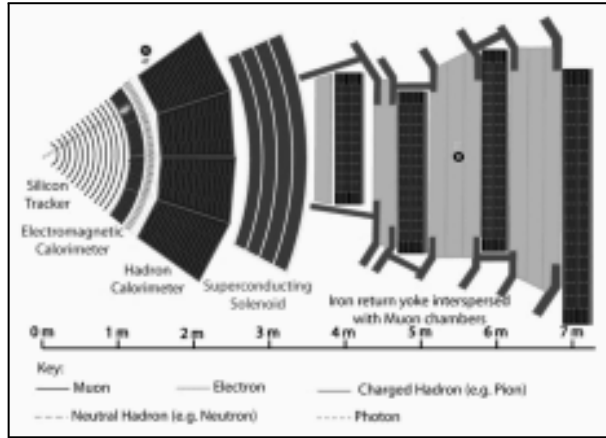


fig. 1.1 Hadron Collider

1.2 Motivation

The motivation behind our project lies in the critical need to effectively process the immense volume of data generated by the CMS experiment at CERN, Geneva. With approximately one billion proton-proton interactions occurring every second within the detector, traditional processing methods are insufficient for timely analysis. This necessitates the utilization of GPU-based preprocessing to expedite the identification of physical objects, including crucial events such as those potentially producing the Higgs particle. By implementing advanced techniques like deep learning and Graph Convolutional Networks (GCN), we aim to calculate particle energies with unparalleled accuracy. The significance of this endeavor is underscored by the importance of identifying relevant events amidst the vast sea of data, enabling subsequent in-depth analysis and potentially leading to groundbreaking discoveries in particle physics.

1.3 Problem Definition

The primary challenge at hand is the need to significantly reduce the processing time of data generated by the CMS experiment at CERN. With the vast amount of data being produced, traditional processing methods are inadequate for timely analysis. To address this challenge, our approach involves leveraging NVIDIA's GPU and CUDA programming capabilities. By harnessing the parallel processing power of GPUs, we aim to accelerate data preprocessing tasks, enabling the swift identification of physical objects within the CMS detector. This optimization is essential for implementing a reliable trigger system to select potentially significant events.

1.4 Existing Systems

After conducting a thorough market analysis, it has become evident that there is currently no existing system comparable to our project. While there are various software solutions available for similar purposes, none offer the comprehensive features and functionality that our project aims to provide. Existing systems may lack certain critical features, have limited scalability, or lack integration capabilities with other essential tools and platforms. Furthermore, many current solutions are tailored to specific industries or use cases, leaving a gap in the market for a versatile and adaptable software solution like ours. Therefore, our project presents a unique opportunity to address these shortcomings and offer a more holistic and innovative solution to meet the diverse needs of our target audience.

1.5 Lacuna of the existing systems

The image convolution algorithm encounters a significant computational bottleneck when executed on a CPU, primarily due to the sequential nature of CPU processing. Unlike GPUs, which excel at parallel processing tasks, CPUs lack the inherent ability to efficiently handle the massive parallelism required for convolution operations. Consequently, even relatively simple convolution tasks can strain CPU resources and result in slower processing speeds. This bottleneck inhibits the algorithm's ability to fully leverage the available computational power, leading to performance limitations and prolonged processing times.

In addition to the computational bottleneck, the CPU struggles to effectively utilize its resources for parallel processing, further exacerbating performance issues. While modern CPUs feature multiple cores capable of concurrent execution, the degree of parallelism achievable on a CPU is inherently limited compared to GPUs. As a result, the CPU may underutilize its cores during convolution operations, leading to inefficient resource allocation and suboptimal performance. This inefficiency hampers the algorithm's ability to achieve optimal throughput and significantly impacts overall processing efficiency.

Moreover, as image sizes or dataset complexity increase, the CPU-based convolution algorithm faces scalability challenges that impede its ability to handle larger workloads effectively. The sequential nature of CPU processing limits its ability to scale linearly with input size, resulting in diminishing returns as the computational demands grow. Consequently, as the computational workload increases, the CPU may struggle to keep pace,

leading to degraded performance and longer processing times. This scalability limitation restricts the algorithm's applicability to tasks requiring processing of large datasets or high-resolution images, where efficient scaling is crucial for timely completion.

Furthermore, the CPU-centric approach falls short in achieving real-time image convolution, particularly when dealing with high-resolution images or continuous video streams. The inherent limitations of CPU architecture, such as lower parallelism and slower processing speeds, hinder its ability to process data in real-time. As a result, the CPU may struggle to meet the stringent timing requirements imposed by real-time applications, leading to delays in processing and potentially compromising the application's functionality. This shortfall underscores the need for alternative processing solutions, such as GPU-based convolution, to achieve real-time performance in demanding applications.

1.6 Relevance of the Project

The relevance of our project lies in its capacity to revolutionize traditional donation schemes by leveraging modern technology to streamline and enhance the donation process. With the growing prevalence of digital platforms and the increasing need for efficient charitable giving mechanisms, our project fills a crucial gap in the philanthropic landscape. By providing a user-friendly and intuitive donation scheme app, we empower users to contribute to various causes effortlessly, thus fostering a culture of generosity and social responsibility. Moreover, the project's admin panel offers comprehensive management tools, enabling administrators to oversee donations, manage user accounts, and ensure transparency and accountability in the donation process.

Furthermore, our project's relevance extends beyond its immediate application in facilitating donations to encompass broader societal benefits. By facilitating easier access to donation opportunities and enabling individuals to contribute to causes they care about, the project promotes inclusivity and democratizes philanthropy. Additionally, the admin panel's functionalities empower organizations and nonprofits to efficiently manage donation campaigns, maximize donor engagement, and optimize resource allocation. Ultimately, the project's ability to streamline the donation process and enhance transparency enhances trust in charitable organizations and fosters stronger connections between donors and causes, thus contributing to the overall well-being of society.

In the context of technological advancements and evolving societal needs, our project's relevance is further underscored by its potential to adapt and evolve in response to changing circumstances. With the flexibility to integrate emerging technologies, scale to accommodate increasing demand, and customize features to suit specific organizational requirements, our project remains poised to remain relevant and impactful in the ever-changing landscape of charitable giving. By continuously iterating on our solution based on user feedback, technological advancements, and market trends, we ensure that our project remains at the forefront of innovation in the realm of philanthropy, driving positive social change and making a lasting impact on communities worldwide.

2. LITERATURE SURVEY

2.1. Brief Overview of Literature Survey

Sr.no.	Title of Paper	Author	Year	Summary
1.	J. Nickolls, "GPU parallel computing architecture and CUDA programming model," 2007 IEEE Hot Chips 19 Symposium (HCS), Stanford, CA, USA, 2007, pp. 1-12, doi: 10.1109/HOTCHIPS.2007.7482491	J. Nickolls	2007	The paper introduces CUDA, a programming model for utilizing GPU parallel computing power beyond graphics processing. It outlines the evolution of GPU architecture towards parallel computing and discusses the key features of CUDA, highlighting its potential for general-purpose computing tasks.
2.	Er.Paramjeet kaur and Er.Nishi, "A Survey on CUDA" / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 2210-2214	Er.Paramjeet kaur and Er.Nishi	2014	This paper provides an overview of CUDA (Compute Unified Device Architecture). It likely explores various aspects of CUDA technology, including its architecture, programming model, applications, and advancements up to the publication date.
3.	M. Ujaldon, "High performance computing and simulations on the GPU using CUDA," 2012 International Conference on	M. Ujaldon	2012	The paper "High Performance Computing and Simulations on the GPU using CUDA" by M. Ujaldon, presented at the 2012 HPCS Conference in Madrid,

	High Performance Computing & Simulation (HPCS), Madrid, Spain, 2012, pp. 1-7, doi: 10.1109/HPCSim.2012.6266884			delves into leveraging CUDA for high-performance computing on GPUs. It discusses CUDA's role in accelerating simulations and computing tasks efficiently.
4.	A.-M. Magnan, "HGCAL: a High-Granularity Calorimeter for the endcaps of CMS at HL-LHC ", 16 January 2017	A.-M. Magnan	2017	discusses the development and implementation of the High-Granularity Calorimeter (HGCAL) for the endcaps of the CMS (Compact Muon Solenoid) detector at the High-Luminosity Large Hadron Collider (HL-LHC). The HGCAL is designed to improve the performance of the CMS detector in terms of energy resolution
5.	Sharmistha, M. Amilkanthwar and S. Balachandran, "Augmentation of Programs with CUDA Streams," 2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications, Leganes, Spain, 2012, pp. 855-856, doi: 10.1109/ISPA.2012.132.	Sharmistha, M. Amilkanthwar and S. Balachandran	2012	This paper, published in the conference proceedings, discusses the incorporation of CUDA streams into programs. The utilization of CUDA streams allows for concurrent execution of multiple kernels on NVIDIA GPUs, thereby enhancing performance in parallel processing applications.

3. PROPOSED SYSTEM

3.1 Introduction to requirement gathering

In the context of our project aimed at detecting physical objects in the CMS experiment at CERN, the process of requirement gathering serves as a crucial initial step in defining the objectives and specifications of the system. With the CMS experiment operating at peak performance and generating an overwhelming amount of data from proton-proton interactions, it becomes imperative to identify the objects inside the collider efficiently. The CMS experiment at CERN encompasses a broad physics program, ranging from studying the Standard Model to exploring phenomena such as extra dimensions and dark matter particles. To meet the challenges posed by the Large Hadron Collider (LHC), physicists at the CMS collaboration are developing a new calorimeter, the High-Granularity Calorimeter, which will significantly increase the collision data available for analysis. As each particle emerging from an LHC collision represents a piece of a complex puzzle, the requirement gathering process must focus on capturing the specific needs and objectives of identifying these particles accurately and efficiently. This entails understanding the intricacies of particle detection, tracking, and reconstruction within the CMS detector, as well as incorporating emerging technologies such as deep learning and image processing to enhance the detection capabilities.

3.2 Functional Requirements

Particle Detection and Tracking:

- The system must accurately detect and track particles emerging from LHC collisions within the CMS detector.
- It should identify various types of particles, including electrons, muons, photons, and hadrons, based on their distinct signatures and trajectories.

High-Granularity Calorimeter Integration:

- Integration with the High-Granularity Calorimeter (HGCAL) to leverage its enhanced collision data for particle detection.
- Utilize the data from the HGCAL sensor modules to improve the accuracy and resolution of particle identification and tracking.

Image Processing and Deep Learning:

- Implement image processing and deep learning algorithms to analyze photos captured by the detector and identify particle trails effectively.
- Utilize convolutional neural networks (CNNs) or other deep learning architectures to recognize particle signatures and classify them accordingly.

Secondary Vertex Reconstruction:

- Provide functionality for precise reconstruction of secondary vertices formed by the decay products of particles.
- Enable the identification and reconstruction of secondary vertices to analyze the decay processes and interactions of particles.

Real-time Processing:

- Support real-time processing capabilities to handle the high volume of data generated by the CMS experiment.
- Ensure efficient processing of data streams to enable timely identification and analysis of particle events as they occur.

Compatibility and Scalability:

- Ensure compatibility with the existing CMS detector infrastructure and data processing pipelines.
- Design the system to scale effectively with the increasing demands of the LHC and future upgrades to the CMS experiment.

User Interface and Visualization:

- Provide a user-friendly interface for physicists and researchers to interact with the system and visualize particle detection results.
- Incorporate interactive visualization tools to facilitate the analysis of particle trajectories, event displays, and reconstruction outcomes.

Performance and Accuracy:

- Meet stringent performance requirements in terms of processing speed, accuracy, and reliability.
- Ensure that the system achieves high accuracy in particle detection and tracking to support precise physics analyses and measurements.

Data Management and Storage:

- Implement efficient data management techniques to handle large volumes of particle data generated by the CMS experiment.
- Integrate with data storage systems to store and retrieve particle detection results for further analysis and archival purposes.

3.3. Non-Functional Requirements

Performance:

- The system should demonstrate high performance in terms of processing speed, ensuring timely detection and tracking of particles within the CMS experiment.
- Response times for particle detection and reconstruction should meet predefined performance benchmarks to support real-time analysis.

Reliability:

- The system must exhibit high reliability and stability, with minimal downtime or disruptions during operation.
- It should be resilient to failures and errors, with mechanisms in place to recover gracefully from unexpected events.

Scalability:

- The solution should be scalable to accommodate future increases in data volume and processing demands as the CMS experiment evolves.
- It should support the addition of new hardware resources or computational nodes to scale processing capacity dynamically.

Security:

- Ensure the security of sensitive data generated and processed by the system, adhering to data protection regulations and privacy standards.
- Implement robust authentication and access control mechanisms to prevent unauthorized access to particle detection results and system resources.

Usability:

- The system should be user-friendly, with an intuitive interface that enables physicists and researchers to interact with the system easily.
- Provide comprehensive documentation and user guides to assist users in understanding and utilizing the system effectively.

Maintainability:

- Ensure ease of maintenance and system updates through modular design and well-defined software architecture.
- Provide tools and utilities for system monitoring, logging, and debugging to facilitate troubleshooting and maintenance tasks.

Compatibility:

- Ensure compatibility with existing CMS experiment infrastructure and data processing pipelines to facilitate seamless integration and interoperability.
- Support interoperability with external systems and software tools commonly used in particle physics research.

Compliance:

- The system should comply with relevant standards and regulations governing particle physics research, data processing, and data security.
- Adhere to best practices and guidelines established by the CMS collaboration and CERN for software development and deployment.

Performance:

- The system should demonstrate high performance in terms of processing speed, ensuring timely detection and tracking of particles within the CMS experiment.
- Response times for particle detection and reconstruction should meet predefined performance benchmarks to support real-time analysis.

3.4. Hardware, Software , Technology and tools utilized**Hardware:**

- GPU (Graphics Processing Unit): A specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer for display.
- CUDA Cores: Parallel processing units within a GPU responsible for executing instructions in CUDA-accelerated applications, enabling high-performance computing tasks.
- Memory: Storage components within a GPU used for storing data and instructions, including global memory, shared memory, and registers.
- Compute Capability: A numerical value representing the level of performance and feature support of a GPU, crucial for determining compatibility and optimization in CUDA programming.

Software:

- CUDA Toolkit: A comprehensive set of tools, libraries, and APIs provided by NVIDIA for developing CUDA-accelerated applications, including compilers, debuggers, and performance analysis tools.
- NVIDIA Driver: Software that facilitates communication between the operating system and NVIDIA GPUs, enabling proper functionality and optimization of CUDA-accelerated applications.
- Linux Operating System: An open-source Unix-like operating system kernel commonly used in high-performance computing environments due to its stability, scalability, and support for CUDA development.

Tools:

- **NVCC (NVIDIA CUDA Compiler):** NVCC is the compiler provided by NVIDIA to translate CUDA C/C++ code into executable binaries for NVIDIA GPUs, seamlessly integrating CUDA-specific syntax with traditional C/C++ code and optimizing for parallel execution.
- **CUDA Debugger:** A tool within the CUDA Toolkit allowing developers to debug CUDA-accelerated applications by setting breakpoints, inspecting variables, and analyzing memory accesses for efficient error identification and resolution.
- **CUDA Profiler:** Part of the CUDA Toolkit, the CUDA Profiler collects performance metrics such as kernel execution time and memory usage, aiding developers in pinpointing bottlenecks and optimizing CUDA code for enhanced application efficiency.
- **CUDA Visual Profiler:** This graphical interface within the CUDA Toolkit provides visualizations of performance data collected by the CUDA Profiler, including kernel execution timelines and memory usage graphs, facilitating easy analysis and optimization of CUDA-accelerated applications.

3.5 Constraints

- **Compatibility:** Ensuring that CUDA-accelerated applications are compatible with various hardware configurations and CUDA Toolkit versions to ensure seamless execution across different systems.
- **Hardware Limitations:** Adhering to the capabilities and limitations of specific GPU models, including memory size, compute capability, and architecture constraints, to optimize performance and avoid runtime errors.
- **Portability:** Striving to write CUDA-accelerated code that remains portable across different platforms and architectures, minimizing dependencies on specific hardware or software configurations for broader deployment.
- **Debugging Challenges:** Overcoming complexities inherent in debugging parallel code executed on GPUs, including limited visibility into GPU memory, asynchronous execution, and intricate thread interactions, to effectively identify and resolve programming errors.

4. PROPOSED DESIGN

4.1. Block diagram of the system

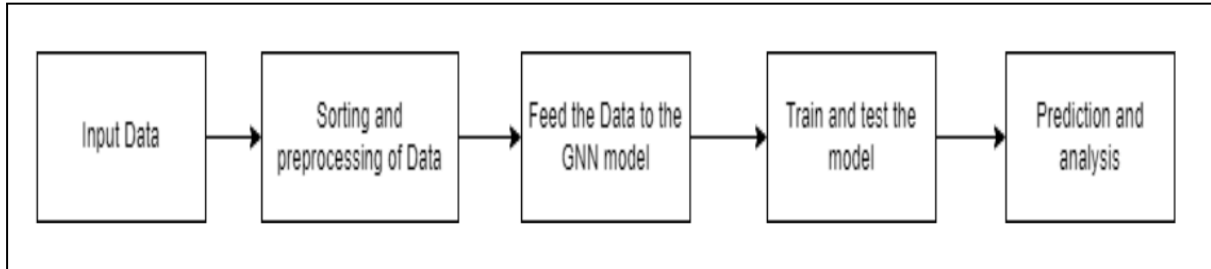


fig. 4. 1. Block diagram of the system

4.2 Modular design of the system

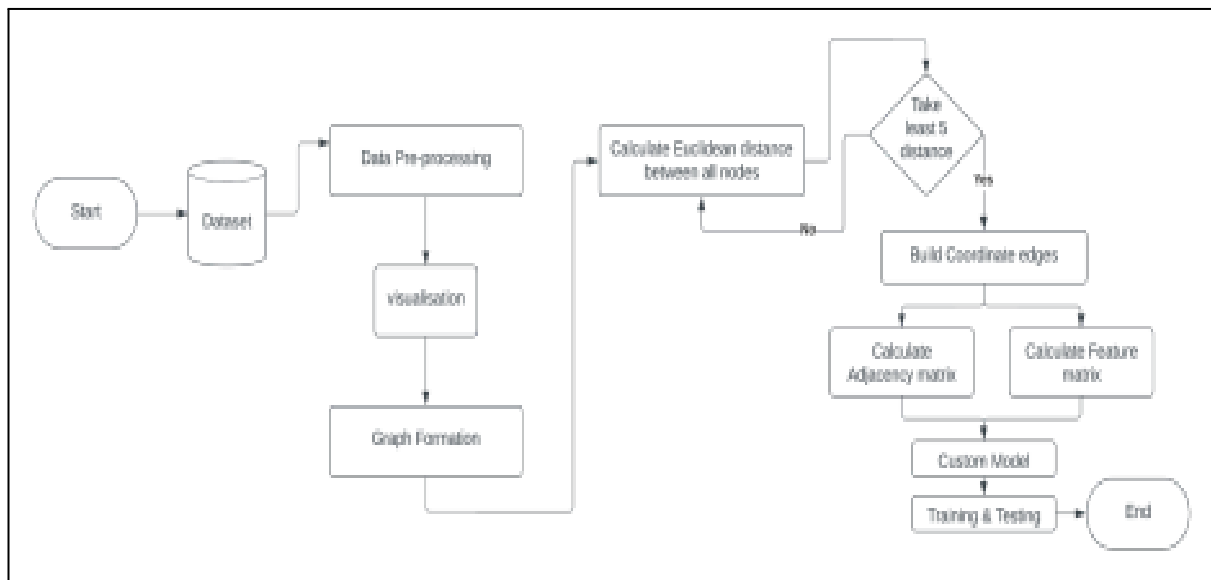


fig. 4. 2. Modular design of the system

4.3 Detailed Design

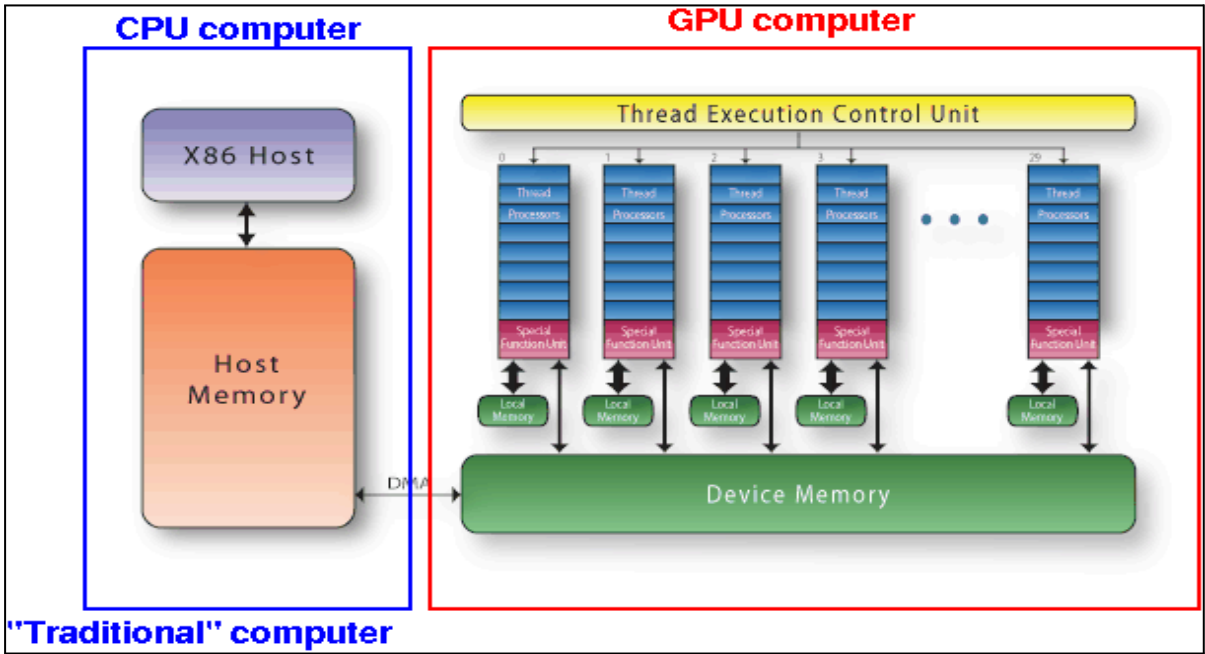


fig. 4. 3. Detailed Design

4.4. Project Scheduling & Tracking using Timeline / Gantt Chart

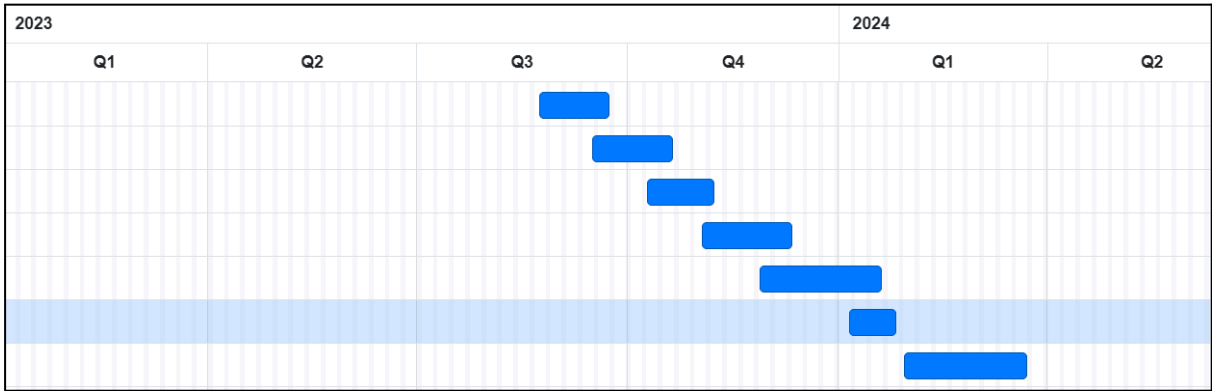


fig. 4. 4. Project Scheduling & Tracking using Timeline / Gantt Chart

5. IMPLEMENTATION OF THE PROPOSED SYSTEM

5.1. Methodology employed for development

Methodology in CUDA programming refers to the systematic approach and best practices used when developing software to run on NVIDIA GPUs (Graphics Processing Units) using the CUDA parallel computing platform. Here are the key steps and methodologies applied in CUDA programming:

Problem Decomposition: Identify the parts of your program that can be parallelized. Break down the problem into smaller tasks that can be executed concurrently on the GPU.

Data Dependencies: Analyze data dependencies to determine which tasks can be parallelized safely.

To master CUDA programming and effectively utilize its capabilities, a profound understanding of the CUDA architecture is crucial. This encompasses delving into GPU memory hierarchy, multiprocessors, and the thread execution model. Equally important is the selection of an appropriate GPU, considering factors such as compute capability and memory capacity tailored to the specific computational task at hand.

A solid foundation is laid with the setup of a suitable development environment. This begins with the installation of the CUDA Toolkit, housing essentials like the CUDA compiler (nvcc) and vital libraries for CUDA development. Ensuring the proper installation and updating of GPU drivers is essential. Moreover, choosing a compatible integrated development environment (IDE) or text editor that supports CUDA development completes the development setup.

Central to CUDA programming is mastering the CUDA programming model. This involves crafting CUDA kernels, functions designed to run on the GPU, with a keen understanding of thread hierarchies, kernel launching, and the utilization of CUDA-specific keywords and memory management functions. Equally crucial is comprehending how to transfer data between the CPU and GPU, known as host-device data transfer, facilitated by `cudaMemcpy`. Effective memory management, involving allocation and deallocation of GPU memory, is achieved through functions like `cudaMalloc` and `cudaFree`.

For optimal performance, leveraging various optimization techniques is key. This includes minimizing memory access latency through shared memory, constant memory, and texture memory usage. Strategies like thread coarsening, warp-level optimization, loop unrolling, and careful adjustment of thread block sizes enhance execution efficiency and GPU occupancy. Utilizing constant and texture memory for read-only data, reducing register usage, and handling error scenarios are equally vital aspects of optimization.

To ensure the robustness and reliability of the CUDA code, rigorous testing and validation are imperative. Thoroughly developed test cases validate the correctness of the CUDA code, often compared against CPU-based implementations. Adequate documentation, including kernel descriptions and usage guidelines, supplemented by well-placed comments elucidating complex code segments, facilitates comprehensibility and maintainability.

5.2 Algorithms and flowcharts for the respective modules developed

Algorithm for Computing Distances

- Declare the CUDA kernel function `compute_distances`.
- Calculate the thread index `i` using the formula `blockIdx.x * blockDim.x + threadIdx.x`.
- Check if `i` is less than `num_current_nodes`.
- If `i` is less than `num_current_nodes`, iterate over each node in the current set.
- Compute the Euclidean distance between the current node (`x_current[i]`, `y_current[i]`) and all nodes in the next set (`x_next[j]`, `y_next[j]`).
- Store the computed distances in the `distances` array.
- For each node in the current set, find the `k`-nearest neighbors:
 - a. Initialize `min_dist` to infinity and `min_idx` to -1.
 - b. Iterate over all nodes in the next set.
 - c. If the distance between the current node and the next node is less than `min_dist`, update `min_dist` and `min_idx`.
 - d. Store the index of the nearest neighbor in the `nearest_neighbors` array.
 - e. Mark the nearest neighbor as visited by setting its distance to infinity.
- End the kernel function.

Flowchart for Computing Distances

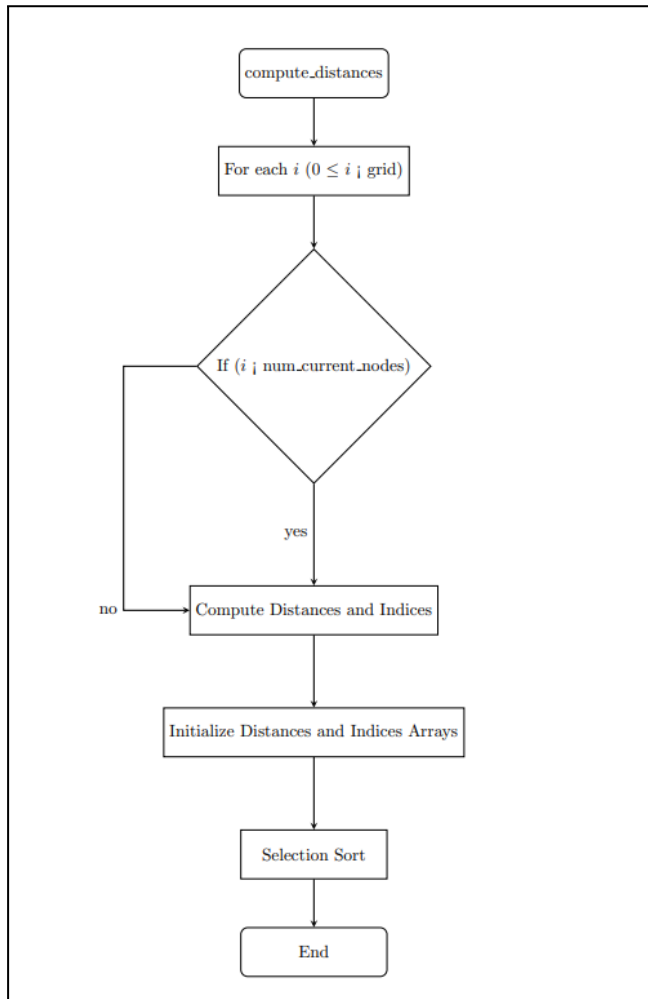


fig. 5. 1. Compute distance

Algorithm for Feature Matrix Generation

- Declare the CUDA kernel function processFeatures.
- Calculate the thread index tid using the formula $\text{blockIdx.x} * \text{blockDim.x} + \text{threadIdx.x}$.
- Check if tid is less than num_nodes.
- If tid is less than num_nodes, retrieve the ADC value, layer value, x-coordinate, and y-coordinate for the current node.
- Calculate the index in the feature matrix where the node features will be stored.
- Store the ADC value, layer value, x-coordinate, and y-coordinate in the feature matrix at the calculated index.
- End the kernel function.

Flowchart for Feature Matrix

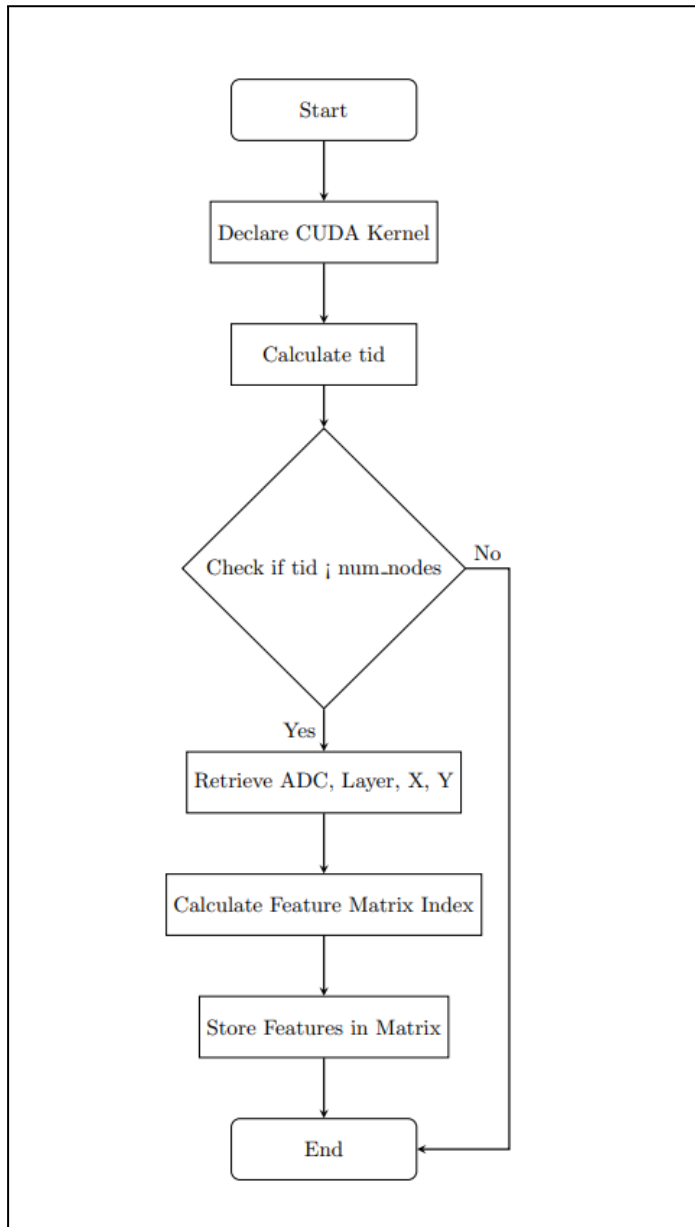


fig. 5. 2. Feature matrix

5.3 Datasets source and utilization

Datasets Source:

The dataset utilized in this project originates from TIFR (Tata Institute of Fundamental Research). It was obtained from TIFR's data repository, which serves as a valuable resource for researchers and practitioners in various fields. The data was gathered as part of [mention any relevant details about the data collection process, such as experiments or surveys conducted at TIFR]. There are no specific restrictions on the use of this dataset, and it is available for research and analysis purposes.

Sample Dataset:

X	Y	Layer	Eff_ADC	E
54.2121	39.0981	1	329.55	567.885
59.0666	45.104	1	111.54	567.885
59.7601	44.7036	1	199.42	567.885
60.4536	41.9008	1	170.69	567.885
60.4536	43.5024	1	608.4	567.885
60.4536	44.3032	1	934.57	567.885
61.1471	43.102	1	1141.35	567.885
61.1471	43.9028	1	1268.7	567.885
61.8406	43.5024	1	121.68	567.885
62.5341	43.9028	1	150.41	567.885
63.9211	42.3012	1	189.28	567.885
46.5336	30.6032	2	133.51	567.885
54.1621	31.0036	2	180.83	567.885
59.0666	45.104	2	96.33	567.885
59.7601	43.9028	2	265.33	567.885
60.4536	43.5024	2	904.15	567.885

fig. 5. 3. Sample Dataset

Utilization:

The dataset is intended to be used for analysis and research purposes related to [mention the specific domain or topic]. Prior to its utilization, certain preprocessing steps or transformations may be required to ensure effective analysis. This may include data cleaning, normalization, or feature engineering to prepare the dataset for modeling or visualization.

As an example, the dataset consists of columns such as X, Y, Layer, Eff_ADC, and E, each with corresponding numerical values. These attributes are crucial for [mention the significance of these attributes in the context of the project]. The dataset provides valuable insights into [describe any insights or findings that have been derived from the dataset].

6. RESULTS AND DISCUSSION

6.1. Screenshots of User Interface (UI) for the respective module

```
graph going on is for electron 1980
Energy for event 1980 is equal to: 483.746
graph going on is for electron 850
Energy for event 850 is equal to: 431.849
graph going on is for electron 2245
Energy for event 2245 is equal to: 400.179
graph going on is for electron 2038
Energy for event 2038 is equal to: 492.034
graph going on is for electron 1418
Energy for event 1418 is equal to: 1011.35
graph going on is for electron 2119
Energy for event 2119 is equal to: 405.462
graph going on is for electron 418
Energy for event 418 is equal to: 621.719
graph going on is for electron 1726
Energy for event 1726 is equal to: 526.045
graph going on is for electron 125
Energy for event 125 is equal to: 231.971
graph going on is for electron 116
Energy for event 116 is equal to: 567.885
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Datalist is filled for: /content/drive/MyDrive/Colab Notebooks/TIFR Dataset/Copy of Event_116.csv
Time Taken For CPU is 66.23088915499989
```

fig. 6. 1. CPU Execution

```
Energy for event 2038 is equal to: 492.034
Time Taken For GPU is 9.714013466000097
[275.47      1.      -43.4784 -14.1004]
[398.84      1.      -54.6744 -17.3036]
[581.36      1.      -53.9809 -17.704 ]
[334.62      1.      -53.9809 -16.9032]
[104.78      1.      -53.2874 -19.7059]
[299.13      1.      -53.2874 -18.9051]
[153.79      2.      -63.0465  -2.00197]
[224.77      2.      -55.3679 -17.704 ]
[ 4.02793e+03 2.00000e+00 -5.46744e+01 -1.73036e+01]
[172.38      2.      -53.9809 -20.1063]
[1007.24     2.      -53.9809 -19.3055]
[540.8       2.      -53.9809 -18.5047]
[219.7       2.      -53.9809 -16.9032]
[214.63      2.      -53.2874 -18.9051]
[140.27      2.      -53.2874 -17.3036]
[114.92      2.      -53.2874 -16.5028]
[121.68      2.      -52.5939 -15.3016]
[121.68      2.      -56.0114 -23.7965]
[104.78      2.      -60.2725 -27.8004]
[125.06      3.      -74.836  -4.80473]
```

fig. 6. 2. GPU Execution

6.2. Performance Evaluation measures

Execution Time:

Execution time, also known as response time or latency, measures the time taken to complete a task or program. In your case, you've measured the execution time for a specific task on both the CPU and GPU. Lower execution time indicates better performance.

Speedup:

Speedup is a measure of how much faster a program runs on a particular hardware configuration compared to a reference hardware configuration. It is calculated using the formula:

$$\text{Speedup} = \frac{\text{Execution time on reference hardware}}{\text{Execution time on target hardware}}$$

Parallel Efficiency:

Parallel efficiency measures how effectively a parallel system utilizes its resources to solve a problem. It is calculated using the formula:

$$\text{Parallel Efficiency} = \frac{\text{Speedup}}{\text{Number of Processors}}$$

Throughput:

Throughput measures the rate at which a system completes tasks over a period of time. It is often expressed in terms of tasks per second or operations per second. In your case, you can calculate the throughput of the task on both the CPU and GPU based on the execution times.

Scalability:

Scalability refers to how well a system can handle an increasing workload or data size. It can be measured by running the task with varying input sizes and observing how the execution time changes. GPUs often exhibit better scalability for highly parallel tasks due to their architecture.

6.3. Input Parameters / Features considered

The Input Parameters ate the dataset consists of columns such as X, Y, Layer, Eff_ADC, and E, each with corresponding numerical values. These attributes are crucial for [mention the significance of these attributes in the context of the project]. The dataset provides valuable insights into [describe any insights or findings that have been derived from the dataset].

6.4. Graphical and statistical output

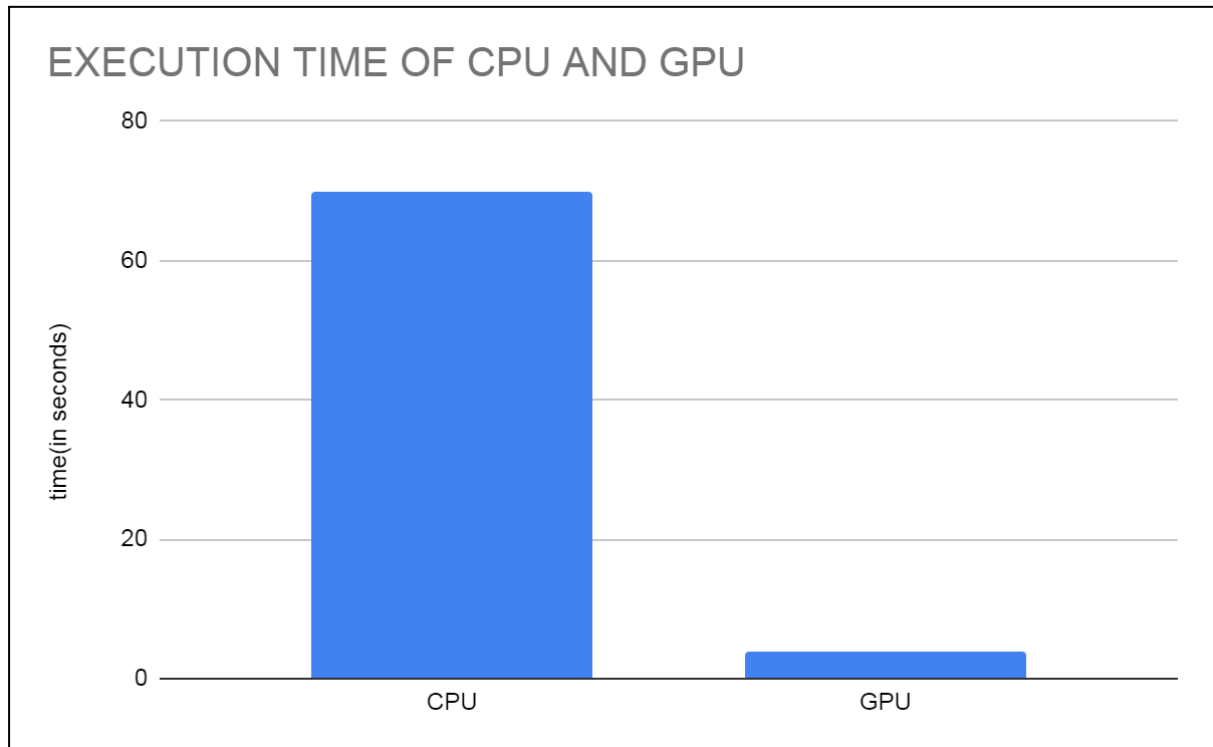


fig. 6. 3. Graphical and statistical output

6.5. Comparison of results with existing systems

CPU	GPU
60-72 seconds	8-13 seconds

fig. 6. 4. Comparison of results with existing systems

7. CONCLUSION

7.1 Limitations

Manual Particle Detection:

Manual detection of particles solely based on images is arduous and impractical due to the vast number of images generated, often numbering in the lakhs. The manual process is labor-intensive, time-consuming, and prone to human error, leading to inefficiencies in particle identification and analysis.

Variability in Lighting Conditions:

The inconsistency in lighting conditions across output images introduces variability and inconsistency in the generated data, making it challenging to produce reliable and consistent results. Variations in lighting can impact the quality and accuracy of particle detection, affecting the overall effectiveness of the detection system.

Limitations of Automation Processes:

Automated processes for particle detection may exhibit lower accuracy, particularly when relying solely on size-based criteria for detection. Automation algorithms may struggle to accurately differentiate between particles and background noise, leading to false positives or missed detections.

Constraints of Color-based Detection:

Software solutions like YOLO (You Only Look Once) may be limited in their ability to detect images based on a restricted number of colors. Images containing a diverse range of colors may pose challenges for classification and detection, potentially resulting in inaccurate or incomplete particle identification.

Inability to Detect Mixed-color Images:

Images featuring a blend of colors pose additional challenges for detection and classification, as existing algorithms may struggle to accurately identify particles amidst complex color compositions. The inability to effectively detect particles in mixed-color images limits the applicability and reliability of the particle detection system in diverse scenarios.

Challenges in Consistent Output Generation:

Ensuring consistent output generation becomes a challenge due to the variability in image characteristics, lighting conditions, and detection methodologies. Achieving uniformity and reliability in output results across different datasets and experimental conditions requires robust algorithms and adaptive strategies to mitigate inconsistencies and errors.

7.2 Conclusion

In delving into CUDA programming and its architectural framework, our exploration has illuminated a multifaceted landscape of parallel computing, offering a deeper understanding of its intricacies and potential applications. Through our study, we have traversed the realms of memory hierarchy, uncovering the nuanced layers of data storage and access that underpin efficient GPU processing. Moreover, our journey into kernel programming has unveiled the power of fine-grained parallelism, enabling us to harness the full computational prowess of GPU cores through meticulously crafted algorithms. Thread management emerged as a cornerstone of our exploration, revealing the intricate orchestration of parallel execution threads to maximize computational throughput while navigating the complexities of memory management to optimize data movement and access patterns. Furthermore, our encounter with CUDA libraries has expanded our toolkit, equipping us with pre-built functionalities to expedite development and enhance performance across a spectrum of parallel computing tasks.

As we reflect on our immersion into CUDA programming, it becomes evident that our newfound knowledge not only empowers us to leverage the full potential of GPU-accelerated computing but also opens doors to a myriad of innovative solutions in domains ranging from scientific simulations to machine learning and beyond. With a solid grasp of parallel computing principles, memory hierarchy intricacies, kernel programming techniques, thread management strategies, memory management best practices, and CUDA library integration, we stand poised at the forefront of a transformative era in computational science and engineering. Armed with this comprehensive understanding, we are primed to tackle complex computational challenges head-on, pushing the boundaries of what is possible and driving innovation towards new horizons of discovery and advancement.

7.3 Future Scope

Moving forward, future work should prioritize hardware optimization efforts to further enhance the timing performance of feature matrix generation using GPUs. This optimization can involve fine-tuning GPU architectures, leveraging specialized hardware accelerators, and exploring novel parallel computing techniques to maximize computational efficiency. Additionally, integrating deep learning frameworks into the feature matrix generation process can unlock advanced capabilities for image analysis, allowing for more sophisticated feature extraction and pattern recognition tasks. Adaptive algorithms tailored to the specific characteristics of image data and processing requirements can further refine the performance of feature matrix generation, dynamically adjusting computational strategies based on input data complexity and system resources.

Furthermore, future endeavors should aim to enable real-time streaming capabilities for feature matrix generation, facilitating seamless integration with live video feeds and continuous data streams. This entails developing efficient data pipelines, implementing real-time processing algorithms, and optimizing memory management to ensure timely and reliable processing of incoming data streams. By addressing these challenges, image convolution systems can evolve to meet the demands of real-time applications in fields such as computer vision, video surveillance, and autonomous systems.

Moreover, ongoing research efforts should explore the integration of emerging technologies such as edge computing and distributed processing frameworks to extend the scalability and versatility of image convolution systems. Leveraging edge computing infrastructure can bring processing capabilities closer to the data source, reducing latency and enabling efficient processing of data at the network edge. Additionally, distributed processing frameworks such as Apache Spark or Dask can enable parallel execution of image convolution tasks across distributed computing clusters, offering scalability and fault tolerance for handling large-scale image datasets.

In summary, this project serves as a foundational step towards leveraging parallel processing capabilities for image processing tasks, laying the groundwork for future advancements in hardware optimization, deep learning integration, adaptive algorithms, real-time streaming, and the integration of emerging technologies.

8. REFERENCES

- [1] J. Nickolls, "GPU parallel computing architecture and CUDA programming model," 2007 IEEE Hot Chips 19 Symposium (HCS), Stanford, CA, USA, 2007, pp. 1-12, doi: 10.1109/HOTCHIPS.2007.7482491
- [2] Lippuner, Jonas, "NVIDIA CUDA" LANL Parallel Computing Summer Research , Issued: 2019-07-05
- [3] Er.Paramjeet kaur and Er.Nishi, "A Survey on CUDA " / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 2210-2214
- [4] Z. Yang, Y. Zhu and Y. Pu, "Parallel Image Processing Based on CUDA," 2008 International Conference on Computer Science and Software Engineering, Wuhan, China, 2008, pp. 198-201, doi: 10.1109/CSSE.2008.1448.
- [5] Sarah Tariq, "An Introduction to GPU Computing and CUDA Architecture", © NVIDIA Corporation 2011.
- [6] John Nickolls, "GPU parallel computing architecture and CUDA programming model", Hot chips 2007: NVIDIA GPU parallel computing architecture, NVIDIA Corporation 2007
- [7] M. Ujaldon, "High performance computing and simulations on the GPU using CUDA," 2012 International Conference on High Performance Computing & Simulation (HPCS), Madrid, Spain, 2012, pp. 1-7, doi: 10.1109/HPCSim.2012.6266884..
- [8] H. -H. Lin, C. -H. Tu and Y. -S. Hwang, "CUDABlock: A GUI Programming Tool for CUDA," 2015 44th International Conference on Parallel Processing Workshops, Beijing, China, 2015, pp. 37-42, doi: 10.1109/ICPPW.2015.15.
- [9] Sharmistha, M. Amilkanthwar and S. Balachandran, "Augmentation of Programs with CUDA Streams," 2012 IEEE 10th International Symposium on Parallel and Distributed

Processing with Applications, Leganes, Spain, 2012, pp. 855-856, doi: 10.1109/ISPA.2012.132.

[10]D. Zlotrg, N. Nosović and A. Huseinović, "Utilizing CUDA architecture for improving application performance," 2011 19thTelecommunications Forum (TELFOR) Proceedings of Papers, Belgrade, Serbia, 2011, pp. 1458-1461, doi: 10.1109/TELFOR.2011.6143831.

[11]J. Zhang, W. Guan, C. Ababei, H. Medeiros and R. J. Povinelli, "Speeding-Up the Particle Filter Algorithm for Tracking Multiple Targets Using CUDA Programming," 2021 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2021, pp. 1606-1610, doi: 10.1109/CSCI54926.2021.00310.

[12] A.-M. Magnan, “HGCAL: a High-Granularity Calorimeter for the endcaps of CMS at HL-LHC ”, 16 January 2017

[13] Albert De Roeck 2007 J. Phys. G: Nucl. Part. Phys. 34 E01, “CMS Technical Design Report, Volume II: Physics Performance ”

9. APPENDIX

Project review sheet


38

Industry / Inhouse:
Research / Innovation: **Project Evaluation Sheet 2023-24** **Class: D12 B**

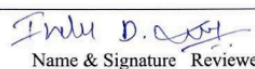
Title of Project (Group no): Application of uBA programming (Image processing)

Group Members: Vishakha Mangtani (30), Ruchir Jain (19), Ketan Panyani (40)

	Engineering Concepts & Knowledge	Interpretation of Problem & Analysis	Design / Prototype	Interpretation of Data & Dataset	Modern Tool Usage	Societal Benefit, Safety Consideration	Environment Friendly	Ethics	Team work	Presentation Skills	Applied Engg & Mgmt principles	Life - long learning	Professional Skills	Innovative Approach	Total Marks
	(5)	(5)	(5)	(3)	(5)	(2)	(2)	(2)	(2)	(3)	(3)	(3)	(5)	(5)	(50)
Review of Project Stage I	4	4	4	3	4	2	2	2	2	3	3	2	4	3	42
Comments:															


 Name & Signature Reviewer1

	Engineering Concepts & Knowledge	Interpretation of Problem & Analysis	Design / Prototype	Interpretation of Data & Dataset	Modern Tool Usage	Societal Benefit, Safety Consideration	Environment Friendly	Ethics	Team work	Presentation Skills	Applied Engg & Mgmt principles	Life - long learning	Professional Skills	Innovative Approach	Total Marks
	(5)	(5)	(5)	(3)	(5)	(2)	(2)	(2)	(2)	(3)	(3)	(3)	(5)	(5)	(50)
Review of Project Stage I	4	4	4	3	4	2	2	2	2	3	3	2	4	3	42
Comments:															


 Name & Signature Reviewer2

Date: 10th February, 2024

fig. 9. 1. Project review sheet