# Visual Assets for GPU Programming Report

This document provides descriptions and links to visual assets that can be used in your GPU programming report and presentation. These visuals will help illustrate key concepts and make your report more engaging.

## Table of Contents

1. [GPU vs CPU Architecture Diagrams](#architecture-diagrams)

2. [Performance Comparison Charts](#performance-charts)

3. [Application-Specific Visuals](#application-visuals)

4. [Code Visualization](#code-visualization)

5. [Hardware Images](#hardware-images)

<a name="architecture-diagrams"></a>

## 1. GPU vs CPU Architecture Diagrams

### 1.1 Basic CPU vs GPU Architecture Comparison

![CPU vs GPU Architecture](https://www.researchgate.net/publication/334157146/figure/fig1/AS:776599045545984@1562167080511/Difference-between-CPU-and-GPU-architecture-9.png)

\*\*Description\*\*: This diagram illustrates the fundamental architectural differences between CPUs and GPUs. CPUs have a few powerful cores with large caches and complex control logic, while GPUs have many simple cores optimized for parallel processing.

### 1.2 Modern GPU Architecture

![NVIDIA GPU Architecture](https://developer-blogs.nvidia.com/wp-content/uploads/2018/09/cuda-10-nvidia-turing-architecture-sm-diagram.png)

\*\*Description\*\*: This diagram shows the architecture of NVIDIA's Turing GPU architecture, including Streaming Multiprocessors (SMs), CUDA cores, Tensor cores, and memory hierarchy.

### 1.3 Memory Hierarchy in GPUs

![GPU Memory Hierarchy](https://www.researchgate.net/publication/301417070/figure/fig1/AS:669499422363656@1536632548553/Memory-hierarchy-of-modern-GPUs.png)

\*\*Description\*\*: This diagram illustrates the memory hierarchy in modern GPUs, showing the relationship between global memory, shared memory, L1/L2 caches, and registers.

<a name="performance-charts"></a>

## 2. Performance Comparison Charts

### 2.1 General Performance Comparison

![CPU vs GPU Performance](https://www.researchgate.net/publication/329530140/figure/fig3/AS:702296356605954@1544476202878/CPU-vs-GPU-performance-comparison-for-different-applications.png)

\*\*Description\*\*: This chart compares CPU and GPU performance across different applications, showing the speedup factor achieved by GPU acceleration.

### 2.2 Deep Learning Training Performance

![Deep Learning Performance](https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-t4/t4-inference-performance-full-precision.png)

\*\*Description\*\*: This chart shows the performance improvement in deep learning training when using GPU acceleration compared to CPU-only implementations.

### 2.3 Scientific Computing Performance

![Scientific Computing Performance](https://www.researchgate.net/publication/301417070/figure/fig4/AS:669499422367748@1536632548556/Performance-comparison-of-CPU-and-GPU-implementations-of-the-FDTD-method.png)

\*\*Description\*\*: This chart compares CPU and GPU performance for scientific computing applications, specifically for Finite-Difference Time-Domain (FDTD) simulations.

<a name="application-visuals"></a>

## 3. Application-Specific Visuals

### 3.1 AI and Machine Learning

![Neural Network Visualization](https://miro.medium.com/max/1400/1\*Uw4zAW6k0v4aTejDgxMwZg.png)

\*\*Description\*\*: This visualization shows a neural network architecture that can be accelerated using GPUs.

### 3.2 Medical Imaging

![Medical Imaging GPU Acceleration](https://www.nvidia.com/content/dam/en-zz/Solutions/gtc/ai-for-healthcare/clara-imaging-workflow-fullwidth.jpg)

\*\*Description\*\*: This image illustrates how GPU acceleration is used in medical imaging workflows, from data acquisition to analysis and visualization.

### 3.3 Molecular Dynamics Simulation

![Molecular Dynamics Visualization](https://www.researchgate.net/publication/301417070/figure/fig5/AS:669499422371844@1536632548557/Visualization-of-the-MD-simulation-of-a-peptide-in-water.png)

\*\*Description\*\*: This visualization shows a molecular dynamics simulation of a peptide in water, a computationally intensive task that benefits greatly from GPU acceleration.

### 3.4 Cryptocurrency Mining

![GPU Mining Farm](https://www.researchgate.net/publication/343089939/figure/fig1/AS:915744311799809@1595351332396/A-GPU-mining-farm-for-cryptocurrency-mining.png)

\*\*Description\*\*: This image shows a GPU mining farm used for cryptocurrency mining, one of the most visible applications of GPU computing.

<a name="code-visualization"></a>

## 4. Code Visualization

### 4.1 CUDA Execution Model

![CUDA Execution Model](https://developer-blogs.nvidia.com/wp-content/uploads/2017/01/cuda\_indexing.png)

\*\*Description\*\*: This diagram illustrates the CUDA execution model, showing how threads are organized into blocks and grids for parallel execution on the GPU.

### 4.2 Parallel vs Sequential Processing

![Parallel vs Sequential](https://www.researchgate.net/publication/301417070/figure/fig2/AS:669499422367744@1536632548554/Comparison-of-sequential-and-parallel-execution-of-a-program.png)

\*\*Description\*\*: This diagram compares sequential processing (CPU) with parallel processing (GPU), showing how tasks can be executed simultaneously on multiple cores.

### 4.3 CUDA Memory Transfer

![CUDA Memory Transfer](https://developer-blogs.nvidia.com/wp-content/uploads/2018/09/unified-memory-cuda-10-paste-2018.png)

\*\*Description\*\*: This diagram shows how data is transferred between CPU and GPU memory in CUDA programming, including the unified memory model introduced in newer CUDA versions.

<a name="hardware-images"></a>

## 5. Hardware Images

### 5.1 Modern GPU Hardware

![NVIDIA RTX GPU](https://www.nvidia.com/content/dam/en-zz/Solutions/geforce/ampere/rtx-3090/geforce-rtx-3090-shop-630-d.png)

\*\*Description\*\*: This image shows a modern NVIDIA RTX GPU, highlighting the physical hardware that enables GPU computing.

### 5.2 GPU Computing Server

![GPU Computing Server](https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-a100/nvidia-dgx-a100-hero-t.jpg)

\*\*Description\*\*: This image shows an NVIDIA DGX server, a high-performance computing system designed specifically for GPU-accelerated workloads.

### 5.3 GPU Architecture Evolution

![GPU Architecture Evolution](https://www.researchgate.net/publication/301417070/figure/fig3/AS:669499422367746@1536632548555/Evolution-of-the-peak-single-precision-floating-point-performance-for-NVIDIA-GPUs-and.png)

\*\*Description\*\*: This chart shows the evolution of GPU performance over time, highlighting the rapid advancement of GPU computing capabilities.

## How to Use These Visuals

1. \*\*For your report\*\*: Include these visuals with proper citations to illustrate key concepts and make your report more engaging.

2. \*\*For your presentation\*\*: Use these visuals in your slides to help your audience understand complex GPU concepts visually.

3. \*\*For code examples\*\*: Pair code examples with the relevant architecture diagrams to show how the code maps to the hardware.

4. \*\*For performance analysis\*\*: Use the performance charts to demonstrate the benefits of GPU acceleration in various applications.

## Creating Custom Visuals

If you need custom visuals for your report, consider using these tools:

1. \*\*Diagrams.net (formerly draw.io)\*\*: Free online diagramming tool for creating architecture diagrams

- URL: https://app.diagrams.net/

2. \*\*Matplotlib (Python)\*\*: For creating custom performance charts

- Example code for creating a CPU vs GPU performance comparison chart:

```python

import matplotlib.pyplot as plt

import numpy as np

applications = ['Neural Networks', 'Molecular Dynamics', 'Video Processing',

'Financial Modeling', 'Medical Imaging']

cpu\_performance = [1, 1, 1, 1, 1] # Baseline

gpu\_performance = [45, 25, 10, 35, 20] # Speedup factors

x = np.arange(len(applications))

width = 0.35

fig, ax = plt.subplots(figsize=(12, 6))

cpu\_bars = ax.bar(x - width/2, cpu\_performance, width, label='CPU')

gpu\_bars = ax.bar(x + width/2, gpu\_performance, width, label='GPU')

ax.set\_ylabel('Relative Performance')

ax.set\_title('CPU vs GPU Performance Comparison')

ax.set\_xticks(x)

ax.set\_xticklabels(applications)

ax.legend()

# Add speedup labels

for i, v in enumerate(gpu\_performance):

ax.text(i + width/2, v + 0.5, f'{v}x', ha='center')

plt.tight\_layout()

plt.savefig('performance\_comparison.png', dpi=300)

plt.show()

```

3. \*\*Blender\*\*: For creating 3D visualizations of complex concepts

- URL: https://www.blender.org/

## References for Visuals

1. NVIDIA Developer Blog. (2023). CUDA Programming Model.

2. Research Gate. (2021). Various publications on GPU computing.

3. NVIDIA. (2023). Product images and performance charts.

4. Various academic papers on GPU computing and parallel processing.