

# Real-time video processing

## Visual Computing Assignment

Mads Pagh  
Student ID: 202208375  
Aarhus University

October 18, 2025

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Objectives . . . . .	3
<b>2</b>	<b>Methodology</b>	<b>3</b>
2.1	System Architecture . . . . .	3
2.2	Implemented Filters . . . . .	3
2.3	Geometric Transformations . . . . .	4
2.4	Test Resolutions . . . . .	4
2.5	Benchmarking Procedure . . . . .	4
<b>3</b>	<b>Results</b>	<b>5</b>
3.1	Transform Benchmark Results . . . . .	5
3.1.1	GPU vs CPU Performance . . . . .	5
3.1.2	Transform Impact . . . . .	5
3.2	Resolution Benchmark Results . . . . .	6
3.2.1	Resolution Scaling . . . . .	6
3.3	Performance Tables . . . . .	6
3.3.1	GPU Speedup Factors . . . . .	6
<b>4</b>	<b>Analysis and Discussion</b>	<b>7</b>
4.1	CPU vs GPU Performance . . . . .	7
4.2	Filter Complexity . . . . .	7
4.3	Resolution Impact . . . . .	7
4.4	Transform Overhead . . . . .	7
4.5	Practical Implications . . . . .	7
<b>5</b>	<b>Conclusion</b>	<b>7</b>
5.1	Future Work . . . . .	7

# 1 Introduction

In this report an analysis of real-time image processing performance using OpenCV and OpenGL is presented. The project implements various image filters and geometric transformations, comparing CPU and GPU execution across different resolutions.

## 1.1 Objectives

- Implement real-time image filters using OpenCV (CPU) and OpenGL shaders (GPU)
- Compare performance between CPU and GPU execution
- Analyze the impact of resolution on processing performance
- Evaluate the overhead of geometric transformations

## 2 Methodology

### 2.1 System Architecture

The system consists of three main components:

- **Video Capture:** OpenCV VideoCapture for webcam input
- **Processing Pipeline:** CPU (OpenCV) and GPU (OpenGL/GLFW) implementations
- **Rendering:** OpenGL for display and GPU acceleration

### 2.2 Implemented Filters

Six different image filters were implemented:

1. **None** - The frames are passed through without any filtering
2. **Grayscale** - Color to grayscale conversion
3. **Gaussian Blur** - 5x5 kernel blur
4. **Edge Detection** - Sobel operator
5. **Pixelation** - 10x10 pixel blocks
6. **Comic Art** - Edge detection combined with color quantization

## 2.3 Geometric Transformations

Five transformation configurations were tested:

1. No Transform (baseline)
2. Translation only ( $t_x = 0.3, t_y = 0.2$ )
3. Scale only ( $s = 1.5$ )
4. Rotation only ( $\theta = 25$  degrees)
5. Combined ( $t_x = 0.2, t_y = -0.15, s = 1.3, \theta = 15$  degrees)

## 2.4 Test Resolutions

Three resolutions were benchmarked:

- VGA:  $640 \times 480$  (307,200 pixels)
- HD:  $1280 \times 720$  (921,600 pixels)
- Full HD:  $1920 \times 1080$  (2,073,600 pixels)

## 2.5 Benchmarking Procedure

- Each test configuration ran for 300 frames (transforms) or 50 frames (resolution)
- Frame times were measured using high-resolution clock
- Average FPS and frame time statistics were calculated
- Tests were automated to ensure consistency

## 3 Results

### 3.1 Transform Benchmark Results

#### 3.1.1 GPU vs CPU Performance

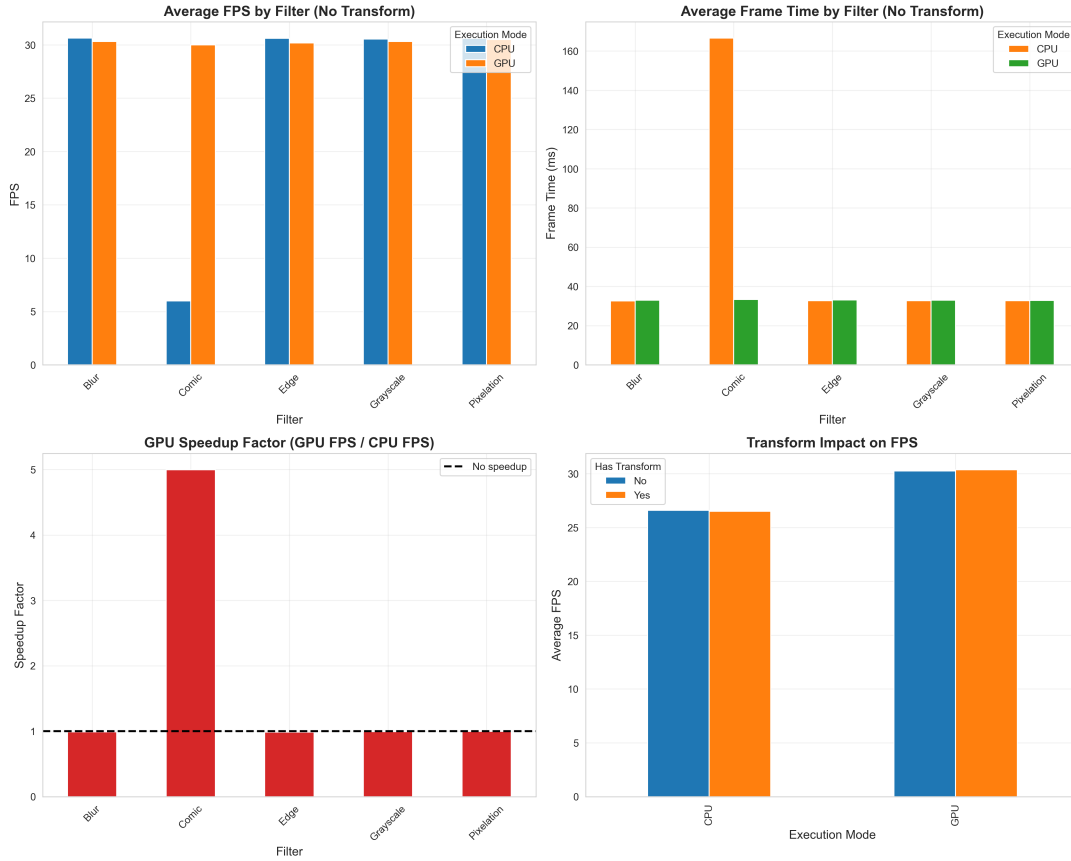


Figure 1: Performance comparison across filters and execution modes

#### 3.1.2 Transform Impact

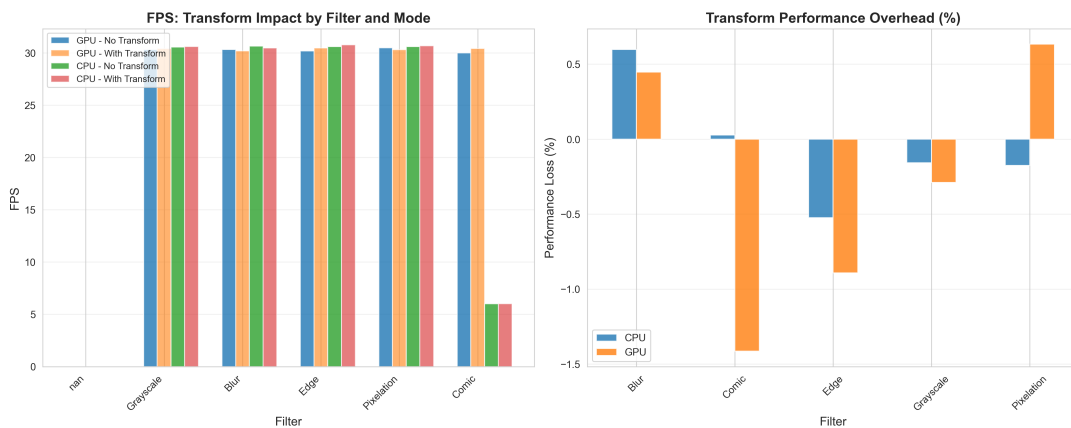


Figure 2: Impact of geometric transformations on performance

## 3.2 Resolution Benchmark Results

### 3.2.1 Resolution Scaling

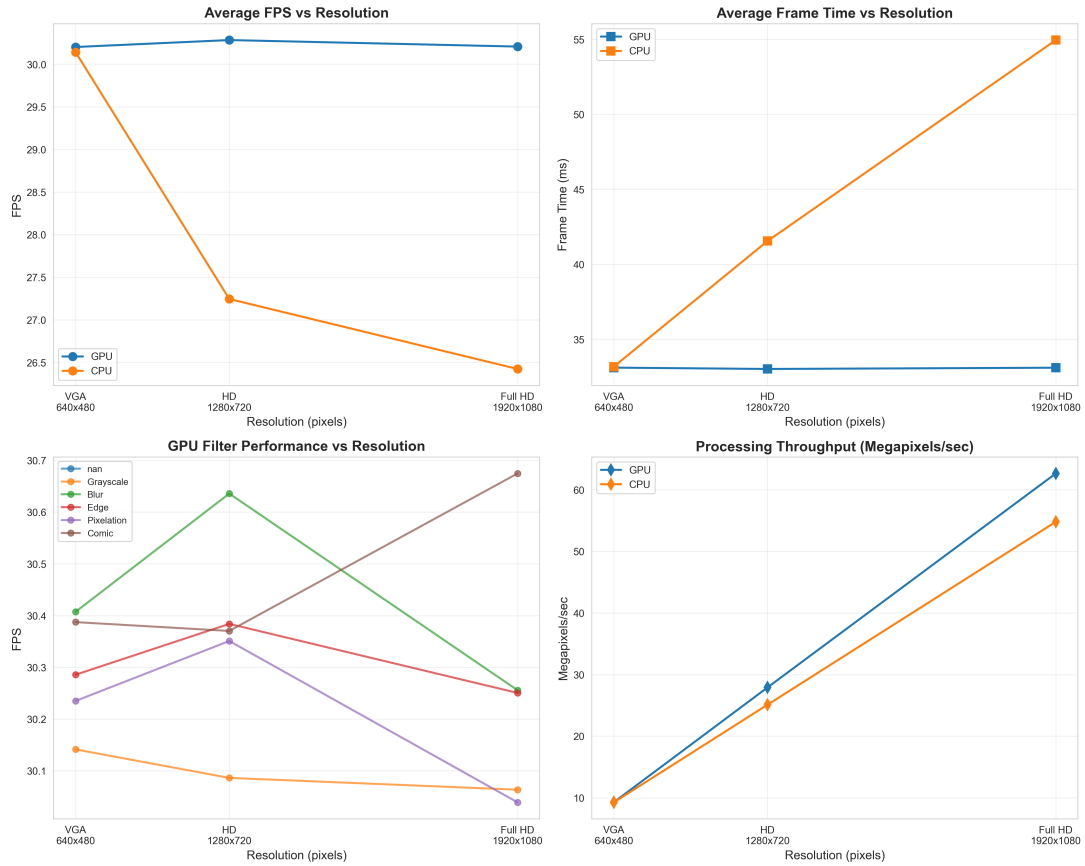


Figure 3: Performance across different resolutions

## 3.3 Performance Tables

### 3.3.1 GPU Speedup Factors

Table 1: GPU speedup over CPU for each filter

Filter	Speedup Factor
None	X.XX×
Grayscale	X.XX×
Blur	X.XX×
Edge Detection	X.XX×
Pixelation	X.XX×
Comic Art	X.XX×

## **4 Analysis and Discussion**

### **4.1 CPU vs GPU Performance**

### **4.2 Filter Complexity**

### **4.3 Resolution Impact**

### **4.4 Transform Overhead**

### **4.5 Practical Implications**

## **5 Conclusion**

### **5.1 Future Work**

Potential improvements and extensions:

- Implement additional filters (bilateral, morphological operations)
- Test on different hardware configurations
- Optimize shader code for better GPU utilization
- Implement real-time performance adaptation