การประยุกต์ใช้ไดเรกต์คอมพิวในการประมวลผลภาพ

APPLICATION OF DIRECTCOMPUTE IN IMAGE PROCESSING

ปิยวัฒน์ บัวขม
กิจจ์พัฒน์ กิตติวัฒนกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2557

การประยุกต์ใช้ไดเรกต์คอมพิวในการประมวลผลภาพ

APPLICATION OF DIRECTCOMPUTE IN IMAGE PROCESSING

ปิยวัฒน์ บัวขม

กิจจ์พัฒน์ กิตติ์วัฒนกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2557

ปริญญานิพนธ์ปีการศึกษา 2557

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การประยุกต์ใช้ไดเรกต์คอมพิวในการประมวลผลภาพ

APPLICATION OF DIRECTCOMPUTE IN IMAGE PROCESSING

ผู้จัดทำ

1. นายปิยวัฒน์ บัวขม รหัสนักศึกษา 54010820
2. นายกิจจ์พัฒน์ กิตติ์วัฒนกุล รหัสนักศึกษา 54011413

_____ อาจารย์ที่ปรึกษา

(อารจารย์ วัจนพงศ์ เกษมศิริ)

การประยุกต์ใช้ไดเรกต์คอมพิวในการประมวลผลภาพ

นายปิยวัฒน์ บัวขม 54010820
นายกิจจ์พัฒน์ กิตติ์วัฒนกุล 54011413
อาจารย์วัจนพงศ์ เกษมศิริ อาจารย์ที่ปรึกษา
ปีการศึกษา 2557

บทคัดย่อ

DirectCompute API (compute shader) คือ API ที่สนับสนุนการประมวลผลทั่วไปบน GPU ซึ่ง DirectCompute เป็นส่วนหนึ่งของ Microsoft DirectX ซึ่งทำให้ผู้ใช้งานสามารถพัฒนาโปรแกรมบน GPU โครงงานนี้จะสร้างและวิเคราะห์ความแตกต่างระหว่างประสิทธิภาพของ image-processing function ที่ใช้ library จาก DirectCompute (parallel) และ Aforge (sequential)

ซึ่ง Image-processing ที่ library ที่พัฒนาขึ้นมาจาก DirectCompute โดยมี Aforge เป็นต้นแบบนั้น จะใช้ SharpDX framework ในการพัฒนา ซึ่งเป็นปรับการพัฒนาให้อยู่บนสภาพแวดล้อมเดียวกับ Aforge ซึ่งใช้ C# ในการพัฒนา โดยยังพัฒนาแอปพลิเคชั่นให้อยู่ใน windows form และแสดงเวลาที่ใช้ในการทำงานเพื่อที่จะได้สามารถเปรียบเทียบประสิทธิภาพการทำงานได้โดยสะดวก

APPLICATION OF DIRECTCOMPUTE IN IMAGE PROCESSING

Mr.Piyawath Boukom	54010820
Mr.Kitphat Kittwattanakul	54011413
Mr.Watjanapong Kasemsiri	Advisor

Academic Year 2014

Abstract

DirectCompute API (compute shader) is an application programming interface (API) that supports general-purpose computing on the GPU, DirectCompute is part of the Microsoft DirectX collection of APIs, which makes users develop general parallel programs. This paper develops and analyzes the distinct features of image-processing function made by Aforge library and image-processing function made by DirectCompute API. We will take compare between 2 library, in term of computational speed.

Further more, we make this application in windows form that show computational time to make it easy to use and compare performance.

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงไปด้วยดี เนื่องจากได้รับความช่วยเหลือ แนะนำเป็นอย่างดีจากอาจารย์ที่ปรึกษา คือ อ.วัจนพงศ์ เกษมศิริ ในการแนะนำ ชี้แนะ ให้ข้อเสนอแนะ ตรวจสอบ ข้อบกพร่องของการทำโครงงาน และติดตามความก้าวหน้าในการดำเนินการโครงงาน กลุ่มผู้จัดทำโครงงานรู้สึกซาบซึ้งในความกรุณาของอาจารย์เป็นอย่างยิ่ง และขอขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ท้ายที่สุด กลุ่มผู้จัดทำโครงงานหวังว่าโครงงานฉบับนี้จะเป็นประโยชน์แก่ผู้สนใจไม่มากก็น้อย สำหรับข้อบกพร่องต่างๆที่อาจจะเกิดขึ้นนั้น กลุ่มผู้จัดทำโครงงานขอน้อมรับผิดทุกประการ และยินดีที่จะรับฟังคำแนะนำจากทุกท่านที่ได้เข้ามาศึกษา เพื่อเป็นประโยชน์ในการพัฒนาโครงงานต่อไป

นายปิยวัฒน์ บัวขม

นายกิจจ์พัฒน์ กิตติ์วัฒนกุล

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

ตาราง **หน้า**

สารบัญตาราง(ต่อ)

ตาราง **หน้า**

สารบัญรูป

รูป หน้า

สารบัญรูป(ต่อ)

สารบัญรูป(ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาของโครงงาน

DirectCompute คือ API ของ Microsoft ที่สนับสนุน GPGPU (General-purpose computing on graphic processing units) บน Microsoft Windows Vista, 7 ซึ่งทำงานแบบ Data-parallel Computing เป็น API ที่ค่อนข้างใหม่ที่ใช้ใน Image processing เมื่อเทียบกับ OpenCL หรือ CUDA ที่มีมาก่อนแล้ว

ใน Image processing library ทั่วๆไปที่ไม่ได้ทำงานแบบ Parallel computing นั้นจะใช้ CPU ในการประมวลผลเพียงอย่างเดียว ซึ่งในการประมวลผลภาพที่ความละเอียดสูงนั้น จะใช้เวลาในการประมวลผลสูง จึงไม่สามารถใช้ Image processing library ทั่วๆไปที่ใช้ Serial computing ในลักษณะงานที่ต้องการประมวลผลแบบ Real-time ดังนั้น Parallel compute image processing library จึงถูกนำมาใช้แก้ปัญหาเหล่านี้ซึ่งในโครงงานนี้จะใช้ Aforge.imaging ที่เป็น opensource มาเป็นตัวต้นแบบ ในการพัฒนา

1.2 วัตถุประสงค์ของโครงงาน

1) ศึกษาการทำงานและพัฒนา Image processing library โดย DirectCompute
2) ศึกษา Aforge.imaging library
3) ศึกษา Image processing algorithm
4) ศึกษา Parallel compute image processing
5) เพื่อให้ผู้พัฒนามีประสบการณ์การพัฒนาโปรแกรมบน GPU
6) เพื่อสร้าง Parallel compute image processing library ที่ใช้ GPU เป็นตัวประมวลผล
7) เพื่อสร้าง Software ทดสอบประสิทธิภาพการทำงานระหว่าง Parallel compute image processing library และ Aforge Library

1.3 ทฤษฎีที่ใช้ในโครงงาน

การพัฒนา Image processing library โดยใช้ DirectCompute ด้วย HLSL (High level shader language) โดยเริ่มจากศึกษา algorithm ของ Serial compute image processing library และนำมาปรับใช้เป็น Parallel compute image processing library โดยใช้ algorithm แบบ Parallel computing ที่ใช้ GPU เป็นตัวประมวลผล

1.4 ขอบเขตของโครงงาน

1) พัฒนา Image processing library โดยใช้ DirectCompute
2) ทดสอบและเปรียบเทียบความเร็วในการทำงานระหว่าง Aforge.imaging และ DirectCompute ที่ใช้ Library Aforge.imaging เป็นต้นแบบ

1.5 ขั้นตอนการดำเนินงาน

1.5.1 หาข้อมูลอย่างกว้าง

1) ศึกษา Image processing ที่ใช้ Aforge.imaging
2) ศึกษาข้อมูลเกี่ยวกับ DirectCompute API
3) ศึกษาข้อมูลเกี่ยวกับ Parallel Algorithm ในการทำ Image Processing
4) ศึกษาข้อมูลอย่างคร่าวๆของ API อื่นๆเช่น OpenCL, CUDA C

1.5.2 หาข้อมูลแบบเจาะจง

1) ศึกษาภาษา HLSL (High-level shader language) เพื่อใช้ในการพัฒนาและหลักการเขียน
2) ศึกษาข้อมลูเกี่ยวกับ Image processing algorithm
3) ศึกษาข้อมูลเกี่ยวกับ Parallel compute image processing algorithm

1.5.3 Implementation

จัดทำ Image processing library ต่างๆ โดยใช้ DirectCompute API

1.5.4 Test & Debugging

ทดลองใช้งานจริงเพื่อทดลอง รวมทั้งตรวจสอบ แก้ไขส่วนที่ผิดพลาด และประเมินผล

1.5.5 Publishing

จัดทำเอกสารคู่มือการใช้ และรูปเล่มรายงาน

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ทฤษฎีเกี่ยวกับ DirectCompute

DirectCompute คือ API ของ Microsoft บน Windows Vista, Windows 7, Windows 8 ซึ่ง DirectCompute เป็นหนึ่งในชุดคำสั่งของ DirectX 11 ซึ่งจะมีคำสั่งเกี่ยวกับ vector math, parallel computing เป็น API ที่เทียบเคียงกับ OpenCL และ CUDA DirectCompute ถูกพัฒนาขึ้นด้วยภาษา HLSL ซึ่ง DirectCompute สามารถทำ high-speed general purpose และใช้ประโยชน์จาก parallel processor จำนวนมากบน GPU และด้วยความสามารถในการทำ memory sharing, thread synchronization ส่งผลให้การทำงานแบบขนานมีประสิทธิภาพเพิ่มขึ้น ซึ่งDirectCompute สามารถทำงานบนหลาย thread แบบขนานกันได้

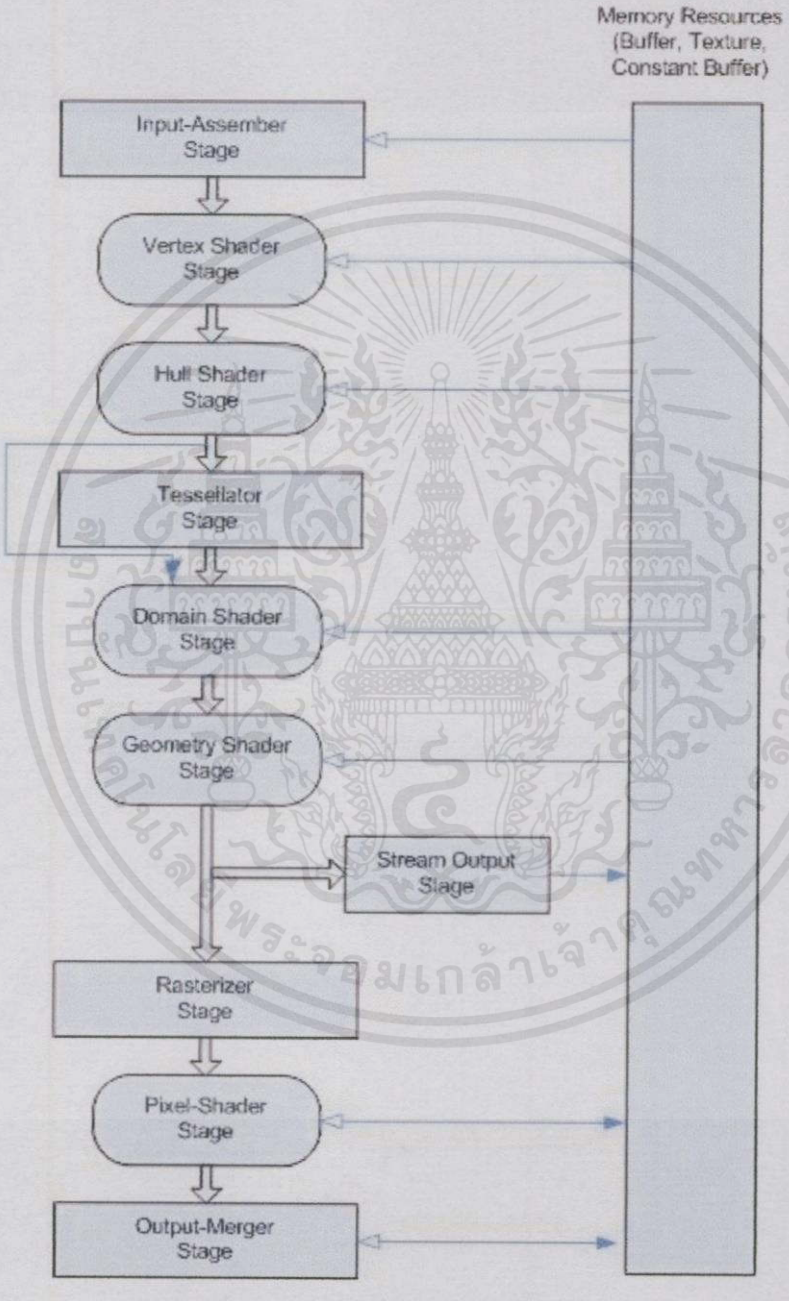
DirectCompute เป็น API ที่สนับสนุน GPU platform ทุกค่ายที่สนับสนุนระบบปฏิบัติการ Windows ซึ่ง DirectCompute การแบ่งงานออกเป็นหลายๆส่วนที่เป็นอิสระต่อกัน ออกจากกันทำให้สามารถประมวลผลได้พร้อมๆกัน โปรแกรมจะแบ่งงาน เป็น Group of Threads และจัดการแบ่งงานไปสู่ Thread Group เพื่อทำการประมวลผล นี้ดังรูป 2.1

รูป 2.1 DirectCompute Programing Model

โดย Thread Group เดียวกันจะทำงานพร้อมกัน และ Thread Group ต่างกันอาจทำงานพร้อมกัน

2.1.1 Graphics pipeline

Programmable pipeline ถูกออกแบบมาเพื่อการสร้างภาพกราฟฟิกสำหรับ real-time application ดังรูป 2.2



รูป 2.2 Flow diagram ของ graphic pipeline

ทุก stage ของ graphics pipeline สามารถควบคุมได้ แต่ละ Shader Stage สามารถควบคุมโดยภาษาHLSL ซึ่งทำให้ pipeline นี้ยืดหยุ่นและปรับแต่งได้ แต่ละ Stage มีหน้าที่ดังนี้

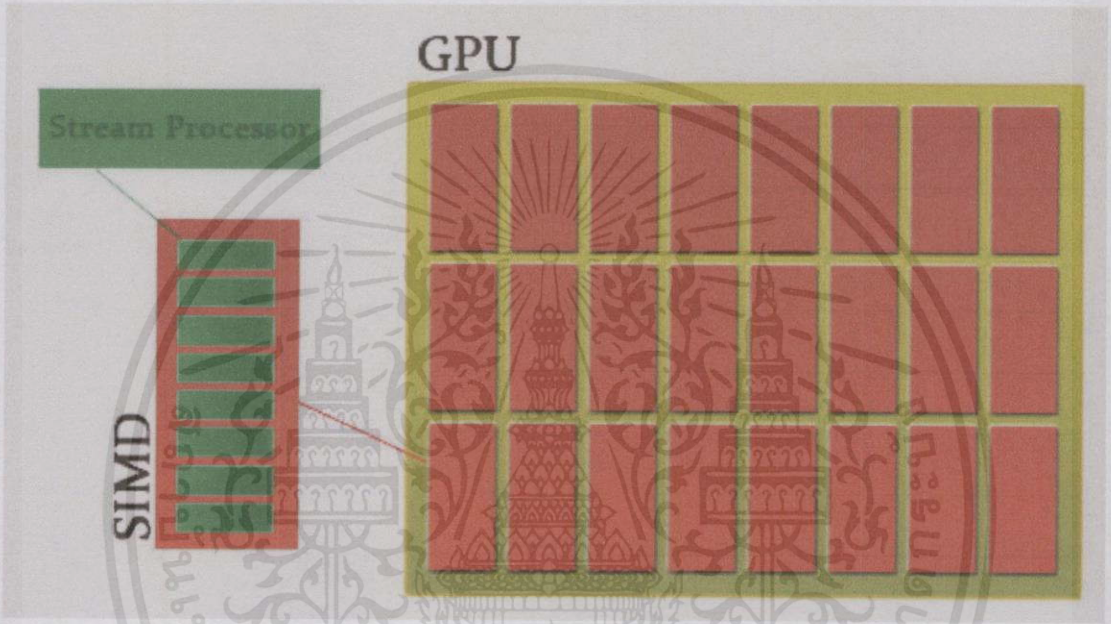
1) Input-Assembler Stage ทำหน้าที่ส่งข้อมูล (triangles, line, point) ให้กับ pipeline
2) Vertex-Shader Stage ทำกระบวนการเกี่ยวกับ vertices, transformations, skinning, lighting ซึ่งจะรับ single input และประมวลผลเป็น single output
3) Geometry-Shader Stage ทำกระบวนการเกี่ยวกับเรขาคณิตพื้นฐาน (3vertices สำหรับ triangle, 2 vertices สำหรับ line, 1 vertex สำหรับ point)
4) Stream-Output Stage ทำกระบวนการลำเลียงข้อมูลซึ่งข้อมูลอาจถูกลำเลียงผ่านไปยัง rasterizer/memory เพื่อนำไปประมวลผลใหม่อีกรอบได้
5) Rasterizer Stage ทำกระบวนการตัดข้อมูลบางส่วน และเตรียมข้อมูลและกำหนดสีให้Pixel-Shader Stage
6) Pixel-Shader Stage ทำกระบวนการรับข้อมูลที่ผ่านการเพิ่มเติมสีและสร้างข้อมูลของแต่ละ pixel
7) Output-Merger Stage ทำการรวมข้อมูลหลายประเภทของ Output (pixel shader values, depth, stencil information) เพื่อการ render
8) Hull-shader, tessellator, and domain-shader stages ทำการแปลง high-order surfac ไปเป็น triangles เพื่อการ render

2.1.1.1 Spectification ของ DirectCompute บน Direct3D 11.X

1) สามารถสร้าง thread ได้สูงสุด 1024 thread ต่อ group
2) สามารถสร้าง thread ใน dimension X, Y ได้สูงสุด 1024 group
3) สามารถสร้าง thread ใน dimension Z ได้สูงสุด 1024 group
4) Supports RWStructuredBuffers, RWByteAddressBuffers,RWTexture1D , RWTexture2D, RWTexture3D
5) Atomic instructions are available.
6) สนับสนุน Double-precision

2.1.2 GPU architecture

GPU มีโครงสร้างที่ต่างจาก CPU ซึ่ง GPU ถูกสร้างมาจาก processor จำนวนมากเรียกว่า stream processor แต่ละ processor สามารถถูกใช้ประมวลผล shader ได้ Stream processor ถูกสร้างให้อยู่ด้วยกันเป็นหน่วยเรียกว่า SIMD unit ซึ่งแต่ละ unit จะมี local data, cache, texture cache, fetch/decode unit เป็นของตัวเอง ดังรูป 2.3



รูป 2.3 โครงสร้างของ GPU

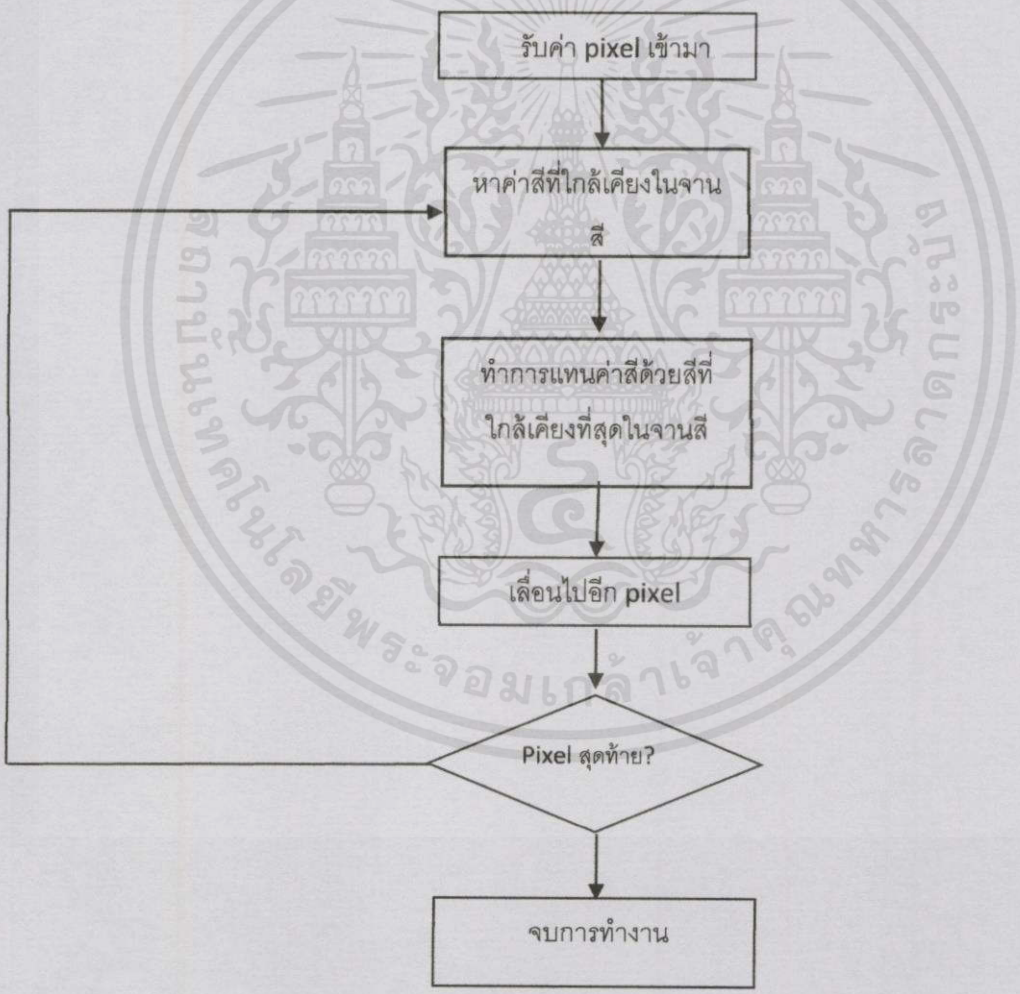
บน GPU มี SIMD processor หลายตัว แต่ละตัวจะถูกใช้ประมวลผล group of thread ซึ่งจะมีshared memory ใช้แบ่งปันข้อมูลระหว่าง thread ที่สามารถส่งไปที่ output buffer ได้ซึ่งจะลดจำนวน hardware เกี่ยวกับ scheduling, caching ได้เนื่องจาก shared memory ทำให้ลดจำนวนการติดต่อกับ external memory ได้และเนื่องด้วยจำนวน processor ที่มากนี้ทำให้ throughput มีจำนวนมากตามไปด้วย

2.2 รายละเอียดและรูปแบบมาตรฐานสำหรับแต่ละฟังก์ชั่นของ Image Processing

Image Processing คือ การประมวลภาพด้วยคอมพิวเตอร์ เพื่อจะให้ได้ข้อมูลที่ต้องการ โดยจะมีขั้นตอนการประมวลภาพดังนี้ เช่น การปรับภาพให้มีความคมชัดขึ้น การกรองสัญญาณรบกวนออกจากภาพซึ่งหลังจากผ่านกระบวนการประมวลผลภาพแล้วก็สามารถนำข้อมูลเหล่านั้นไปใช้ในการพัฒนาระบบ เพื่อใช้ประโยชน์ในด้านต่างๆได้โดยฟังก์ชันที่จะนำมาใช้ในโครงงานนี้มีทั้งหมดดังนี้

2.2.1 ฟังก์ชัน Color Reduction

เป็นฟังก์ชันที่จะทำการปรับลดสีของภาพโดยลดจำนวนจานสีที่ใช้ขั้นตอนการทำงานดังรูป 2.4



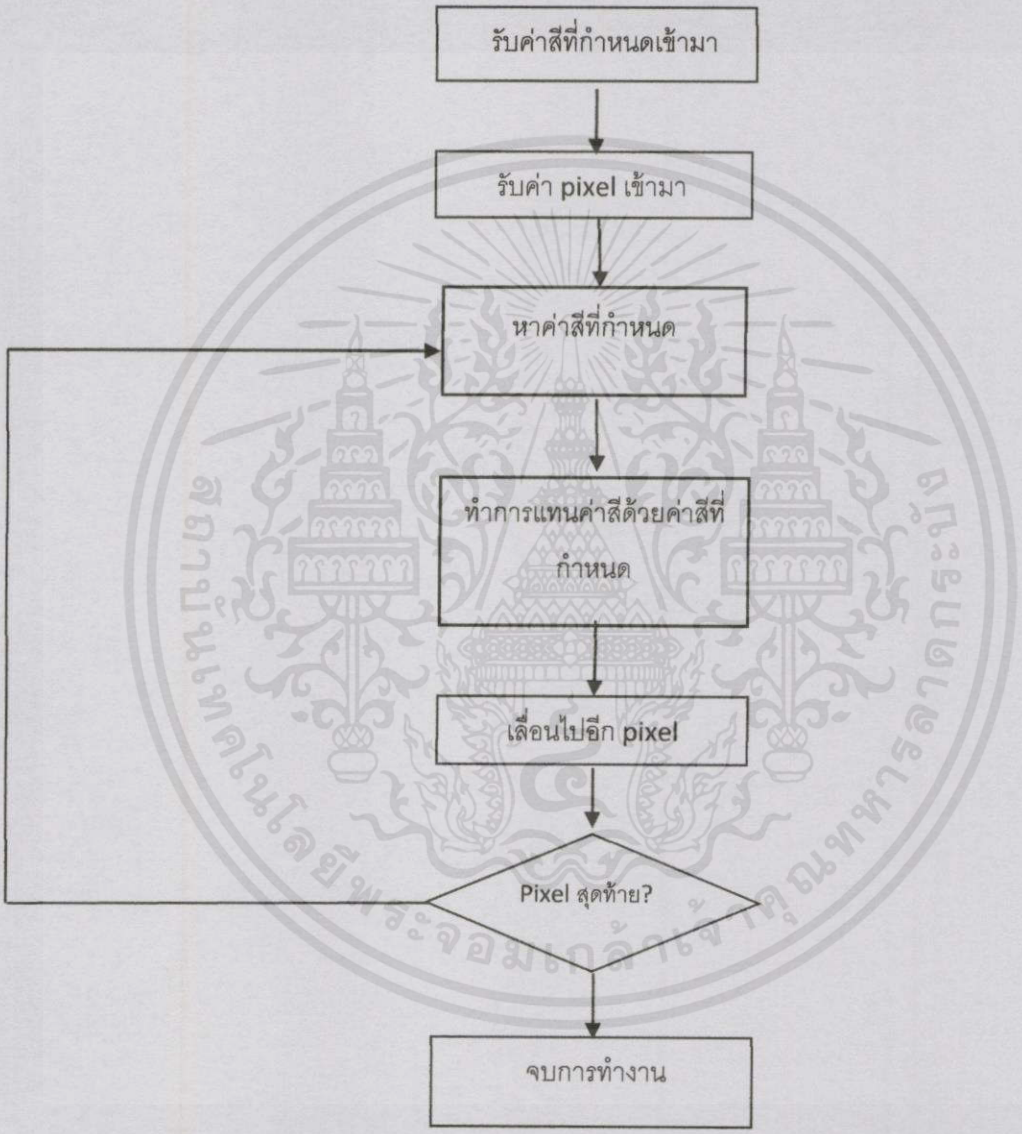
รูป 2.4 ขั้นตอนการทำงานของ color reduction

2.2.2 ฟังก์ชัน Linear Color Correction Filters

เป็นฟังก์ชันที่รวบรวม filters ที่ทำโดยวิธี linear correction ไว้ ซึ่งจะมีฟังก์ชัน ดังนี้

2.2.2.1 RGB Levels Linear Correction

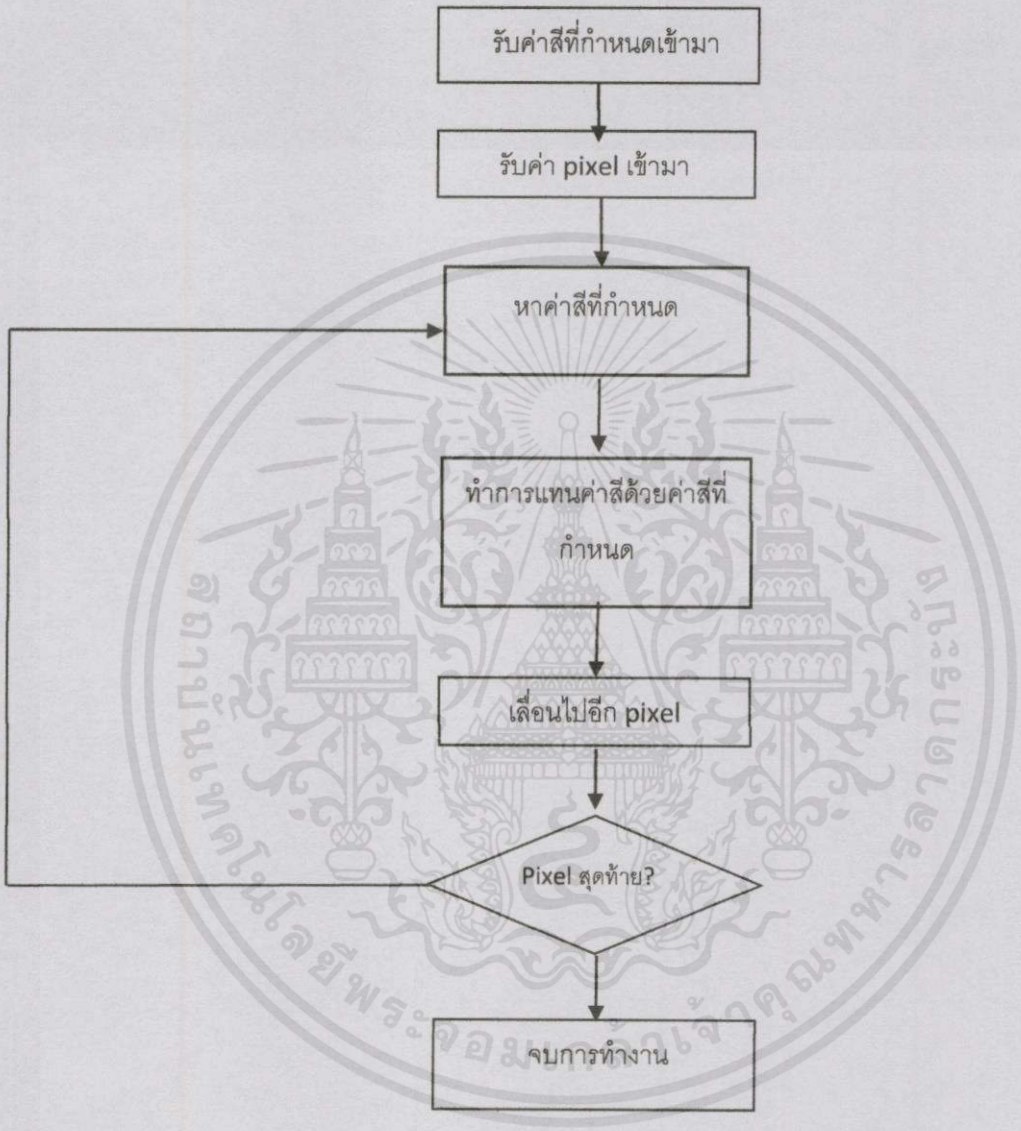
เป็นฟังก์ชันที่ทำการปรับค่าสี RGB ให้อยู่ในช่วงที่กำหนดขั้นตอนการทำงานดังรูป 2.5



รูป 2.5 ขั้นตอนทำงาน RGB Linear Correction

2.2.2.2 HSL Levels Linear Correction

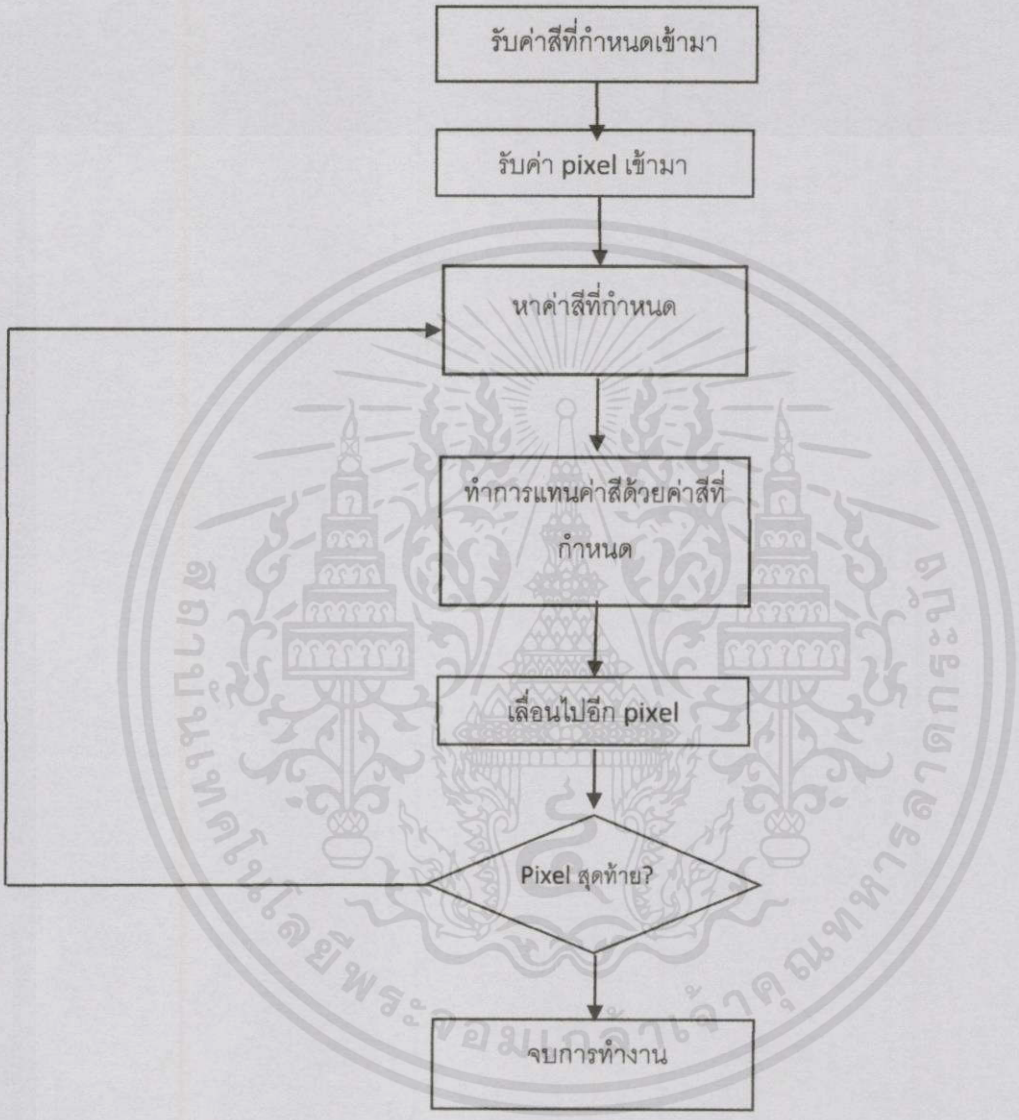
เป็นฟังก์ชันที่ทำการเปลี่ยน HSL ให้อยู่ในช่วงที่กำหนดขั้นตอนการทำงานดังรูป 2.6



รูป 2.6 ขั้นตอนทำงาน HSL LinearCorrection

2.2.2.3 YCbCr Levels Linear Correction

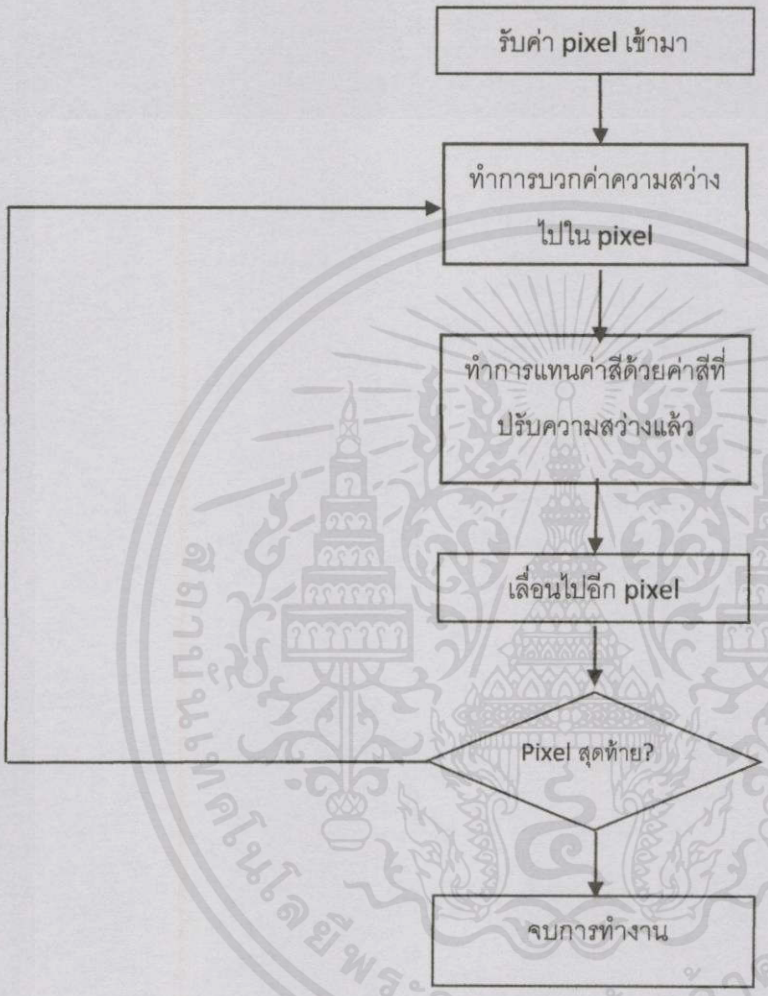
เป็นฟังก์ชันที่ทำการเปลี่ยน YCbCr ให้อยู่ในช่วงที่กำหนดขั้นตอนการทำงาน ดังรูป 2.7



รูป 2.7 ขั้นตอนทำงาน YCbCr Linear Correction

2.2.2.4 Brightness Correction

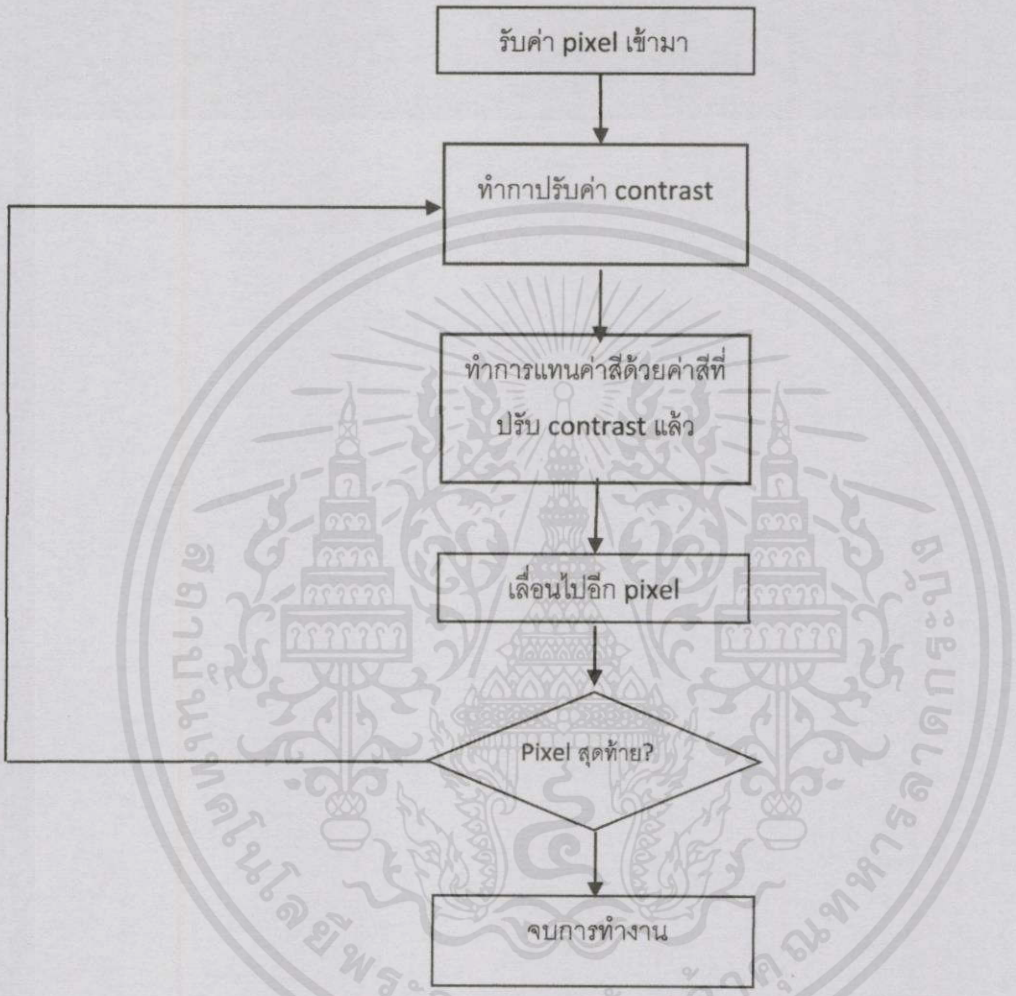
เป็นฟังก์ชันที่ทำการปรับเพิ่ม – ลด ค่าความสว่างของภาพขั้นตอนการทำงาน ดังรูป 2.8



รูป 2.8 ขั้นตอนการทำงาน Brightness Correction

2.2.2.5 Contrast Correction

เป็นฟังก์ชันที่ทำการปรับเพิ่ม – ลด ความค่า contrast ภาพ ขั้นตอนการทำงาน ดังรูป 2.9



รูป 2.9 ขั้นตอนการทำงาน Contrast Correction

2.2.2.6 Saturation Correction

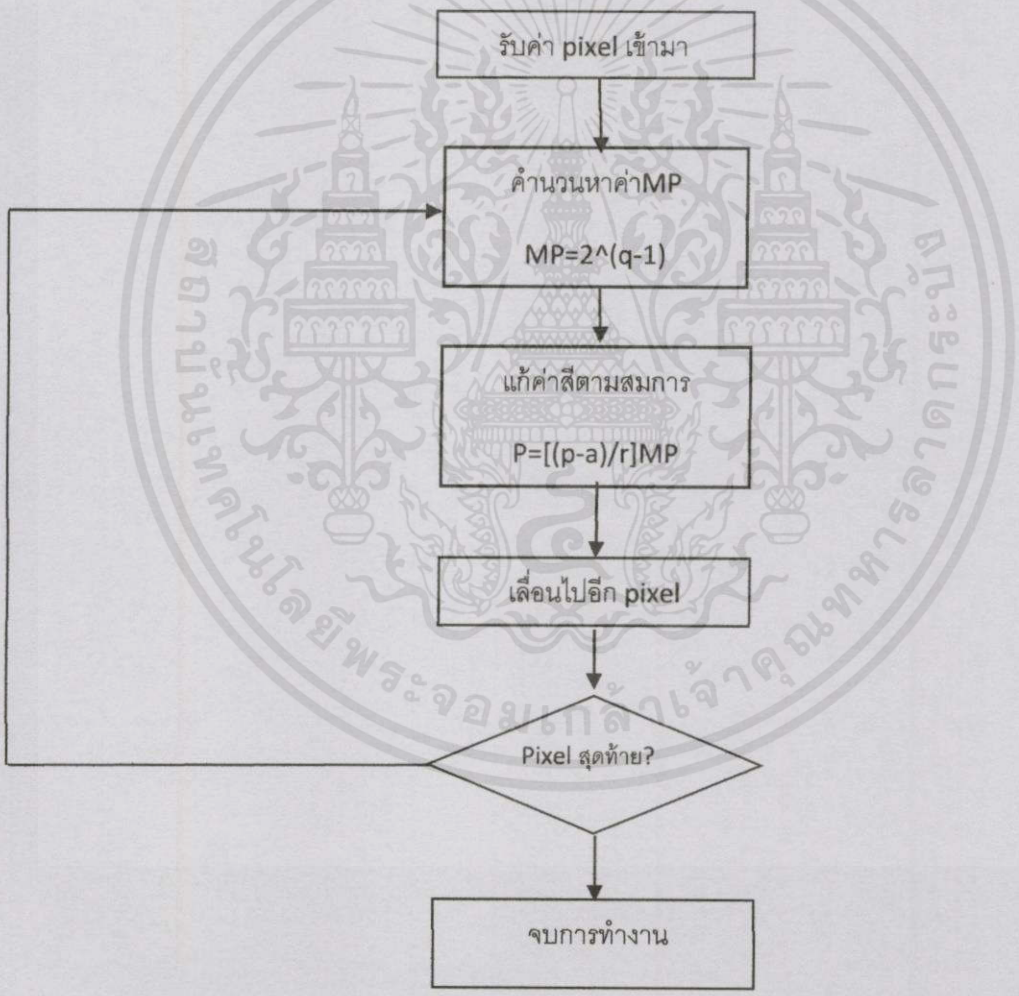
เป็นฟังก์ชันที่ทำการปรับเพิ่ม – ลด ความค่า saturation โดยทำการแปลงสีจาก RGBเป็น HSL และทำการปรับเพิ่ม-ลดค่า S

2.2.3 ฟังก์ชัน Non – Linear Color Correction

เป็นฟังก์ชันที่รวบรวม filters ที่ทำโดยวิธี non – linear correction ไว้ซึ่งจะมีฟังก์ชัน ดังนี้

2.2.3.1 Contrast Stretch

Contrast Stretch หรือ เรียกอีกอย่างว่า Normalization โดยเป็นการปรับค่า Contrast โดยใช้วิธี Stretching โดยดังรูป 2.10



รูป 2.10 Contrast stretch

2.2.3.2 Gamma Correction

ปรับภาพโดยใช้วิธีการ Gamma Correction มีวิธีการดังนี้ทำการหาค่า Gamma จาก สมการ 2.1 และ คำนวนสีใหม่จากสมการ 2.2

$$G = \frac{1}{\text{Gamma}} \tag{2.1}$$

$$\text{NewColor} = 255 * (\frac{\text{RGB}}{255})\text{^G} \tag{2.2}$$

2.2.4 ฟังก์ชัน Image Re-coloring Filter

เป็นฟังก์ชันที่รวบรวม filters ที่ในการเปลี่ยนสีภาพ

2.2.4.1 Grayscale

เป็นฟังก์ชันที่ทำการเปลี่ยนสีภาพเป็น grayscale ตามสมการ2.3

$$\text{Grayscale} = \frac{\text{Red+Green+Blue}}{3} \tag{2.3}$$

2.2.4.2 Sepia

เป็นการเปลี่ยนให้เป็นสี Sepiaซึ่งจะสีที่คล้ายสีน้ำตาลเก่าๆทำการเปลี่ยนเป็นสี sepia ทำได้ตามการ 2.4-2.6

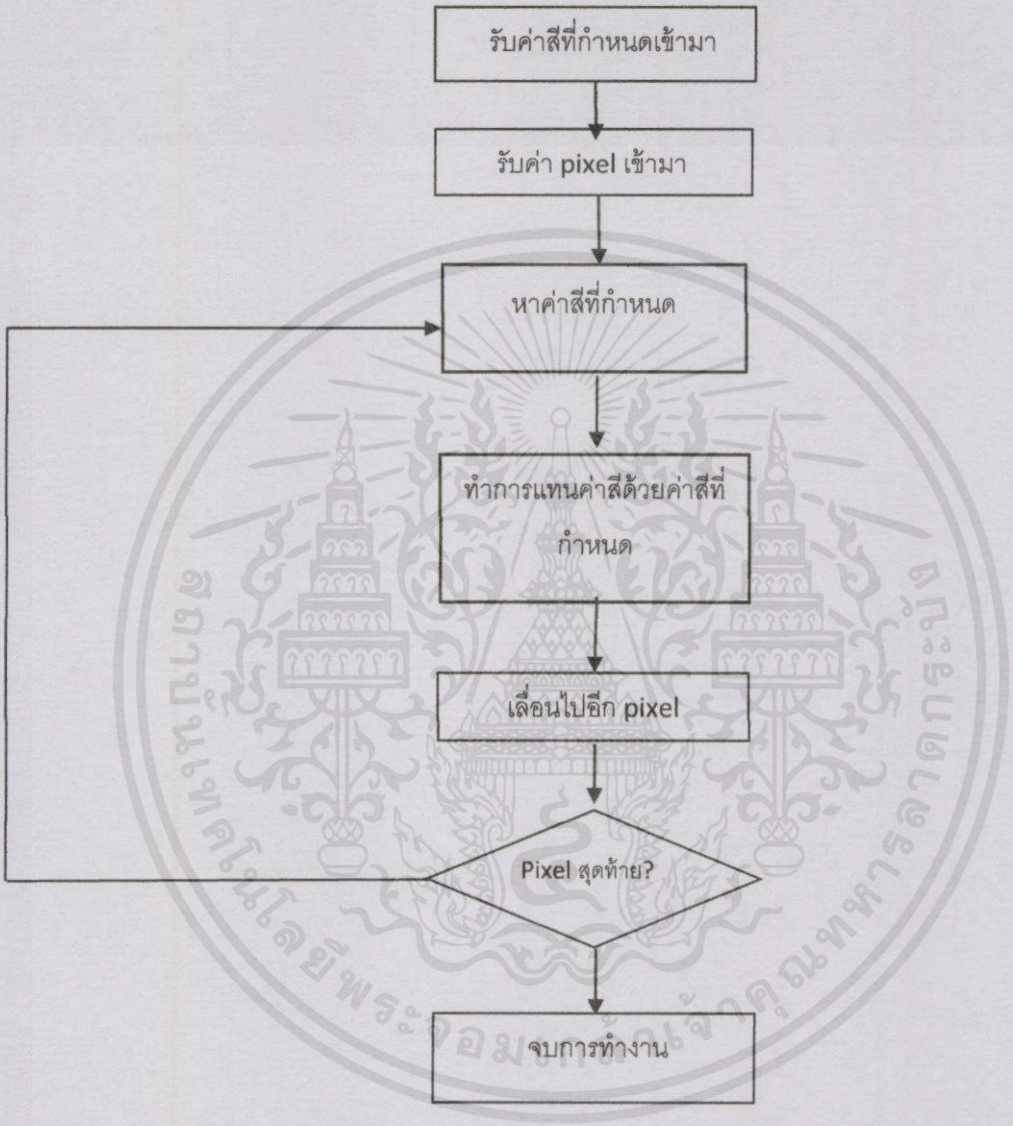
$$oRed = (iRed * 0.393) + (iGreen * 0.769) + (iBlue * 0.189) \tag{2.4}$$

$$oGreen = (iRed * 0.349) + (iGreen * 0.686) + (iBlue * 0.168) \tag{2.5}$$

$$oBlue = (iRed * 0.272) + (iGreen * 0.534) + (iBlue * 0.131) \tag{2.6}$$

2.2.4.3 Hue Modifier

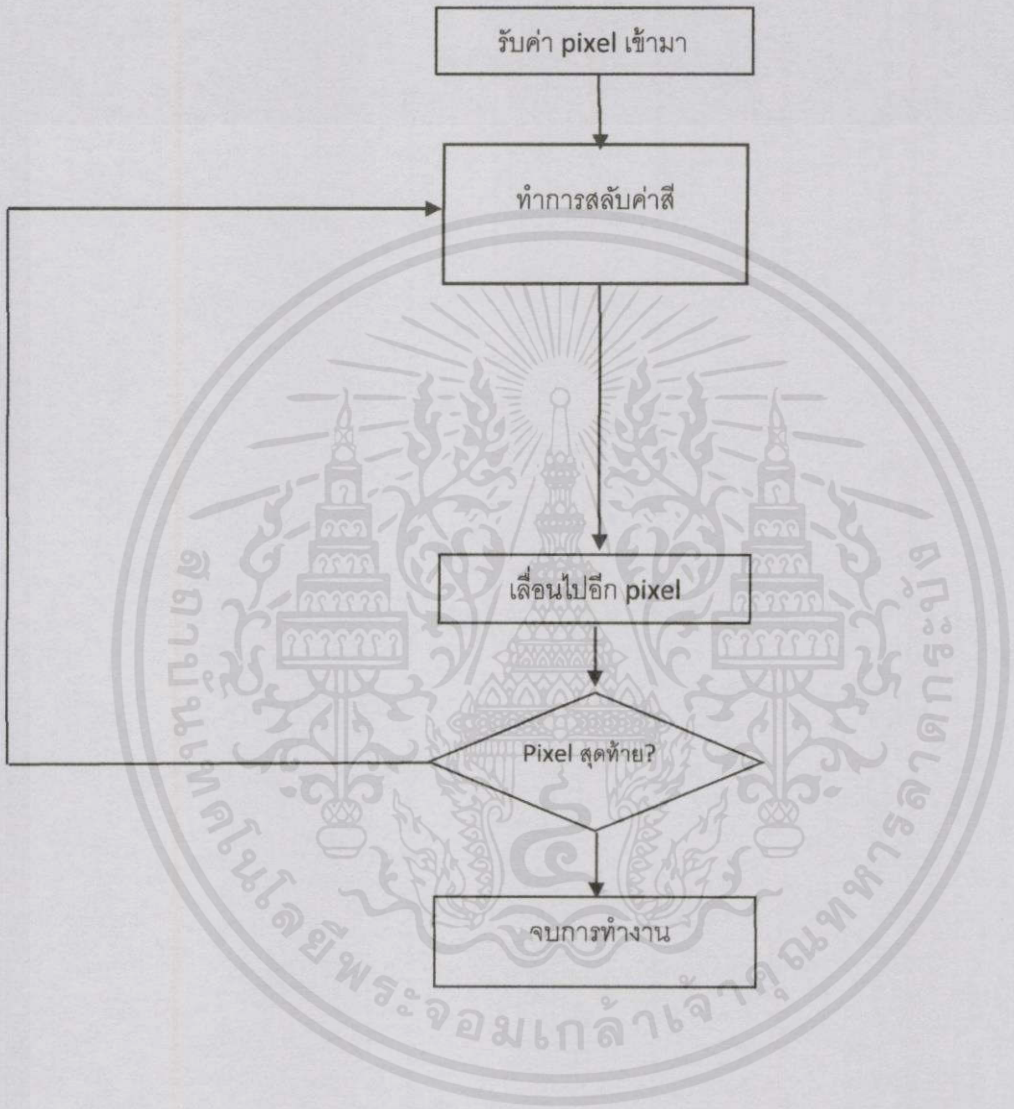
เป็นการทำงานที่แม่สี HSL โดยจะทำการเปลี่ยนเฉพาะค่า Hue โดยที่ค่า Luminance กับ ค่า Saturation จะไม่เปลี่ยน ขั้นตอนการทำงานดังรูป 2.11



รูป 2.11 ขั้นตอนการทำงานของ Hue modifier

2.2.4.4 Rotate Channel

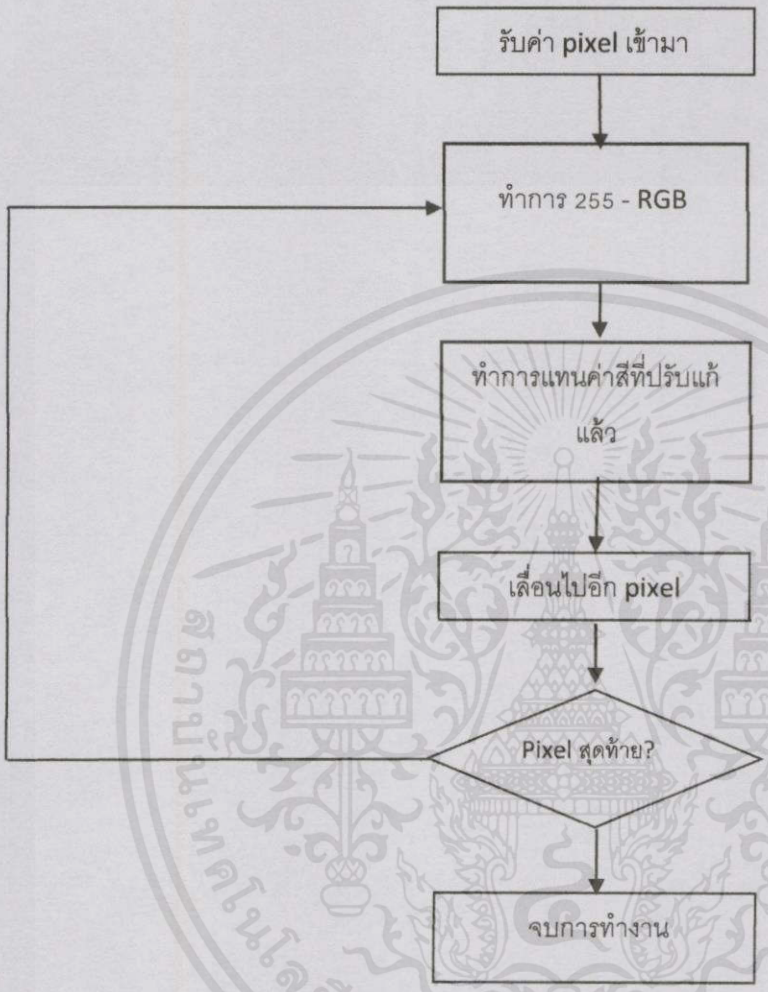
เป็นการสลับค่าสีให้ Red แทน Green , Green แทน Blue และ Blue แทน Red ขั้นตอนการทำงานดังรูป 2.12



รูป 2.12 ขั้นตอนการทำงานของ rotate channel

2.2.4.5 Invert

เป็นการ Invert ค่าสีของภาพ ขั้นตอนการทำงานดังรูป 2.13



รูป 2.13 ขั้นตอนของการทำงาน Invert

2.2.5 Convolution Filters

เป็น filters ที่ใช้การ convolution ในการกรองโดยมีสูตรตามสมการ 2.7

$$V = \sum_{i=1}^{n}(\sum_{j=1}^{n} f(i,j)d(i,j)) \quad (2.7)$$

โดยที่ f(i,j) = ค่าในหน้าต่างที่ตำแหน่ง i,j

d(i,j) = ค่าของภาพที่ตำแหน่ง i,j

F = ผลรวมของค่าทุกค่าในหน้าต่าง แต่จะเป็น 1 เมื่อผลรวมเป็น 0

V = ผลของการทำ convolution ถ้าน้อยกว่า 0 ให้ปรับเป็น 0

2.2.5.1 Convolution

เป็นการทำ Convolution

2.2.5.2 Sharpen

เป็นการทำ Convolution โดยใช้หน้าต่างที่เหมาะสมเพื่อทำให้ภาพมีความชัดมากขึ้นจะมีสมการเหมือน convolution ทุกอย่าง แต่ใช้ หน้าต่าง K = [0,-1,0,-1,5,-1,0,-1,0]

2.2.5.3 Edge

เป็นการทำ Convolution โดยใช้หน้าต่างที่เหมาะสม เพื่อกรองภาพให้เหลือแต่ขอบจะมีสมการเหมือน convolution ทุกอย่าง แต่ใช้ หน้าต่าง K = [0,-1,0,-1,4,-1,0,-1,0]

2.2.5.4 Sobel Edge

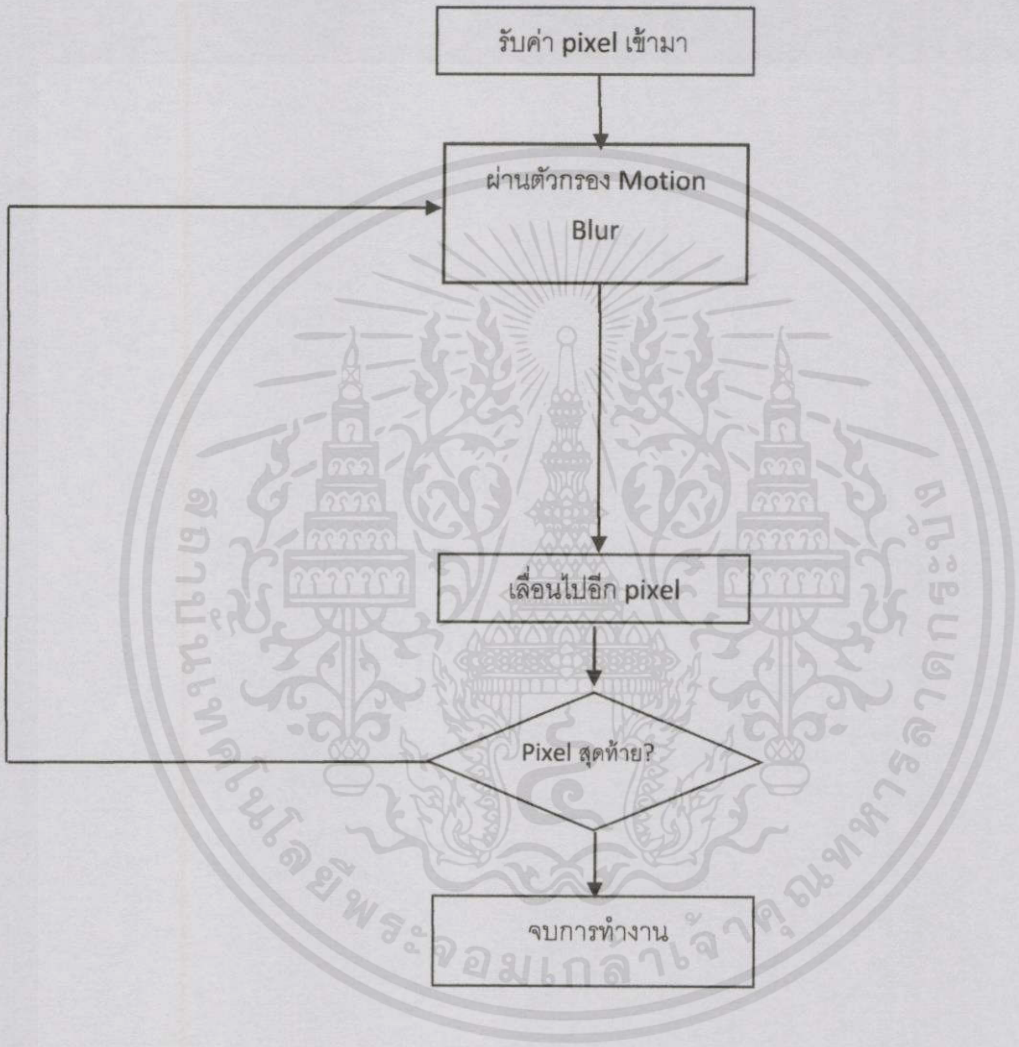
เป็นการทำ Convolution โดยใช้หน้าต่างที่คำนวณจากสมการ |G| = |Gx|+|Gy| โดย Gx =[-1,0,1,-2,0,2,-1,0,1] Gy = [1,2,1,0,0,0,-1,-2,-1]

2.2.6 Blur

เป็นการทำ Convolution โดยใช้หน้าต่างที่เหมาะสม เพื่อทำให้ภาพมีความชัดน้อยลง

2.2.6.1 Motion Blur

ปรับภาพให้เสมือนเคลื่อนไหว ดังรูป 2.14



รูป 2.14 Motion Blur

2.2.6.2 Gaussian Blur

เป็นการทำ Convolution โดยใช้หน้าต่างที่คำนวณจากสมการ Guassian 2 มิติ ซึ่งตามสมการ 2.8

$$g(u,v) = \left(\frac{1}{2\pi\sigma^2}\right) e^{\left(-\frac{(u^2+v^2)}{2\sigma^2}\right)} \quad (2.8)$$

โดย σ คือ ค่า sigma

2.2.7 Median Filter

เป็นฟังก์ชันที่ลดการกวนสัญญาณของภาพ ตามสมการ 2.9

$$\text{Median}(A) = \text{Median}[A((x+i),(y+j))] \quad (2.9)$$

โดยที่ x+i , y+j เป็นค่าที่อยู่บน Image A

i ,j เป็นค่าบนหน้าต่าง M

M คือ จำนวน pixel ที่อยู่ในการคำนวณ Median Culculation

2.2.8 Noise Reduction

เป็นฟังก์ชันที่ทำการลดสัญญาณรบกวนในภาพลง

2.2.8.1 Mean Filter

เป็นการลดสัญญาณบกวนของภาพโดยใช้วิธีหาค่าเฉลี่ยโดยใช้หน้าต่าง K = [1,1,1,1,1,1,1,1,1]

2.2.8.2 ConservativeSmoothing

จะเป็นการลดสัญญาณรบกวนของภาพ โดยทำตามขั้นตอนนี้ทำการหาค่า Minimum และ Maximum สุดรอบจุดกลางภาพ แล้วเปรียบเทียบค่าที่จุดกลางภาพกับค่า Minimum และ Maximumถ้ามากกว่า Maximum ให้แทนค่าจุดกลางด้วย ค่า Maximum หรือ ถ้าน้อยกว่า Minimum ให้แทนด้วยค่า Minimum

2.2.9 Texturer Filter

เป็น filters ที่ทำการนำ texture ที่ได้จาก texture generator มาใช้งานกับภาพจริง

2.2.9.1 Texturer

เป็นการนำ texture ที่มีมาใช้กับรูปภาพ ตามสมการ 2.12

$$dst = src * PreserveLevel + (src * FilterLevel * texturevalue) \quad (2.10)$$

โดยที่ src คือ ค่า pixel ในภาพ

PreserveLevel คือ ส่วนของภาพที่ได้จากการกรองปกติตั้ง ไว้ที่ 0.5

FilterLevel คือ ส่วนของภาพที่ทำการคูณกับค่าของ Texture ปกติตั้งไว้ที่0.5

Texturevalue คือ ค่าของ texture

2.2.9.2 Textured Merge

เป็นนำรูปภาพต้นฉบับกับรูปภาพ overlay มา merge รวมเข้ากับ texture ตามสมการ 2.13

$$dst = src * textureValue + ovr * (1.0 - textureValue) \quad (2.11)$$

โดยที่ src คือ ค่า pixel ในภาพ

Texturevalue คือ ค่าของ texture

ovr คือ ค่า pixel ใน overlayimage

2.2.9.3 Textured Filter

เป็นการ merge ภาพต้นฉบับรวมกับหลาย texture ได้ตามสมการนี้

$$dst = FilterLevel * (src1 * textureValue + src * (1.0 - textureValue)) + PreserveLevel * src2 \quad (2.12)$$

โดยที่ src1 คือ ค่า pixel ในภาพที่สร้างด้วย filter1

src2 คือ ค่า pixel ในภาพที่สร้างด้วย filter2

PreserveLevel คือ ส่วนของภาพที่ได้จากการกรองปกติตั้ง ไว้ที่ 0.5

FilterLevel คือ ส่วนของภาพที่ทำการคูณกับค่าของ Texture ปกติตั้งไว้ที่0.5

Texturevalue คือ ค่าของ texture

2.3 Parallel-Image processing

เนื่องจากข้อมูลที่มากของรูปภาพความละเอียดสูง (อาจมีขนาดถึง 10,000 x 10,000 pixel) โดยทั่วไปแล้วหากทำการประมวลผลแบบ sequential computing จะทำให้เกิดปัญหา Bottleneck ขึ้นในกระบวนการประมวลผลซึ่งสามารถแก้ได้โดยแบ่งปัญหาออกเป็นหลายๆส่วน และแยกข้อมูลเป็น local data เพื่อประมวลผลแบบ parallel computing ซึ่งแต่ละ processor จะทำงานด้วย input data จาก shared memory Algorithm ของ image processing ส่วนใหญ่ใน low-level, mid-level image processing คือData Distributionในหลายๆ image processing algorithm แสดงให้เห็นว่า แต่ละ output pixel ถูกคำนวณโดยไม่ขึ้นกับ pixel อื่นๆของ input image กล่าวได้ว่า ค่าของ output ไม่ขึ้นอยู่กับ pixel อื่นๆ ซึ่งทำให้เราสามารถคำนวนแต่ละ pixel ได้ในแบบ parallel แต่หากมี algorithm บางส่วนที่เป็น data dependency จึงจำเป็นที่จะต้องมีการวิเคราะห์ เช่น mutual exclusion(race condition) หากมี 2 thread ต้องการที่จะเพิ่มค่าของ global variable ไป 1 ดังรูป 2.15

Thread 1	Thread 2		Integer value
			0
read value		←	0
increase value			0
write back		→	1
	read value	←	1
	increase value		1
	write back	→	2

รูป 2.15 ทำงาน 2 thread แบบ sequential

ค่าที่ได้จะเป็น 2 ตามที่คาดการณ์ไว้ แต่หากถ้าทั้ง 2 thread ทำงานแบบ parallel โดยไม่มีการ synchronize ผลลัพธ์ที่ได้จะผิดเพี้ยนดังรูป 2.16 ซึ่งแก้ปัญหาได้โดยการใช้ Locking/Synchronization ซึ่งบางครั้ง OS จะจัดการให้ใน ส่วนนี้

Thread 1	Thread 2		Integer value
			0
read value		←	0
	read value	←	0
increase value			0
	increase value		0
write back		→	1
	write back	→	1

รูป 2.16 การทำงาน 2 thread แบบ parallel โดยไม่มี synchronization

ผลลัพธ์ที่ได้เป็น 1 แทนที่จะได้ 2 แสดงให้เห็นถึงว่าการทำงานในรูปที่ 2.16 ไม่ได้เป็น mutually exclusive Mutually exclusive คือ operation ที่ไม่สามารถถูก interrupt ได้ ในขณะที่ข้อมูลเดียวกันกำลังถูกใช้งานหรือเข้าถึงอยู่

2.4 เครื่องมือสำหรับการพัฒนา

2.4.1 SharpDX framework เลือกใช้ตาม feature ดังนี้

1) สามารถ compile ได้ทั้ง 32, 64 bit system

2) ประกอบไปด้วย common class ของ API ที่ต้องการใช้

3) เป็น .NET ทำให้ implement ได้ด้วย C#

2.4.2 Visual Studio compiler

เลือกตามความสามารถที่ระบบต้องการซึ่ง Visual studio มีตามที่โครงงานนี้ต้องการที่จะใช้

2.4.3 Supported DirectX11 GPU driver

2.4.4 Supported DirecX11 GPU

บทที่ 3
การพัฒนาและส่วนประกอบ

3.1 ภาพรวมของระบบ

โครงงานนี้ต้องการที่จะพัฒนา Parallel compute image processing library ด้วย DirectCompute เพื่อเป็นหนึ่งในทางเลือกในการเลือกใช้ Image processing library ซึ่ง DirectCompute สามารถใช้ได้กับทุก platform graphic card ทั้ง Nvidia, ATI, Intel HD series โดย เมื่อจบโครงงานจะได้เป็น Parallel Image Processing Library พร้อมกับ ซอฟต์แวรทดสอบ ประสิทธิภาพ

3.2 องค์ประกอบของระบบ

Parallel compute image processing library ที่จะจัดทำขึ้นนั้น ประกอบไปด้วย function พื้นฐานของ image processing ดังนี้

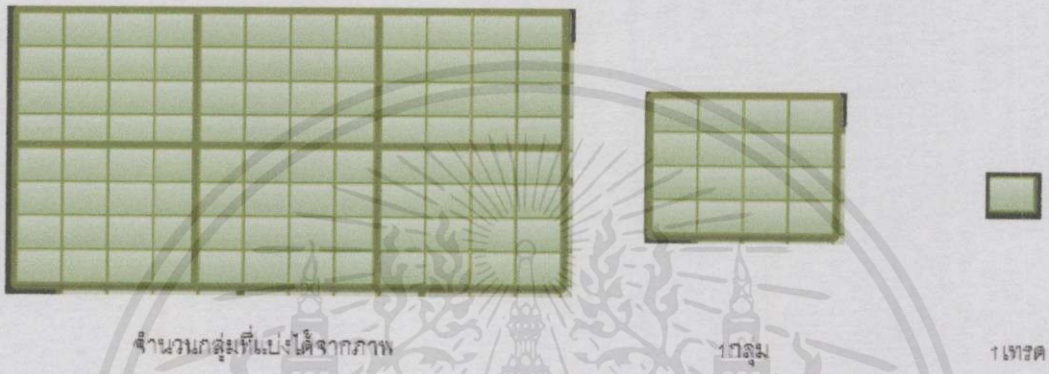
1.) Color reduction
2.) Linear color filter
3.) Non – linear color filter
4.) Image re – color
5.) Convolution filter
6.) Gaussian blur
7.) Motion blur
8.) Median filter
9.) Noise reduction

และโปรแกรมทดสอบ ประสิทธิภาพของ library ที่พัฒนาขึ้นมา เพื่อเปรียบเทียบประสิทธิภาพกับ library ต้นแบบ (Aforge) และ เทียบกับ library ที่เขียนบน CUDA

3.3 ตัวอย่างการออกแบบและวิเคราะห์ parallel algorithm

3.3.1 แนวคิดการประมวลผลของแบบขนานของ DirectCompute

โดย Directcompute นั้นจะมีแนวคิดในการแบ่งงานด้วยวิธีการแบ่งจำนวนเทรดในแกน X ,Y ไปหารจำนวนพิกเซลของภาพเพื่อแบ่งออกเป็นกลุ่ม โดยในแต่ละกลุ่มนั้นก็จะมีจำนวนเทรดตามที่แบ่งไปตามรูป3.1



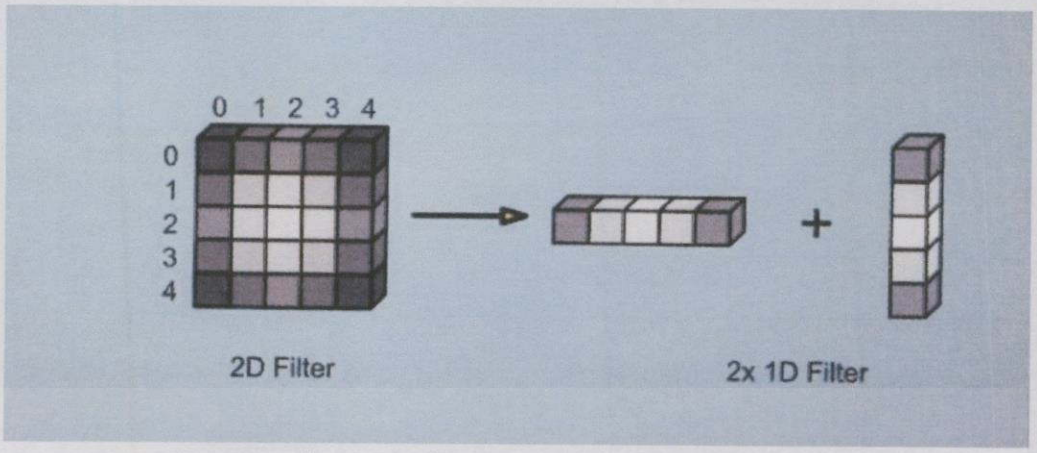
รูป 3.1 การแบ่งงานของ DirectCompute

เช่น ภาพขนาด 640*480 ให้มี เทรดแกน X,Y จำนวนละ32 เทรด ก็จะแบ่งได้เป็น 20*15 = 300 กลุ่ม โดยแนวคิดสามารถนำไปใช้กับฟังก์ชั่นที่ไม่ต้องใช้ kernel ในการคำนวนได้ เช่น Grayscale , Brightness , Sepia Tone เป็นต้น

3.3.2 แนวคิดการประมวลผลของฟังก์ชั่นที่ต้องใช้ Kernel

โดยปกติฟังก์ชั่นจำพวก convolution จะมี kernel ในการนำมาประมวล เช่น ฟิลเตอร์มีขนาด m*n และภาพมีขนาด p*q การคำนวนที่ต้องดำเนินการก็จะมีขนาดเท่ากับ p*q*m*n ซึ่งเมื่อใช้หลักการการคำนวนในขั้นแรกก็จะเป็น p*m*n ขั้นต่อมาก็จะเป็น q*m*n ทำให้ขนาดของการคำนวนที่ต้องดำเนินการเหลือเพียงแค่ (p+q)*m*n โดยวิธีนี้จะมีประสิทธิภาพอย่างมากเมื่อเป็นภาพขนาดใหญ่

วิธีแบ่งจะเป็นตามรูป 3.2



รูป 3.2 การแปลง 2d filter เป็น 2*1d filter

3.4 ขั้นตอนการพัฒนา

1) ศึกษา Image processing algorithm ขั้นตอนการทำงานและผลลัพธ์หลังจากเสร็จกระบวนการ
2) ศึกษา Parallel algorithm ขั้นตอนการทำงาน การที่จะปรับจากการประมวลผลจาก Sequential ไปสู่ parallel
3) ศึกษา DirectCompute โครงสร้าง Feature, Input, Output, Buffer, Graphic pipeline, Thread management
4) ศึกษา class library ของ SharpDX วิธีการเรียกใช้ Function ในแต่ละ classจำเป็นต่อการพัฒนา
5) พัฒนาฟังก์ชั่นทาง image processing
6) เปรียบเทียบผลลัพธ์ และความเร็วระหว่าง ฟังก์ชั่นทาง image processing ที่พัฒนาขึ้นกับ Aforge และ Cuda

3.5 ตัวอย่างการสร้าง Resource และ การเขียน HLSL เบื้องต้น

ในการเรียกใช้ SharpDX นั้นจะต้องทำการประการเพิ่ม dll เข้าไปใน reference ของโปรเจคด้วย

โปรแกรม 3.1 ประกาศใช้ dll ของ SharpDX

```
using SharpDX;
using SharpDX.D3DCompiler;
using SharpDX.Direct3D;
using SharpDX.Direct3D11;
using SharpDX.DXGI;
```

using SharpDX เป็นการประกาศเพื่อเรียกฟังก์ชั่นต่างๆของ SharpDX

using SharpDX.D3DCompiler เป็นการประกาศเพื่อเรียกงานฟังก์ชั่นต่างๆของ D3DCompiler ซึ่งมีการใช้งานฟังก์ชั่นในการ compile ภาษา HLSL

using SharpDX.Direct3D เป็นการเป็นการประกาศเพื่อเรียกงานฟังก์ชั่นต่างๆของ Direct3D

using SharpDX.Direct3D11 เป็นการเป็นการประกาศเพื่อเรียกงานฟังก์ชั่นต่างๆของ Direct3D11 เพื่อเรียกใช้งานพวก Device , Shader Resource View ,Unordered Access View และ Texture2D

using SharpDX.Direct3D11 เป็นการเป็นการประกาศเพื่อเรียกงานฟังก์ชั่นต่างๆของ DXGI เพื่อเรียกงานใช้ SwapChain

โปรแกรม 3.2 การสร้าง Device และ SwapChain เบื้องต้น

```
var device = Device.CreateWithSwapChain(
    DriverType driverType,
    DeviceCreationFlags flags,
    SwapChainDescription swapChainDescription,
    out Device device,
    out SwapChain swapChain);
```

เป็นการสร้าง Device พร้อมกับการสร้าง SwapChain เพื่อใช้ในการ render

DriverType เป็นการเลือกว่าจะใช้ driver type ไหน ซึ่งถ้าเลือก hardware ก็จะทำการดำเนินการเกี่ยวกับ Direct3D ส่วนใหญ่โดยใช้ hardware และส่วนที่ pipelines ไม่สนับสนุนก็จะใช้ software

DeviceCreationFlags เป็นการเลือกว่าจะสร้างขึ้นเพื่อจุดประสงค์ไหน ถ้า Debug ก็ใช้สำหรับ debug

ซึ่งผลลัพธ์ก็จะได้เป็น Device กับ SwapChain

โปรแกรม 3.3 การสร้าง Texture2D , Shader Resource View และ Unordered Acces View เบื้องต้น

```
var textureView = ShaderResourceView.FromFile("PathFile");
var texturevt = textureView.ResourceAs<Texture2D>();
var descp = texturevt.Description;
descp.BindFlags = BindFlags.UnorderedAccess |
BindFlags.RenderTarget;
var texture = new Texture2D(device, descp);
var targetUAV = new UnorderedAccessView(device, texture);
```

เป็นการสร้าง Shader Resource View จากไฟล์ และให้เป็น resource กับ Texture2D โดย descp จะเป็นตัวบ่งบอกลักษณะของ Texture2D ว่าทำไรได้บ้าง ขึ้นอยู่กับที่ BindFlags ไว้สร้าง Unordered Access View ให้เข้าถึง Resource ของ Texture2D ได้

โปรแกรม 3.4 การเขียน HLSL เบื้องต้น

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define THREADSX
#define THREADSY
[numthreads(THREADSX, THREADSY, 1)]
void Function(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
}
```

รับอินพุตเข้าเป็น Texture2D ซึ่งรับมาจาก register ตำแหน่งที่ t0 โดย t หมายถึงเป็นตัวแปร texture หรือ texture buffer เอ้าท์พุทเป็น Texture2D โดยให้ไปเก็บที่ register ของ UAV ซึ่งคือ u0 สามารถกำหนด Thread ได้ด้วยการ Define ได้เลยเ วลาสร้างฟังก์ชั่นจะต้องประกาศตัวแปรที่เกี่ยวกับการแบ่งงานไว้ด้วยโดยสามารถประกาศได้ตรงๆลย คือ uint ชื่อ : SV_GroupIndex , uint3 ชื่อ : SV_GroupID , uint3 ชื่อ : SV_GroupThreadID และ uint3 ชื่อ : SV_dispatchThreadID

บทที่ 4

การทดลอง

4.1 ฟังก์ชั่นที่ทำการทดลอง

โดยฟังก์ชั่นที่จะนำมาทำการทดลองก็จะมีทั้งหมด ดังนี้

1) Color reduction
2) Linear color filter
3) Non – linear color filter
4) Image re – color
5) Convolution filter
6) Gaussian blur
7) Motion blur
8) Median filter
9) Noise reduction

4.2 สภาพแวดล้อมในการทดลอง

ในการทดลองนี้มีการจัดการตัวแปรที่เกี่ยวข้องกับการทดลอง ดังนี้

4.2.1 คอมพิวเตอร์ที่ใช้ในการประมวลผล

คอมพิวเตอร์ที่ใช้ในการประมวลผลมีรายละเอียด ดังนี้

1) CPU : Intel® Core™ i7-4790 3.60 GHz
2) VGA : NVIDIA GeForce GTX 980
3) RAM : DDR III 16 GB Bus 1600

4.2.2 ขนาดของภาพที่ใช้ทำการทดลอง

ขนาดของภาพที่ใช้ในการทดลองมีขนาด ดังนี้

1) 1920*1080 (Full HD) จำนวน 2 รูป
2) 3840*2160 (UHD) จำนวน 2 รูป
3) 7680*4120 (8K HD) จำนวน 2 รูป

4.2.3 จำนวนของเทรดที่ใช้ในการทดลอง

จำนวนของที่ใช้ในการทดลองจะแบ่งเป็น 3 ระดับ ดังนี้

1) จำนวนเทรดเท่ากับ 1

2) จำนวนเทรดเท่ากับ 16

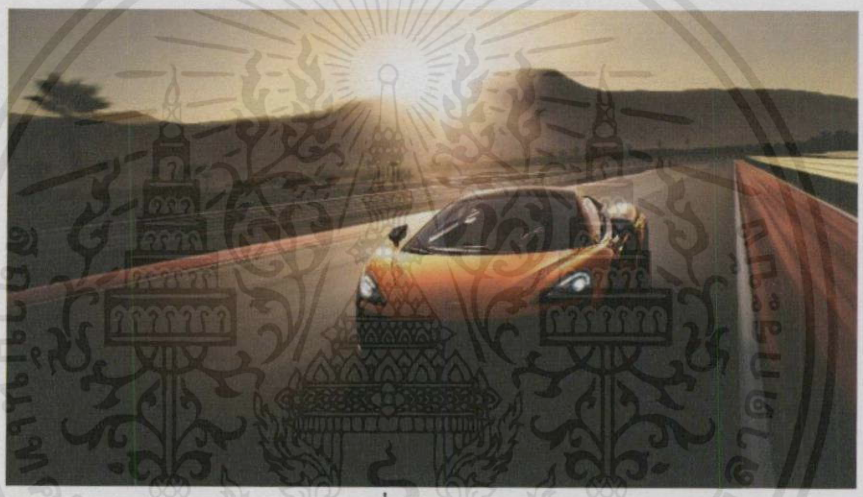
3) จำนวนเทรดเท่ากับ 32

4.2.4 ผลการทดลอง

4.2.4.1 ภาพขนาด 1920*1080 (Full HD)

4.2.4.1.1 จำนวนเทรดเท่ากับ 1

4.2.4.1.1.1 รูปที่1



รูป4.1 รูปที่ใช้ในการทดลอง

ตาราง 4.1 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	100.40	78.00	22.00	29.20	25.33	3.21	8.24
2	109.24	81.00	13.00	30.44	26.74	2.96	3.17
3	104.91	91.00	13.00	30.11	26.13	3.28	3.19
Sepia							
1	108.01	91.00	17.00	n/a	n/a	n/a	14.93
2	102.37	88.00	14.00	n/a	n/a	n/a	12.32
3	109.09	86.00	14.00	n/a	n/a	n/a	12.42

ตาราง 4.1 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Hue Modifier							
1	108.99	87.00	21.00	43.27	30.37	4.20	123.02
2	105.02	84.00	21.00	43.22	29.29	4.00	122.06
3	102.60	82.00	20.00	45.60	29.70	7.22	122.06
Rotate Channel							
1	101.36	85.00	16.00	51.58	36.18	8.88	4.59
2	98.51	85.00	13.00	35.67	25.70	5.97	4.04
3	94.31	80.00	14.00	37.07	27.14	6.02	4.05
Invert							
1	100.67	83.00	17.00	37.88	26.30	6.42	6.06
2	103.02	85.00	18.00	34.54	26.42	4.60	5.57
3	100.62	84.00	16.00	34.47	26.10	4.53	5.36
Color Reduction							
1	179.67	96.00	83.00	n/a	n/a	n/a	3056.77
2	156.93	94.00	62.00	n/a	n/a	n/a	3111.71
3	156.33	94.00	62.00	n/a	n/a	n/a	3081.56
Contrast Strecth							
1	102.54	86.00	16.00	n/a	n/a	n/a	11.06
2	123.31	107.00	16.00	n/a	n/a	n/a	3.85
3	107.74	84.00	16.00	n/a	n/a	n/a	3.87
Gamma Correction							
1	112.13	89.00	23.00	108.38	40.00	18.03	6.46
2	105.09	89.00	16.00	98.45	40.88	16.57	7.06
3	96.97	82.00	14.00	113.57	45.81	26.95	6.08
Brightness Correction							
1	117.50	96.00	21.00	34.76	26.82	3.86	8.58
2	116.19	99.00	17.00	34.25	26.24	3.90	7.09
3	100.24	85.00	15.00	34.67	26.47	3.81	7.04
Contrast Correction							
1	121.13	102.00	19.00	39.22	29.25	6.04	8.09
2	115.55	99.00	16.00	38.99	29.16	5.91	7.44
3	96.43	80.00	16.00	38.58	28.87	5.86	7.48

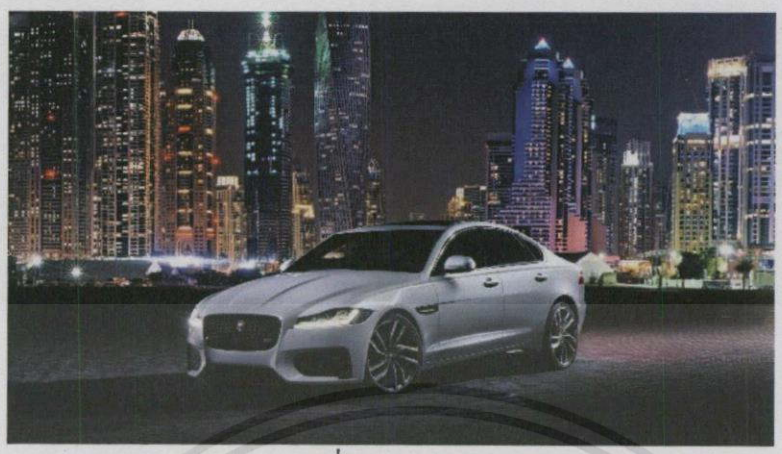
ตาราง 4.1 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Saturation Correction							
1	123.76	86.00	37.00	56.54	39.56	5.07	139.93
2	124.88	95.00	29.00	42.84	30.00	5.00	138.44
3	123.88	93.00	30.00	42.22	29.80	4.67	136.00
RGB Level Linear							
1	112.92	83.00	29.00	n/a	n/a	n/a	7.51
2	104.48	87.00	17.00	n/a	n/a	n/a	7.30
3	96.47	79.00	17.00	n/a	n/a	n/a	7.33
YCbCr Level Linear							
1	128.81	88.00	37.00	n/a	n/a	n/a	105.67
2	107.67	85.00	22.00	n/a	n/a	n/a	119.42
3	104.76	87.00	17.00	n/a	n/a	n/a	104.74
HSL Level Linear							
1	107.78	86.00	21.00	n/a	n/a	n/a	138.49
2	119.06	84.00	35.00	n/a	n/a	n/a	138.51
3	120.30	85.00	35.00	n/a	n/a	n/a	137.73
Blur							
1	170.34	86.00	84.00	216.31	136.26	38.84	428.32
2	177.90	93.00	84.00	101.11	39.01	20.11	426.86
3	183.85	97.00	86.00	129.15	41.77	45.29	429.38
Edge							
1	122.50	98.00	24.00	100.23	40.51	19.06	200.21
2	128.13	103.00	25.00	101.54	40.37	19.02	200.91
3	113.18	89.00	24.00	117.78	58.41	19.19	203.98
Sharpen							
1	127.93	91.00	30.00	107.11	41.10	24.08	191.94
2	118.23	90.00	28.00	105.31	42.10	20.89	192.86
3	118.87	90.00	28.00	130.63	38.75	21.17	193.18
Sobel							
1	113.71	82.00	31.00	117.02	42.12	32.10	n/a
2	122.98	92.00	30.00	98.72	39.49	17.92	n/a
3	112.73	81.00	31.00	100.87	39.32	19.71	n/a

ตาราง 4.1 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	171.10	78.00	93.00	116.55	39.03	23.33	424.00
2	154.70	83.00	71.00	103.28	40.94	21.80	425.76
3	159.27	86.00	73.00	101.76	39.99	20.50	425.50
Gaussian Blur							
1	234.26	87.00	147.00	36.22	26.43	5.58	196.77
2	205.85	87.00	118.00	36.05	26.27	5.61	197.22
3	199.47	81.00	118.00	40.03	26.49	9.03	195.02
Mean Filter							
1	139.39	83.00	56.00	112.39	45.82	21.35	197.05
2	148.58	92.00	56.00	167.16	53.75	24.53	194.95
3	141.85	86.00	55.00	122.34	41.73	21.48	194.33
Median Filter							
1	669.09	248.00	421.00	n/a	n/a	n/a	616.33
2	678.30	263.00	415.00	n/a	n/a	n/a	616.00
3	660.53	242.00	418.00	n/a	n/a	n/a	621.88
Conservative Smoothing							
1	153.31	90.00	63.00	n/a	n/a	n/a	110.38
2	152.49	91.00	61.00	n/a	n/a	n/a	109.23
3	157.41	93.00	64.00	n/a	n/a	n/a	111.36
Texturing							
1	175.67	157.00	14.00	n/a	n/a	n/a	24.97
2	174.47	159.00	12.00	n/a	n/a	n/a	23.98
3	178.57	162.00	14.00	n/a	n/a	n/a	24.67
Texture Merge							
1	185.49	142.00	35.00	n/a	n/a	n/a	156.26
2	154.93	128.00	24.00	n/a	n/a	n/a	144.47
3	155.53	128.00	25.00	n/a	n/a	n/a	146.45
Texture Filter							
1	173.25	140.00	28.00	n/a	n/a	n/a	152.39
2	157.57	131.00	24.00	n/a	n/a	n/a	149.64
3	150.57	125.00	22.00	n/a	n/a	n/a	148.79

4.2.4.1.1.2 รูปที่2



รูป4.2 รูปที่ใช้ในการทดลอง

ตาราง 4.2 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	106.35	89.00	17.00	38.28	34.28	3.23	4.62
2	103.41	88.00	15.00	39.91	36.08	3.17	3.07
3	100.88	85.00	15.00	38.94	34.78	3.40	3.32
Sepia							
1	100.99	82.00	18.00	n/a	n/a	n/a	17.89
2	102.99	87.00	15.00	n/a	n/a	n/a	14.32
3	99.43	83.00	16.00	n/a	n/a	n/a	14.92
Hue Modifier							
1	114.56	90.00	24.00	55.80	41.42	4.52	125.26
2	110.29	88.00	22.00	55.55	40.46	4.99	125.16
3	109.47	86.00	23.00	51.33	38.83	3.86	126.68
Rotate Channel							
1	103.31	86.00	17.00	58.69	44.81	6.77	4.25
2	97.34	82.00	15.00	44.36	35.01	5.51	3.84
3	102.55	87.0	15.00	45.62	35.77	5.89	4.14
Invert							
1	106.71	89.00	17.00	44.16	35.64	4.48	5.58
2	101.48	86.00	15.00	43.42	35.47	3.66	5.19
3	110.91	95.00	15.00	44.17	35.83	4.52	5.21

ตาราง 4.2 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Color Reduction							
1	161.62	96.00	66.00	n/a	n/a	n/a	3129.24
2	154.09	91.00	63.00	n/a	n/a	n/a	3182.55
3	154.33	93.00	61.00	n/a	n/a	n/a	3208.87
Contrast Strecth							
1	109.25	90.00	19.00	n/a	n/a	n/a	6.03
2	114.49	94.00	19.00	n/a	n/a	n/a	3.82
3	100.67	84.00	16.00	n/a	n/a	n/a	3.80
Gamma Correction							
1	110.54	93.00	17.00	114.27	51.84	19.81	6.80
2	102.83	88.00	14.00	117.78	54.23	17.45	6.15
3	108.41	93.00	15.00	127.01	48.05	20.33	6.06
Brightness Correction							
1	105.70	90.00	15.00	44.34	36.09	4.30	8.08
2	106.93	90.00	16.00	42.56	34.24	4.32	6.89
3	104.65	89.00	15.00	42.69	34.26	4.40	6.97
Contrast Correction							
1	107.67	90.00	17.00	48.08	37.08	6.73	8.04
2	108.95	92.00	16.00	49.06	38.80	6.09	6.95
3	105.86	89.00	16.00	49.73	39.74	6.06	6.94
Saturation Correction							
1	121.05	87.00	34.00	55.93	41.15	4.75	156.84
2	121.78	92.00	32.00	60.01	42.92	7.68	154.11
3	106.62	86.00	20.00	73.51	59.08	5.11	151.96
RGB Level Linear							
1	108.64	89.00	19.00	n/a	n/a	n/a	7.40
2	110.80	92.00	18.00	n/a	n/a	n/a	6.94
3	105.49	88.00	17.00	n/a	n/a	n/a	6.92

ตาราง 4.2 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

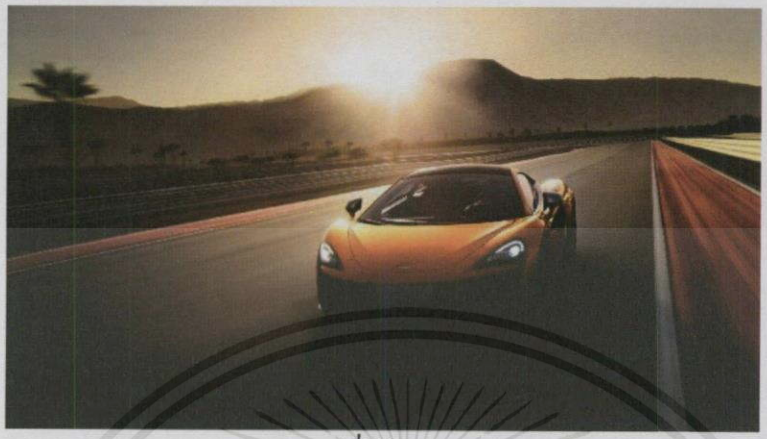
ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
YCbCr Level Linear							
1	116.21	91.00	25.00	n/a	n/a	n/a	108.64
2	120.70	92.00	28.00	n/a	n/a	n/a	107.50
3	117.02	92.00	25.00	n/a	n/a	n/a	104.72
HSL Level Linear							
1	130.54	95.00	35.00	n/a	n/a	n/a	154.13
2	169.03	128.00	41.00	n/a	n/a	n/a	151.70
3	130.95	95.00	35.00	n/a	n/a	n/a	151.07
Blur							
1	270.43	91.00	179.00	141.86	53.56	35.99	430.87
2	202.98	108.00	94.00	112.62	49.85	20.03	427.44
3	192.92	103.00	93.00	116.87	48.02	26.30	433.44
Edge							
1	125.57	100.00	25.00	121.80	50.10	25.38	201.32
2	120.15	94.00	26.00	112.11	47.62	18.45	200.81
3	124.83	100.00	24.00	112.43	49.45	20.24	201.00
Sharpen							
1	126.61	98.00	28.00	117.14	54.22	19.36	197.87
2	130.26	101.00	29.00	114.57	49.96	20.38	196.99
3	123.69	94.00	29.00	117.85	52.34	20.95	197.06
Sobel							
1	120.33	89.00	31.00	124.38	49.24	29.68	n/a
2	135.32	99.00	36.00	136.01	52.27	22.06	n/a
3	130.43	95.00	35.00	125.35	51.09	29.49	n/a
Motion Blur							
1	168.51	89.00	79.00	113.50	48.47	21.72	426.72
2	162.19	88.00	74.00	119.13	54.28	21.72	429.50
3	163.42	89.00	74.00	117.00	52.16	20.63	429.16
Gaussian Blur							
1	226.29	104.00	122.00	55.39	37.56	13.48	196.37
2	231.00	111.00	120.00	46.25	34.75	7.24	195.01
3	216.46	95.00	121.00	47.70	36.38	6.78	195.39

ตาราง 4.2 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Mean Filter							
1	123.39	94.00	29.00	112.93	49.43	17.92	193.46
2	126.05	97.00	29.00	115.11	50.33	20.06	211.37
3	122.98	93.00	29.00	158.92	51.79	19.94	195.55
Median Filter							
1	692.75	271.00	421.00	n/a	n/a	n/a	682.56
2	659.36	234.00	425.00	n/a	n/a	n/a	681.74
3	701.83	274.00	427.00	n/a	n/a	n/a	682.00
Conservative Smoothing							
1	154.84	93.00	61.00	n/a	n/a	n/a	125.60
2	157.23	94.00	63.00	n/a	n/a	n/a	125.76
3	147.79	86.00	61.00	n/a	n/a	n/a	123.96
Texturing							
1	140.43	123.00	14.00	n/a	n/a	n/a	24.43
2	142.89	127.00	13.00	n/a	n/a	n/a	24.82
3	146.24	130.00	13.00	n/a	n/a	n/a	24.91
Texture Merge							
1	160.02	132.00	25.00	n/a	n/a	n/a	154.46
2	151.79	124.00	25.00	n/a	n/a	n/a	156.15
3	150.36	122.00	25.00	n/a	n/a	n/a	151.95
Texture Filter							
1	157.18	131.00	23.00	n/a	n/a	n/a	152.69
2	153.99	128.00	23.00	n/a	n/a	n/a	157.57
3	161.43	134.00	24.00	n/a	n/a	n/a	152.58

4.2.4.1.2 จำนวนเทรดเท่ากับ16

4.2.4.1.2.1 รูปที่1



รูป4.3 รูปที่ใช้ในการทดลอง

ตาราง 4.3 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	115.81	100.00	15.00	29.20	25.33	3.21	8.24
2	114.40	97.00	17.00	30.44	26.74	2.96	3.17
3	114.58	98.00	16.00	30.11	26.13	3.28	3.19
Sepia							
1	123.48	104.00	19.00	n/a	n/a	n/a	14.93
2	108.69	91.00	17.00	n/a	n/a	n/a	12.32
3	108.64	91.00	17.00	n/a	n/a	n/a	12.42
Hue Modifier							
1	111.10	85.00	26.00	43.27	30.37	4.20	123.02
2	107.40	84.00	23.00	43.22	29.29	4.00	122.06
3	122.22	99.00	23.00	45.60	29.70	7.22	122.06
Rotate Channel							
1	120.12	103.00	17.00	51.58	36.18	8.88	4.59
2	106.54	90.00	16.00	35.67	25.70	5.97	4.04
3	106.29	90.00	16.00	37.07	27.14	6.02	4.05

ตาราง 4.3 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	137.54	119.00	18.00	37.88	26.30	6.42	6.06
2	111.87	95.00	16.00	34.54	26.42	4.60	5.57
3	96.35	77.00	19.00	34.47	26.10	4.53	5.36
Color Reduction							
1	163.96	96.00	67.00	n/a	n/a	n/a	3056.77
2	150.78	87.00	63.00	n/a	n/a	n/a	3111.71
3	161.37	98.00	63.00	n/a	n/a	n/a	3081.56
Contrast Strecth							
1	117.22	98.00	19.00	n/a	n/a	n/a	11.06
2	105.32	89.00	16.00	n/a	n/a	n/a	3.85
3	105.71	88.00	17.00	n/a	n/a	n/a	3.87
Gamma Correction							
1	121.88	102.00	19.00	108.38	40.00	18.03	6.46
2	91.46	75.00	16.00	98.45	40.88	16.57	7.06
3	101.17	84.00	17.00	113.57	45.81	26.95	6.08
Brightness Correction							
1	148.04	128.00	20.00	34.76	26.82	3.86	8.58
2	115.06	98.00	17.00	34.25	26.24	3.90	7.09
3	111.73	94.00	17.00	34.67	26.47	3.81	7.04
Contrast Correction							
1	95.27	77.00	18.00	39.22	29.25	6.04	8.09
2	97.81	79.00	18.00	38.99	29.16	5.91	7.44
3	100.61	83.00	17.00	38.58	28.87	5.86	7.48
Saturation Correction							
1	118.24	85.00	33.00	56.54	39.56	5.07	139.93
2	118.92	86.00	32.00	42.84	30.00	5.00	138.44
3	121.53	87.00	34.00	42.22	29.80	4.67	136.00

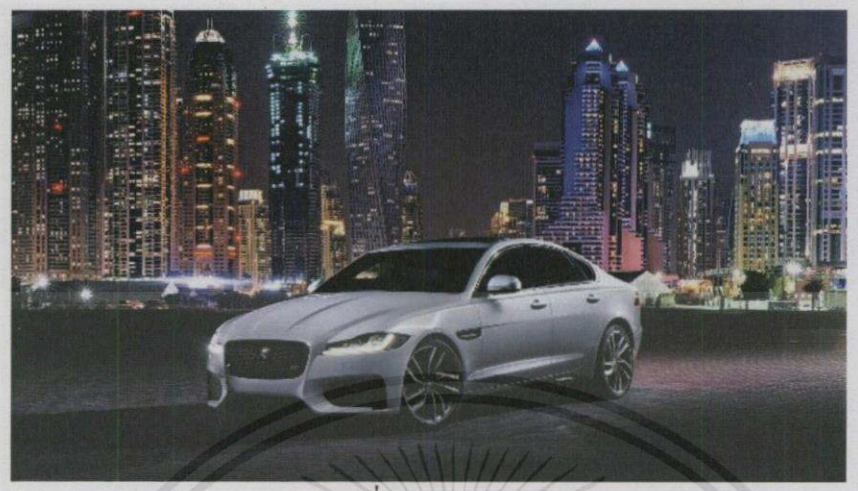
ตาราง 4.3 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	129.85	105.00	24.00	n/a	n/a	n/a	7.51
2	108.85	88.00	20.00	n/a	n/a	n/a	7.30
3	101.84	81.00	20.00	n/a	n/a	n/a	7.33
YCbCr Level Linear							
1	196.18	145.00	51.00	n/a	n/a	n/a	105.67
2	110.49	85.00	25.00	n/a	n/a	n/a	119.42
3	103.44	84.00	19.00	n/a	n/a	n/a	104.74
HSL Level Linear							
1	119.00	83.00	36.00	n/a	n/a	n/a	138.49
2	120.55	83.00	37.00	n/a	n/a	n/a	138.51
3	117.78	80.00	37.00	n/a	n/a	n/a	137.73
Blur							
1	214.32	115.00	99.00	216.31	136.26	38.84	428.32
2	194.50	93.00	101.00	101.11	39.01	20.11	426.86
3	182.53	81.00	101.00	129.15	41.77	45.29	429.38
Edge							
1	142.85	85.00	57.00	100.23	40.51	19.06	200.21
2	214.49	157.00	57.00	101.54	40.37	19.02	200.91
3	208.63	150.00	58.00	117.78	58.41	19.19	203.98
Sharpen							
1	215.49	147.00	68.00	107.11	41.10	24.08	191.94
2	202.55	137.00	65.00	105.31	42.10	20.89	192.86
3	198.81	133.00	65.00	130.63	38.75	21.17	193.18
Sobel							
1	163.11	130.00	33.00	117.02	42.12	32.10	n/a
2	162.54	129.00	33.00	98.72	39.49	17.92	n/a
3	149.95	117.00	32.00	100.87	39.32	19.71	n/a

ตาราง 4.3 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	213.73	127.00	86.00	116.55	39.03	23.33	424.00
2	212.43	126.00	86.00	103.28	40.94	21.80	425.76
3	220.72	133.33	87.00	101.76	39.99	20.50	425.50
Gaussian Blur							
1	254.20	125.00	129.00	36.22	26.43	5.58	196.77
2	265.98	138.00	127.00	36.05	26.27	5.61	197.22
3	263.77	136.00	127.00	40.03	26.49	9.03	195.02
Mean Filter							
1	194.48	129.00	65.00	112.39	45.82	21.35	197.05
2	209.25	141.00	68.00	167.16	53.75	24.53	194.95
3	201.10	134.00	67.00	122.34	41.73	21.48	194.33
Median Filter							
1	555.63	122.00	433.00	n/a	n/a	n/a	616.33
2	541.59	120.00	421.00	n/a	n/a	n/a	616.00
3	560.30	137.00	423.00	n/a	n/a	n/a	621.88
Conservative Smoothing							
1	180.42	115.00	65.00	n/a	n/a	n/a	110.38
2	179.87	115.00	64.00	n/a	n/a	n/a	109.23
3	184.53	119.00	65.00	n/a	n/a	n/a	111.36
Texturing							
1	144.84	126.00	16.00	n/a	n/a	n/a	24.97
2	145.18	128.00	13.00	n/a	n/a	n/a	23.98
3	149.51	133.00	14.00	n/a	n/a	n/a	24.67
Texture Merge							
1	155.74	127.00	26.00	n/a	n/a	n/a	156.26
2	153.16	125.00	25.00	n/a	n/a	n/a	144.47
3	157.15	128.00	26.00	n/a	n/a	n/a	146.45
Texture Filter							
1	149.40	123.00	24.00	n/a	n/a	n/a	152.39
2	150.32	124.00	24.00	n/a	n/a	n/a	149.64
3	162.80	136.00	24.00	n/a	n/a	n/a	148.79

4.2.4.1.2.2 รูปที่2



รูป4.4 รูปที่ใช้ในการทดลอง

ตาราง 4.4 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	115.83	97.00	18.00	38.28	34.28	3.23	4.62
2	136.13	117.00	19.00	39.91	36.08	3.17	3.07
3	145.45	127.00	18.00	38.94	34.78	3.40	3.32
Sepia							
1	151.15	131.00	20.00	n/a	n/a	n/a	17.89
2	128.30	109.00	19.00	n/a	n/a	n/a	14.32
3	139.39	119.00	20.00	n/a	n/a	n/a	14.92
Hue Modifier							
1	148.27	123.00	25.00	55.80	41.42	4.52	125.26
2	139.72	115.00	24.00	55.55	40.46	4.99	125.16
3	138.85	113.00	25.00	51.33	38.83	3.86	126.68
Rotate Channel							
1	140.93	123.00	17.00	58.69	44.81	6.77	4.25
2	132.77	115.00	17.00	44.36	35.01	5.51	3.84
3	145.47	87.0	18.00	45.62	35.77	5.89	4.14

ตาราง 4.4 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	141.47	123.00	18.00	44.16	35.64	4.48	5.58
2	138.04	120.00	18.00	43.42	35.47	3.66	5.19
3	144.28	127.00	17.00	44.17	35.83	4.52	5.21
Color Reduction							
1	194.15	126.00	68.00	n/a	n/a	n/a	3129.24
2	192.76	125.00	67.00	n/a	n/a	n/a	3182.55
3	191.99	126.00	65.00	n/a	n/a	n/a	3208.87
Contrast Strecth							
1	139.11	120.00	19.00	n/a	n/a	n/a	6.03
2	134.76	116.00	18.00	n/a	n/a	n/a	3.82
3	140.70	121.00	19.00	n/a	n/a	n/a	3.80
Gamma Correction							
1	141.11	122.00	19.00	114.27	51.84	19.81	6.80
2	135.97	118.00	17.00	117.78	54.23	17.45	6.15
3	139.93	115.00	18.00	127.01	48.05	20.33	6.06
Brightness Correction							
1	145.56	128.00	17.00	44.34	36.09	4.30	8.08
2	138.66	120.00	18.00	42.56	34.24	4.32	6.89
3	131.20	112.00	19.00	42.69	34.26	4.40	6.97
Contrast Correction							
1	138.06	120.00	18.00	48.08	37.08	6.73	8.04
2	154.32	134.00	20.00	49.06	38.80	6.09	6.95
3	155.98	137.00	18.00	49.73	39.74	6.06	6.94
Saturation Correction							
1	167.29	131.00	36.00	55.93	41.15	4.75	156.84
2	169.51	133.00	36.00	60.01	42.92	7.68	154.11
3	152.68	117.00	35.00	73.51	59.08	5.11	151.96

ตาราง 4.4 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

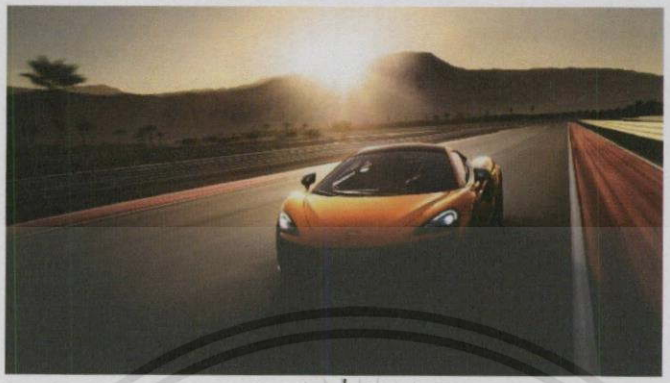
ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	146.14	124.00	22.00	n/a	n/a	n/a	7.40
2	142.85	120.00	22.00	n/a	n/a	n/a	6.94
3	157.82	136.00	21.00	n/a	n/a	n/a	6.92
YCbCr Level Linear							
1	152.80	127.00	25.00	n/a	n/a	n/a	108.64
2	146.45	125.00	21.00	n/a	n/a	n/a	107.50
3	146.89	127.00	19.00	n/a	n/a	n/a	104.72
HSL Level Linear							
1	153.64	129.00	24.00	n/a	n/a	n/a	154.13
2	143.19	120.00	23.00	n/a	n/a	n/a	151.70
3	164.46	126.00	38.00	n/a	n/a	n/a	151.07
Blur							
1	250.92	146.00	104.00	141.86	53.56	35.99	430.87
2	209.42	108.00	101.00	112.62	49.85	20.03	427.44
3	231.52	127.00	104.00	116.87	48.02	26.30	433.44
Edge							
1	193.07	134.00	59.00	121.80	50.10	25.38	201.32
2	201.68	142.00	59.00	112.11	47.62	18.45	200.81
3	205.11	142.00	63.00	112.43	49.45	20.24	201.00
Sharpen							
1	202.22	134.00	68.00	117.14	54.22	19.36	197.87
2	219.89	151.00	68.00	114.57	49.96	20.38	196.99
3	192.20	125.00	67.00	117.85	52.34	20.95	197.06
Sobel							
1	160.79	126.00	34.00	124.38	49.24	29.68	n/a
2	156.54	122.00	34.00	136.01	52.27	22.06	n/a
3	162.81	129.00	33.00	125.35	51.09	29.49	n/a

ตาราง 4.4 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	230.30	142.00	88.00	113.50	48.47	21.72	426.72
2	231.50	143.00	88.00	119.13	54.28	21.72	429.50
3	233.33	145.00	88.00	117.00	52.16	20.63	429.16
Gaussian Blur							
1	268.62	138.00	130.00	55.39	37.56	13.48	196.37
2	271.46	141.00	130.00	46.25	34.75	7.24	195.01
3	256.68	133.00	132.00	47.70	36.38	6.78	195.39
Mean Filter							
1	209.05	141.00	68.00	112.93	49.43	17.92	193.46
2	194.46	126.00	68.00	115.11	50.33	20.06	211.37
3	204.44	138.00	66.00	158.92	51.79	19.94	195.55
Median Filter							
1	571.57	145.00	426.00	n/a	n/a	n/a	682.56
2	563.96	136.00	427.00	n/a	n/a	n/a	681.74
3	554.25	135.00	419.00	n/a	n/a	n/a	682.00
Conservative Smoothing							
1	197.86	129.00	68.00	n/a	n/a	n/a	125.60
2	183.58	121.00	62.00	n/a	n/a	n/a	125.76
3	185.29	123.00	62.00	n/a	n/a	n/a	123.96
Texturing							
1	144.59	128.00	14.00	n/a	n/a	n/a	24.43
2	137.91	121.00	13.00	n/a	n/a	n/a	24.82
3	141.03	125.00	13.00	n/a	n/a	n/a	24.91
Texture Merge							
1	155.55	128.00	25.00	n/a	n/a	n/a	154.46
2	157.36	128.00	26.00	n/a	n/a	n/a	156.15
3	155.20	126.00	26.00	n/a	n/a	n/a	151.95
Texture Filter							
1	155.94	129.00	24.00	n/a	n/a	n/a	152.69
2	158.20	131.00	24.00	n/a	n/a	n/a	157.57
3	157.42	131.00	23.00	n/a	n/a	n/a	152.58

4.2.4.1.3 จำนวนเทรด 32

4.2.4.1.3.1 รูปที่1



รูป4.5 รูปที่ใช้ในการทดลอง

ตาราง 4.5 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	101.26	84.00	17.00	29.20	25.33	3.21	8.24
2	97.52	82.00	15.00	30.44	26.74	2.96	3.17
3	100.76	84.00	16.00	30.11	26.13	3.28	3.19
Sepia							
1	99.99	83.00	16.00	n/a	n/a	n/a	14.93
2	102.37	85.00	17.00	n/a	n/a	n/a	12.32
3	99.37	81.00	18.00	n/a	n/a	n/a	12.42
Hue Modifier							
1	108.03	85.00	23.00	43.27	30.37	4.20	123.02
2	103.50	80.00	23.00	43.22	29.29	4.00	122.06
3	103.74	99.00	23.00	45.60	29.70	7.22	122.06
Rotate Channel							
1	101.35	85.00	16.00	51.58	36.18	8.88	4.59
2	98.65	83.00	15.00	35.67	25.70	5.97	4.04
3	99.02	83.00	16.00	37.07	27.14	6.02	4.05

ตาราง 4.5 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	99.82	83.00	16.00	37.88	26.30	6.42	6.06
2	100.07	83.00	17.00	34.54	26.42	4.60	5.57
3	96.15	78.00	18.00	34.47	26.10	4.53	5.36
Color Reduction							
1	155.82	91.00	64.00	n/a	n/a	n/a	3056.77
2	153.47	90.00	63.00	n/a	n/a	n/a	3111.71
3	154.80	91.00	63.00	n/a	n/a	n/a	3081.56
Contrast Strecth							
1	97.15	80.00	17.00	n/a	n/a	n/a	11.06
2	96.00	80.00	16.00	n/a	n/a	n/a	3.85
3	101.25	82.00	19.00	n/a	n/a	n/a	3.87
Gamma Correction							
1	98.20	81.00	17.00	108.38	40.00	18.03	6.46
2	100.44	84.00	16.00	98.45	40.88	16.57	7.06
3	100.06	83.00	17.00	113.57	45.81	26.95	6.08
Brightness Correction							
1	96.58	79.00	17.00	34.76	26.82	3.86	8.58
2	96.17	79.00	17.00	34.25	26.24	3.90	7.09
3	96.64	80.00	16.00	34.67	26.47	3.81	7.04
Contrast Correction							
1	98.20	79.00	19.00	39.22	29.25	6.04	8.09
2	101.01	84.00	17.00	38.99	29.16	5.91	7.44
3	97.66	80.00	17.00	38.58	28.87	5.86	7.48
Saturation Correction							
1	117.55	84.00	33.00	56.54	39.56	5.07	139.93
2	113.67	79.00	34.00	42.84	30.00	5.00	138.44
3	116.75	84.00	32.00	42.22	29.80	4.67	136.00

ตาราง 4.5 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	100.17	83.00	17.00	n/a	n/a	n/a	7.51
2	101.72	85.00	16.00	n/a	n/a	n/a	7.30
3	98.44	80.00	18.00	n/a	n/a	n/a	7.33
YCbCr Level Linear							
1	101.13	81.00	20.00	n/a	n/a	n/a	105.67
2	103.18	83.00	20.00	n/a	n/a	n/a	119.42
3	199.29	80.00	19.00	n/a	n/a	n/a	104.74
HSL Level Linear							
1	121.39	85.00	36.00	n/a	n/a	n/a	138.49
2	99.84	79.00	20.00	n/a	n/a	n/a	138.51
3	104.15	83.00	21.00	n/a	n/a	n/a	137.73
Blur							
1	197.42	87.00	110.00	216.31	136.26	38.84	428.32
2	195.44	83.00	112.00	101.11	39.01	20.11	426.86
3	195.46	85.00	111.00	129.15	41.77	45.29	429.38
Edge							
1	160.31	95.00	65.00	100.23	40.51	19.06	200.21
2	186.79	122.00	64.00	101.54	40.37	19.02	200.91
3	204.79	140.00	64.00	117.78	58.41	19.19	203.98
Sharpen							
1	203.87	129.00	74.00	107.11	41.10	24.08	191.94
2	195.61	123.00	72.00	105.31	42.10	20.89	192.86
3	197.98	124.00	73.00	130.63	38.75	21.17	193.18
Sobel							
1	147.36	115.00	32.00	117.02	42.12	32.10	n/a
2	144.24	113.00	31.00	98.72	39.49	17.92	n/a
3	139.34	105.00	34.00	100.87	39.32	19.71	n/a

ตาราง 4.5 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	233.84	133.00	100.00	116.55	39.03	23.33	424.00
2	247.32	134.00	113.00	103.28	40.94	21.80	425.76
3	250.28	136.00	114.00	101.76	39.99	20.50	425.50
Gaussian Blur							
1	280.59	137.00	143.00	36.22	26.43	5.58	196.77
2	277.07	135.00	142.00	36.05	26.27	5.61	197.22
3	280.17	138.00	142.00	40.03	26.49	9.03	195.02
Mean Filter							
1	207.09	132.00	75.00	112.39	45.82	21.35	197.05
2	215.39	140.00	75.00	167.16	53.75	24.53	194.95
3	212.57	138.00	74.00	122.34	41.73	21.48	194.33
Median Filter							
1	551.26	126.00	425.00	n/a	n/a	n/a	616.33
2	541.44	118.00	423.00	n/a	n/a	n/a	616.00
3	537.27	113.00	424.00	n/a	n/a	n/a	621.88
Conservative Smoothing							
1	169.72	102.00	67.00	n/a	n/a	n/a	110.38
2	177.39	111.00	66.00	n/a	n/a	n/a	109.23
3	176.51	110.00	66.00	n/a	n/a	n/a	111.36
Texturing							
1	144.16	127.00	14.00	n/a	n/a	n/a	24.97
2	142.71	125.00	14.00	n/a	n/a	n/a	23.98
3	138.25	123.00	15.00	n/a	n/a	n/a	24.67
Texture Merge							
1	151.34	122.00	26.00	n/a	n/a	n/a	156.26
2	158.26	129.00	26.00	n/a	n/a	n/a	144.47
3	154.11	124.00	27.00	n/a	n/a	n/a	146.45
Texture Filter							
1	143.24	115.00	25.00	n/a	n/a	n/a	152.39
2	146.98	119.00	25.00	n/a	n/a	n/a	149.64
3	146.60	119.00	24.00	n/a	n/a	n/a	148.79

4.2.4.1.3.2 รูปที่2

รูป4.6 รูปที่ใช้ในการทดลอง

ตาราง 4.6 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	143.45	123.00	20.00	38.28	34.28	3.23	4.62
2	138.49	11900	19.00	39.91	36.08	3.17	3.07
3	143.02	126.00	17.00	38.94	34.78	3.40	3.32
Sepia							
1	144.77	126.00	18.00	n/a	n/a	n/a	17.89
2	140.94	122.00	18.00	n/a	n/a	n/a	14.32
3	151.46	134.00	19.00	n/a	n/a	n/a	14.92
Hue Modifier							
1	140.02	116.00	24.00	55.80	41.42	4.52	125.26
2	151.36	126.00	25.00	55.55	40.46	4.99	125.16
3	155.24	130.00	25.00	51.33	38.83	3.86	126.68
Rotate Channel							
1	150.21	132.00	18.00	58.69	44.81	6.77	4.25
2	145.30	127.00	18.00	44.36	35.01	5.51	3.84
3	145.01	127.00	18.00	45.62	35.77	5.89	4.14

ตาราง 4.6 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	134.11	116.00	18.00	44.16	35.64	4.48	5.58
2	113.58	95.00	18.00	43.42	35.47	3.66	5.19
3	108.23	89.00	19.00	44.17	35.83	4.52	5.21
Color Reduction							
1	163.39	98.00	65.00	n/a	n/a	n/a	3129.24
2	182.00	116.00	66.00	n/a	n/a	n/a	3182.55
3	191.41	126.00	65.00	n/a	n/a	n/a	3208.87
Contrast Strecth							
1	131.45	112.00	19.00	n/a	n/a	n/a	6.03
2	132.90	114.00	18.00	n/a	n/a	n/a	3.82
3	135.14	116.00	19.00	n/a	n/a	n/a	3.80
Gamma Correction							
1	129.45	110.00	19.00	114.27	51.84	19.81	6.80
2	138.45	120.00	18.00	117.78	54.23	17.45	6.15
3	152.62	133.00	19.00	127.01	48.05	20.33	6.06
Brightness Correction							
1	152.85	135.00	17.00	44.34	36.09	4.30	8.08
2	114.50	96.00	18.00	42.56	34.24	4.32	6.89
3	149.35	130.00	19.00	42.69	34.26	4.40	6.97
Contrast Correction							
1	140.50	119.00	21.00	48.08	37.08	6.73	8.04
2	145.78	126.00	19.00	49.06	38.80	6.09	6.95
3	152.26	134.00	18.00	49.73	39.74	6.06	6.94
Saturation Correction							
1	163.78	128.00	35.00	55.93	41.15	4.75	156.84
2	167.70	132.00	35.00	60.01	42.92	7.68	154.11
3	171.40	136.00	35.00	73.51	59.08	5.11	151.96

ตาราง 4.6 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	136.39	116.00	20.00	n/a	n/a	n/a	7.40
2	126.99	106.00	20.00	n/a	n/a	n/a	6.94
3	146.18	126.00	20.00	n/a	n/a	n/a	6.92
YCbCr Level Linear							
1	155.80	135.00	20.00	n/a	n/a	n/a	108.64
2	160.09	140.00	20.00	n/a	n/a	n/a	107.50
3	162.70	142.00	20.00	n/a	n/a	n/a	104.72
HSL Level Linear							
1	175.66	138.00	37.00	n/a	n/a	n/a	154.13
2	155.88	119.00	36.00	n/a	n/a	n/a	151.70
3	168.51	130.00	38.00	n/a	n/a	n/a	151.07
Blur							
1	252.47	140.00	112.00	141.86	53.56	35.99	430.87
2	256.62	143.00	113.00	112.62	49.85	20.03	427.44
3	269.13	156.00	113.00	116.87	48.02	26.30	433.44
Edge							
1	207.12	141.00	66.00	121.80	50.10	25.38	201.32
2	206.40	140.00	66.00	112.11	47.62	18.45	200.81
3	221.78	156.00	65.00	112.43	49.45	20.24	201.00
Sharpen							
1	221.74	145.00	76.00	117.14	54.22	19.36	197.87
2	209.55	134.00	75.00	114.57	49.96	20.38	196.99
3	217.98	143.00	74.00	117.85	52.34	20.95	197.06
Sobel							
1	174.44	169.55	33.00	124.38	49.24	29.68	n/a
2	169.55	134.00	35.00	136.01	52.27	22.06	n/a
3	174.56	141.00	34.00	125.35	51.09	29.49	n/a

ตาราง 4.6 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	250.11	153.00	97.00	113.50	48.47	21.72	426.72
2	241.32	128.00	113.00	119.13	54.28	21.72	429.50
3	265.71	151.00	114.00	117.00	52.16	20.63	429.16
Gaussian Blur							
1	298.81	151.00	147.00	55.39	37.56	13.48	196.37
2	300.21	149.00	151.00	46.25	34.75	7.24	195.01
3	292.33	147.00	145.00	47.70	36.38	6.78	195.39
Mean Filter							
1	221.71	146.00	75.00	112.93	49.43	17.92	193.46
2	218.92	143.00	75.00	115.11	50.33	20.06	211.37
3	226.14	149.00	77.00	158.92	51.79	19.94	195.55
Median Filter							
1	561.76	135.00	426.00	n/a	n/a	n/a	682.56
2	561.57	135.00	426.00	n/a	n/a	n/a	681.74
3	615.40	112.00	503.00	n/a	n/a	n/a	682.00
Conservative Smoothing							
1	189.61	126.00	63.00	n/a	n/a	n/a	125.60
2	185.86	119.00	66.00	n/a	n/a	n/a	125.76
3	191.91	130.00	61.00	n/a	n/a	n/a	123.96
Texturing							
1	144.78	128.00	14.00	n/a	n/a	n/a	24.43
2	142.99	126.00	13.00	n/a	n/a	n/a	24.82
3	143.95	125.00	13.00	n/a	n/a	n/a	24.91
Texture Merge							
1	159.42	131.00	26.00	n/a	n/a	n/a	154.46
2	154.86	127.00	25.00	n/a	n/a	n/a	156.15
3	155.15	125.00	27.00	n/a	n/a	n/a	151.95
Texture Filter							
1	156.50	130.00	24.00	n/a	n/a	n/a	152.69
2	157.43	130.00	25.00	n/a	n/a	n/a	157.57
3	150.42	123.00	25.00	n/a	n/a	n/a	152.58

4.2.4.2 ภาพขนาด 3480*2160 (UHD)

4.2.4.2.1 จำนวนเทรดเท่ากับ 1

4.2.4.2.2.1 รูปที่1

รูป4.7 รูปที่ใช้ในการทดลอง

ตาราง 4.7 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	716.86	693.00	23.00	239.14	226.46	10.02	13.31
2	781.55	763.00	18.00	238.90	226.89	9.26	12.48
3	703.15	686.00	17.00	251.33	229.68	11.01	12.54
Sepia							
1	913.16	895.00	18.00	n/a	n/a	n/a	51.83
2	970.99	953.00	17.00	n/a	n/a	n/a	50.52
3	887.53	872.00	15.00	n/a	n/a	n/a	52.84
Hue Modifier							
1	719.06	694.00	25.00	296.39	248.50	13.98	492.82
2	715.77	692.00	23.00	312.96	247.28	20.48	493.52
3	696.19	673.00	23.00	302.34	247.56	15.14	492.52
Rotate Channel							
1	703.28	684.00	19.00	295.27	250.68	25.03	15.99
2	827.85	809.00	18.00	263.14	230.16	17.96	15.83
3	703.02	687.00	16.00	260.46	228.79	16.92	15.76

ตาราง 4.7 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	785.73	767.00	18.00	253.70	225.72	13.58	20.85
2	714.77	697.00	17.00	423.23	355.25	13.37	20.86
3	735.63	719.00	16.00	261.30	228.04	14.06	20.77
Color Reduction							
1	729.87	664.00	65.00	n/a	n/a	n/a	13337.78
2	744.90	681.00	63.00	n/a	n/a	n/a	13326.45
3	777.02	705.00	65.00	n/a	n/a	n/a	13222.22
Contrast Strecth							
1	456.69	435.00	21.00	n/a	n/a	n/a	17.75
2	417.10	399.00	17.00	n/a	n/a	n/a	18.27
3	416.60	396.00	16.00	n/a	n/a	n/a	17.72
Gamma Correction							
1	705.94	688.00	18.00	644.54	398.82	66.38	24.38
2	700.69	682.00	18.00	685.26	389.16	76.37	24.46
3	700.69	682.00	18.00	501.52	277.10	63.41	24.64
Brightness Correction							
1	707.25	686.00	21.00	254.23	226.05	13.72	28.26
2	676.48	661.00	15.00	256.20	227.16	13.88	27.30
3	679.32	661.00	18.00	262.18	233.76	14.09	26.93
Contrast Correction							
1	686.14	667.00	19.00	257.99	235.03	18.70	28.82
2	736.47	720.00	16.00	266.51	233.03	19.07	28.83
3	687.22	669.00	18.00	268.33	234.90	18.79	28.00
Saturation Correction							
1	837.92	803.00	34.00	283.93	242.02	14.00	576.90
2	767.76	732.00	35.00	281.50	239.37	12.93	607.90
3	835.37	798.00	37.00	280.56	237.23	13.05	579.11

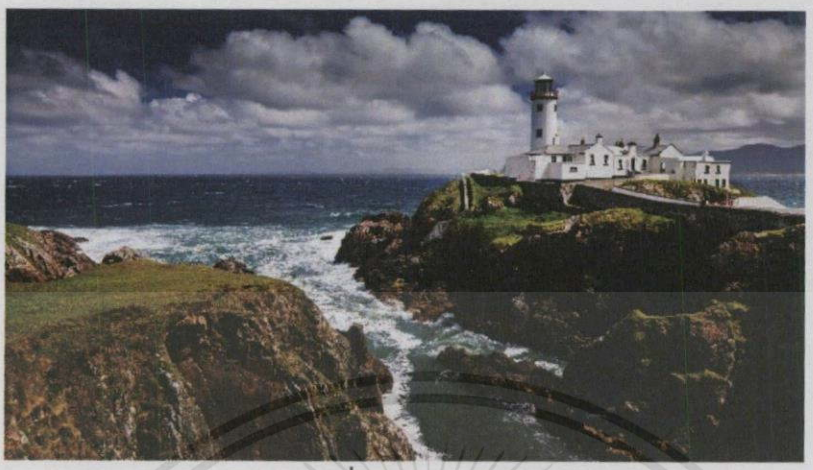
ตาราง 4.7 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	688.72	666.00	22.00	n/a	n/a	n/a	27.76
2	754.15	731.00	23.00	n/a	n/a	n/a	27.71
3	699.18	682.00	17.00	n/a	n/a	n/a	27.71
YCbCr Level Linear							
1	727.75	704.00	23.00	n/a	n/a	n/a	407.93
2	762.06	736.00	26.00	n/a	n/a	n/a	407.14
3	713.31	691.00	21.00	n/a	n/a	n/a	407.89
HSL Level Linear							
1	717.07	697.00	20.00	n/a	n/a	n/a	576.34
2	693.03	671.00	22.00	n/a	n/a	n/a	340.60
3	783.35	745.00	38.00	n/a	n/a	n/a	574.66
Blur							
1	936.68	839.00	97.00	759.55	530.28	68.89	1672.92
2	926.10	828.00	98.00	693.96	458.08	73.63	1672.79
3	979.88	880.00	99.00	611.97	377.24	71.52	1668.49
Edge							
1	934.72	905.00	29.00	598.83	272.07	79.64	790.47
2	970.78	940.00	30.00	520.84	290.94	66.11	788.60
3	970.53	943.00	27.00	521.70	282.69	66.51	790.36
Sharpen							
1	953.35	916.00	37.00	519.02	283.29	72.63	811.67
2	951.01	919.00	32.00	508.61	272.66	67.83	764.10
3	988.48	955.00	33.00	530.69	296.49	65.83	761.46
Sobel							
1	897.59	860.00	37.00	521.44	276.32	74.97	n/a
2	881.87	846.00	35.00	526.07	285.20	72.10	n/a
3	939.39	905.00	34.00	509.65	278.94	68.09	n/a

ตาราง 4.7 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	944.24	864.00	80.00	856.22	275.20	73.80	1670.52
2	940.07	860.00	80.00	658.97	272.47	71.41	1669.12
3	995.85	915.00	80.00	594.39	277.09	73.54	1670.30
Gaussian Blur							
1	1011.55	887.00	124.00	263.85	230.22	18.74	769.74
2	1035.87	914.00	121.00	60.51	228.04	17.17	763.34
3	1046.51	922.00	124.00	282.46	232.12	20.52	770.12
Mean Filter							
1	991.28	957.00	34.00	507.73	276.87	67.16	767.71
2	971.98	940.00	31.00	507.68	278.78	64.88	766.11
3	962.40	929.00	33.00	504.37	277.05	64.06	772.76
Median Filter							
1	1825.81	1403.00	422.00	n/a	n/a	n/a	2323.47
2	1810.79	1388.00	422.00	n/a	n/a	n/a	2327.33
3	1832.68	1407.00	425.00	n/a	n/a	n/a	2370.53
Conservative Smoothing							
1	851.36	788.00	63.00	n/a	n/a	n/a	477.04
2	957.91	893.00	64.00	n/a	n/a	n/a	509.69
3	882.39	813.00	69.00	n/a	n/a	n/a	492.25
Texturing							
1	712.86	692.00	14.00	n/a	n/a	n/a	95.11
2	707.36	689.00	13.00	n/a	n/a	n/a	94.24
3	710.55	692.00	13.00	n/a	n/a	n/a	96.91
Texture Merge							
1	856.51	825.00	25.00	n/a	n/a	n/a	598.68
2	880.49	848.00	26.00	n/a	n/a	n/a	600.22
3	854.03	821.00	25.00	n/a	n/a	n/a	609.70
Texture Filter							
1	877.80	846.00	25.00	n/a	n/a	n/a	619.13
2	889.35	860.00	24.00	n/a	n/a	n/a	615.16
3	879.49	845.00	22.00	n/a	n/a	n/a	613.96

4.2.4.2.1.2 รูปที่2



รูป4.8 รูปที่ใช้ในการทดลอง

ตาราง 4.8 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	556.21	539.00	17.00	130.83	119.04	8.98	13.36
2	589.44	570.00	19.00	131.23	119.68	8.60	12.50
3	603.57	586.00	17.00	133.97	122.14	8.94	13.05
Sepia							
1	582.41	559.00	23.00	n/a	n/a	n/a	53.72
2	621.36	602.00	19.00	n/a	n/a	n/a	54.00
3	563.61	544.00	19.00	n/a	n/a	n/a	53.68
Hue Modifier							
1	599.08	571.00	28.00	176.40	131.80	13.49	493.11
2	608.79	583.00	25.00	180.74	132.38	12.95	489.03
3	575.48	551.00	24.00	176.62	132.18	12.94	488.69
Rotate Channel							
1	604.33	584.00	20.00	154.41	122.04	17.07	15.50
2	579.98	563.00	16.00	154.00	121.23	18.17	16.48
3	588.46	572.00	16.00	154.41	122.36	17.08	16.19

ตาราง 4.8 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	623.50	604.00	19.00	146.56	117.91	13.49	21.49
2	572.72	555.00	17.00	147.47	117.57	14.30	21.80
3	605.71	585.00	20.00	152.17	121.39	12.83	21.44
Color Reduction							
1	649.22	584.00	65.00	n/a	n/a	n/a	13696.07
2	624.39	559.00	65.00	n/a	n/a	n/a	13714.35
3	617.41	553.00	64.00	n/a	n/a	n/a	13679.39
Contrast Strecth							
1	567.47	545.00	22.00	n/a	n/a	n/a	16.96
2	564.08	545.00	18.00	n/a	n/a	n/a	15.81
3	581.98	560.00	21.00	n/a	n/a	n/a	15.74
Gamma Correction							
1	426.92	410.00	16.00	398.35	168.64	64.11	23.47
2	452.84	436.00	16.00	405.60	175.27	61.70	24.28
3	524.24	509.00	15.00	399.56	169.85	65.74	23.79
Brightness Correction							
1	674.39	656.00	18.00	147.88	119.47	13.05	27.74
2	592.33	574.00	18.00	149.71	120.62	11.04	27.48
3	613.43	595.00	18.00	150.16	119.13	14.59	27.30
Contrast Correction							
1	615.26	597.00	18.00	164.69	131.45	18.36	27.81
2	624.73	608.00	16.00	162.98	130.02	18.07	27.83
3	591.21	576.00	15.00	162.75	127.73	18.83	27.78
Saturation Correction							
1	600.78	567.00	33.00	232.87	130.52	14.18	571.35
2	612.19	579.00	33.00	181.37	135.20	14.06	566.63
3	624.88	591.00	33.00	181.77	134.59	13.86	572.62

ตาราง 4.8 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	592.97	57.00	18.00	n/a	n/a	n/a	27.93
2	634.07	616.00	18.00	n/a	n/a	n/a	27.80
3	668.41	647.00	21.00	n/a	n/a	n/a	27.40
YCbCr Level Linear							
1	471.92	449.00	25.00	n/a	n/a	n/a	399.82
2	440.96	418.00	22.00	n/a	n/a	n/a	412.96
3	458.62	440.00	18.00	n/a	n/a	n/a	398.84
HSL Level Linear							
1	462.95	423.00	39.00	n/a	n/a	n/a	574.65
2	492.72	453.00	39.00	n/a	n/a	n/a	579.50
3	445.50	411.00	34.00	n/a	n/a	n/a	576.94
Blur							
1	534.06	433.00	101.00	408.74	172.78	70.16	1725.81
2	538.89	440.00	98.00	463.62	224.13	71.75	1704.70
3	516.65	420.00	96.00	565.74	172.86	70.90	1715.04
Edge							
1	485.36	460.00	25.00	495.85	169.63	65.90	799.96
2	474.09	424.00	50.00	405.27	173.31	64.88	794.87
3	471.74	422.00	49.00	609.38	374.42	65.80	797.11
Sharpen							
1	552.81	518.00	34.00	538.59	305.30	67.02	769.63
2	494.15	465.00	29.00	489.70	172.55	70.86	769.60
3	505.44	474.00	31.00	406.44	171.25	70.95	769.47
Sobel							
1	527.24	491.00	36.00	510.44	275.47	66.83	n/a
2	579.99	545.00	34.00	565.21	170.51	66.91	n/a
3	533.43	502.00	31.00	504.81	178.28	69.29	n/a

ตาราง 4.8 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	675.07	596.00	79.00	406.42	196.85	71.94	1705.54
2	648.53	572.00	76.00	411.03	170.45	69.99	1701.35
3	683.77	606.00	77.00	407.34	171.46	70.76	1711.33
Gaussian Blur							
1	686.52	560.00	126.00	150.62	119.69	17.26	766.70
2	693.77	571.00	122.00	153.42	120.42	17.88	772.76
3	735.52	610.00	122.00	153.26	119.82	18.19	770.69
Mean Filter							
1	651.92	619.00	32.00	485.41	232.35	64.97	783.04
2	639.66	610.00	29.00	403.30	169.16	67.88	763.33
3	668.25	63.700	31.00	409.17	173.75	70.12	769.75
Median Filter							
1	1572.00	1146.00	426.00	n/a	n/a	n/a	2601.17
2	1541.21	1116.00	425.00	n/a	n/a	n/a	2604.60
3	1569.35	1143.00	426.00	n/a	n/a	n/a	2643.47
Conservative Smoothing							
1	574.81	506.00	68.00	n/a	n/a	n/a	495.05
2	577.19	512.00	65.00	n/a	n/a	n/a	457.28
3	642.75	574.00	68.00	n/a	n/a	n/a	459.69
Texturing							
1	855.80	819.00	25.00	n/a	n/a	n/a	94.86
2	849.35	820.00	25.00	n/a	n/a	n/a	100.03
3	871.49	845.00	26.00	n/a	n/a	n/a	95.44
Texture Merge							
1	877.80	846.00	24.00	n/a	n/a	n/a	595.98
2	889.35	860.00	24.00	n/a	n/a	n/a	594.78
3	879.49	845.00	25.00	n/a	n/a	n/a	599.10
Texture Filter							
1	850.51	825.00	24.00	n/a	n/a	n/a	601.46
2	865.49	840.00	24.00	n/a	n/a	n/a	606.76
3	854.26	821.00	25.00	n/a	n/a	n/a	601.65

4.2.4.2.2 จำนวนเทรด 16

4.2.4.2.2.1 รูปที่1

รูป4.9 รูปที่ใช้ในการทดลอง

ตาราง 4.9 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	734.55	717.00	17.00	239.14	226.46	10.02	13.31
2	761.70	745.00	16.00	238.90	226.89	9.26	12.48
3	686.45	669.00	17.00	251.33	229.68	11.01	12.54
Sepia							
1	776.19	756.00	20.00	n/a	n/a	n/a	51.83
2	808.59	788.00	20.00	n/a	n/a	n/a	50.52
3	782.05	762.00	20.00	n/a	n/a	n/a	52.84
Hue Modifier							
1	773.01	745.00	28.00	296.39	248.50	13.98	492.82
2	813.00	789.00	24.00	312.96	247.28	20.48	493.52
3	764.79	739.00	25.00	302.34	247.56	15.14	492.52
Rotate Channel							
1	775.47	758.00	17.00	295.27	250.68	25.03	15.99
2	777.97	761.00	16.00	263.14	230.16	17.96	15.83
3	805.43	787.00	18.00	260.46	228.79	16.92	15.76

ตาราง 4.9 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	783.39	767.00	16.00	253.70	225.72	13.58	20.85
2	811.06	792.00	19.00	423.23	355.25	13.37	20.86
3	826.88	809.00	17.00	261.30	228.04	14.06	20.77
Color Reduction							
1	959.76	892.00	67.00	n/a	n/a	n/a	13337.78
2	893.95	826.00	67.00	n/a	n/a	n/a	13326.45
3	903.47	828.00	75.00	n/a	n/a	n/a	13222.22
Contrast Strecth							
1	535.51	518.00	17.00	n/a	n/a	n/a	17.75
2	550.84	533.00	17.00	n/a	n/a	n/a	18.27
3	562.41	544.00	18.00	n/a	n/a	n/a	17.72
Gamma Correction							
1	879.55	860.00	19.00	644.54	398.82	66.38	24.38
2	810.54	791.00	19.00	685.26	389.16	76.37	24.46
3	864.75	843.00	21.00	501.52	277.10	63.41	24.64
Brightness Correction							
1	816.20	797.00	19.00	254.23	226.05	13.72	28.26
2	828.53	807.00	21.00	256.20	227.16	13.88	27.30
3	776.60	756.00	20.00	262.18	233.76	14.09	26.93
Contrast Correction							
1	827.03	809.00	18.00	257.99	235.03	18.70	28.82
2	818.29	799.00	19.00	266.51	233.03	19.07	28.83
3	800.88	783.00	17.00	268.33	234.90	18.79	28.00
Saturation Correction							
1	789.75	755.00	34.00	283.93	242.02	14.00	576.90
2	726.34	694.00	32.00	281.50	239.37	12.93	607.90
3	735.05	702.00	33.00	280.56	237.23	13.05	579.11

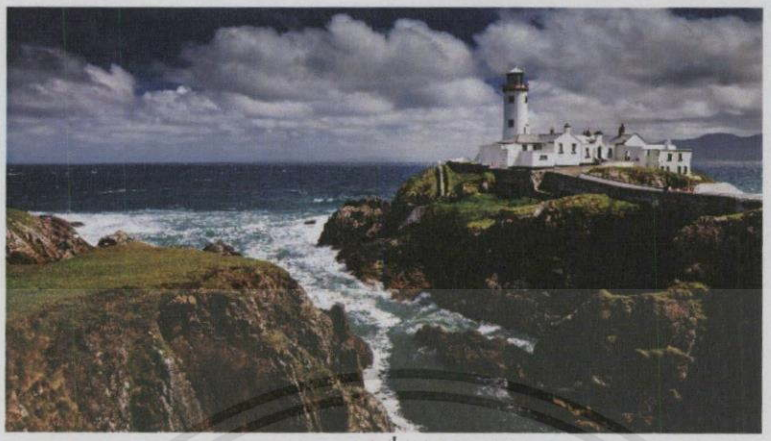
ตาราง 4.9 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	791.58	769.00	22.00	n/a	n/a	n/a	27.76
2	811.42	788.00	23.00	n/a	n/a	n/a	27.71
3	823.11	800.00	23.00	n/a	n/a	n/a	27.71
YCbCr Level Linear							
1	813.44	791.00	22.00	n/a	n/a	n/a	407.93
2	882.27	860.00	22.00	n/a	n/a	n/a	407.14
3	811.72	792.0	19.00	n/a	n/a	n/a	407.89
HSL Level Linear							
1	839.51	801.00	38.00	n/a	n/a	n/a	576.34
2	812.71	771.00	41.00	n/a	n/a	n/a	340.60
3	862.54	824.00	38.00	n/a	n/a	n/a	574.66
Blur							
1	986.89	881.00	105.00	759.55	530.28	68.89	1672.92
2	891.55	705.00	186.00	693.96	458.08	73.63	1672.79
3	852.21	754.00	98.00	611.97	377.24	71.52	1668.49
Edge							
1	955.64	897.00	58.00	598.83	272.07	79.64	790.47
2	955.25	901.00	54.00	520.84	290.94	66.11	788.60
3	978.46	922.00	56.00	521.70	282.69	66.51	790.36
Sharpen							
1	954.50	888.00	66.00	519.02	283.29	72.63	811.67
2	953.36	891.00	62.00	508.61	272.66	67.83	764.10
3	949.90	887.00	62.00	530.69	296.49	65.83	761.46
Sobel							
1	981.60	948.00	33.00	521.44	276.32	74.97	n/a
2	896.88	859.00	37.00	526.07	285.20	72.10	n/a
3	766.71	732.00	34.00	509.65	278.94	68.09	n/a

ตาราง 4.9 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	733.30	646.00	87.00	856.22	275.20	73.80	1670.52
2	848.67	764.00	84.00	658.97	272.47	71.41	1669.12
3	741.62	657.00	84.00	594.39	277.09	73.54	1670.30
Gaussian Blur							
1	851.78	720.00	131.00	263.85	230.22	18.74	769.74
2	815.73	689.00	126.00	60.51	228.04	17.17	763.34
3	810.63	680.00	130.00	282.46	232.12	20.52	770.12
Mean Filter							
1	791.22	723.00	68.00	507.73	276.87	67.16	767.71
2	767.06	700.00	67.00	507.68	278.78	64.88	766.11
3	797.33	730.00	67.00	504.37	277.05	64.06	772.76
Median Filter							
1	1128.82	707.00	421.00	n/a	n/a	n/a	2323.47
2	1163.56	741.00	422.00	n/a	n/a	n/a	2327.33
3	1158.70	737.00	421.00	n/a	n/a	n/a	2370.53
Conservative Smoothing							
1	955.78	806.00	149.00	n/a	n/a	n/a	477.04
2	864.83	800.00	64.00	n/a	n/a	n/a	509.69
3	758.93	695.00	63.00	n/a	n/a	n/a	492.25
Texturing							
1	835.79	817.00	13.00	n/a	n/a	n/a	95.11
2	850.47	834.00	13.00	n/a	n/a	n/a	94.24
3	842.91	825.00	13.00	n/a	n/a	n/a	96.91
Texture Merge							
1	854.40	825.00	25.00	n/a	n/a	n/a	598.68
2	883.56	848.00	26.00	n/a	n/a	n/a	600.22
3	857.85	815.00	25.00	n/a	n/a	n/a	609.70
Texture Filter							
1	860.12	825.00	25.00	n/a	n/a	n/a	619.13
2	884.60	848.00	26.00	n/a	n/a	n/a	615.16
3	859.32	821.00	25.00	n/a	n/a	n/a	613.96

4.2.4.2.1.2 รูปที่2



รูป4.10 รูปที่ใช้ในการทดลอง

ตาราง 4.10 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	439.03	419.00	20.00	130.83	119.04	8.98	13.36
2	408.98	392.00	16.00	131.23	119.68	8.60	12.50
3	464.45	445.00	19.00	133.97	122.14	8.94	13.05
Sepia							
1	407.03	388.00	19.00	n/a	n/a	n/a	53.72
2	453.80	434.00	19.00	n/a	n/a	n/a	54.00
3	513.24	495.00	18.00	n/a	n/a	n/a	53.68
Hue Modifier							
1	539.85	515.00	24.00	176.40	131.80	13.49	493.11
2	529.77	505.00	24.00	180.74	132.38	12.95	489.03
3	580.33	553.00	27.00	176.62	132.18	12.94	488.69
Rotate Channel							
1	521.87	500.00	21.00	154.41	122.04	17.07	15.50
2	532.74	515.00	17.00	154.00	121.23	18.17	16.48
3	518.95	502.00	16.00	154.41	122.36	17.08	16.19

ตาราง 4.10 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	536.76	517.00	19.00	146.56	117.91	13.49	21.49
2	540.16	522.00	18.00	147.47	117.57	14.30	21.80
3	576.36	556.00	20.00	152.17	121.39	12.83	21.44
Color Reduction							
1	599.01	531.00	68.00	n/a	n/a	n/a	13696.07
2	591.94	527.00	64.00	n/a	n/a	n/a	13714.35
3	618.53	552.00	63.00	n/a	n/a	n/a	13679.39
Contrast Strecth							
1	527.44	506.00	21.00	n/a	n/a	n/a	16.96
2	523.99	505.00	18.00	n/a	n/a	n/a	15.81
3	524.24	506.00	18.00	n/a	n/a	n/a	15.74
Gamma Correction							
1	550.87	533.00	17.00	398.35	168.64	64.11	23.47
2	509.30	491.00	18.00	405.60	175.27	61.70	24.28
3	555.63	535.00	20.00	399.56	169.85	65.74	23.79
Brightness Correction							
1	531.38	511.00	20.00	147.88	119.47	13.05	27.74
2	492.57	480.00	17.00	149.71	120.62	11.04	27.48
3	488.15	1472.00	16.00	150.16	119.13	14.59	27.30
Contrast Correction							
1	526.82	506.00	20.00	164.69	131.45	18.36	27.81
2	539.86	519.00	20.00	162.98	130.02	18.07	27.83
3	551.14	530.00	21.00	162.75	127.73	18.83	27.78
Saturation Correction							
1	515.94	483.00	32.00	232.87	130.52	14.18	571.35
2	529.12	495.00	34.00	181.37	135.20	14.06	566.63
3	563.31	526.00	37.00	181.77	134.59	13.86	572.62

ตาราง 4.10 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	534.73	510.00	24.00	n/a	n/a	n/a	27.93
2	504.27	482.00	22.00	n/a	n/a	n/a	27.80
3	537.37	518.00	19.00	n/a	n/a	n/a	27.40
YCbCr Level Linear							
1	508.77	484.00	24.00	n/a	n/a	n/a	399.82
2	557.13	533.00	24.00	n/a	n/a	n/a	412.96
3	543.39	523.00	20.00	n/a	n/a	n/a	398.84
HSL Level Linear							
1	506.67	467.00	39.00	n/a	n/a	n/a	574.65
2	637.46	598.00	39.00	n/a	n/a	n/a	579.50
3	601.70	563.00	38.00	n/a	n/a	n/a	576.94
Blur							
1	716.21	614.00	102.00	408.74	172.78	70.16	1725.81
2	684.55	581.00	103.00	463.62	224.13	71.75	1704.70
3	705.78	602.00	103.00	565.74	172.86	70.90	1715.04
Edge							
1	627.13	568.00	59.00	495.85	169.63	65.90	799.96
2	669.12	611.00	58.00	405.27	173.31	64.88	794.87
3	661.10	601.00	60.00	609.38	374.42	65.80	797.11
Sharpen							
1	683.85	617.00	66.00	538.59	305.30	67.02	769.63
2	667.61	602.00	65.00	489.70	172.55	70.86	769.60
3	676.17	610.00	66.00	406.44	171.25	70.95	769.47
Sobel							
1	557.55	524.00	33.00	510.44	275.47	66.83	n/a
2	632.91	599.00	33.00	565.21	170.51	66.91	n/a
3	600.85	563.00	37.00	504.81	178.28	69.29	n/a

ตาราง 4.10 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	722.83	635.00	87.00	406.42	196.85	71.94	1705.54
2	729.05	640.00	89.00	411.03	170.45	69.99	1701.35
3	513.97	427.00	83.00	407.34	171.46	70.76	1711.33
Gaussian Blur							
1	616.65	463.00	153.00	150.62	119.69	17.26	766.70
2	520.87	394.00	126.00	153.42	120.42	17.88	772.76
3	559.65	429.00	130.00	153.26	119.82	18.19	770.69
Mean Filter							
1	429.36	399.00	93.00	485.41	232.35	64.97	783.04
2	508.77	444.00	64.00	403.30	169.16	67.88	763.33
3	465.43	401.00	64.00	409.17	173.75	70.12	769.75
Median Filter							
1	863.59	428.00	435.00	n/a	n/a	n/a	2601.17
2	840.90	420.00	420.00	n/a	n/a	n/a	2604.60
3	884.79	464.00	420.00	n/a	n/a	n/a	2643.47
Conservative Smoothing							
1	514.72	420.00	94.00	n/a	n/a	n/a	495.05
2	483.74	421.00	62.00	n/a	n/a	n/a	457.28
3	452.43	388.00	64.00	n/a	n/a	n/a	459.69
Texturing							
1	888.72	873.00	13.00	n/a	n/a	n/a	94.86
2	901.84	881.00	15.00	n/a	n/a	n/a	100.03
3	854.91	837.00	15.00	n/a	n/a	n/a	95.44
Texture Merge							
1	865.22	825.00	25.00	n/a	n/a	n/a	595.98
2	840.07	828.00	26.00	n/a	n/a	n/a	594.78
3	849.33	821.00	25.00	n/a	n/a	n/a	599.10
Texture Filter							
1	850.15	815.00	25.00	n/a	n/a	n/a	601.46
2	844.66	828.00	26.00	n/a	n/a	n/a	606.76
3	869.56	841.00	25.00	n/a	n/a	n/a	601.65

4.2.4.2.3 จำนวนเทรดมาก

4.2.4.2.3.1 รูปที่1

รูป4.11 รูปที่ใช้ในการทดลอง

ตาราง 4.11 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	673.42	652.00	21.00	239.14	226.46	10.02	13.31
2	677.79	660.00	17.00	238.90	226.89	9.26	12.48
3	677.42	659.00	18.00	251.33	229.68	11.01	12.54
Sepia							
1	695.33	662.00	33.00	n/a	n/a	n/a	51.83
2	674.95	655.00	19.00	n/a	n/a	n/a	50.52
3	689.86	672.00	17.00	n/a	n/a	n/a	52.84
Hue Modifier							
1	672.46	631.00	41.00	296.39	248.50	13.98	492.82
2	671.03	648.00	23.00	312.96	247.28	20.48	493.52
3	657.13	634.00	23.00	302.34	247.56	15.14	492.52
Rotate Channel							
1	671.50	653.00	18.00	295.27	250.68	25.03	15.99
2	663.18	648.00	15.00	263.14	230.16	17.96	15.83
3	732.70	714.00	18.00	260.46	228.79	16.92	15.76

ตาราง 4.11 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	661.44	635.00	26.00	253.70	225.72	13.58	20.85
2	687.80	669.00	18.00	423.23	355.25	13.37	20.86
3	655.12	639.00	16.00	261.30	228.04	14.06	20.77
Color Reduction							
1	801.38	721.00	80.00	n/a	n/a	n/a	13337.78
2	722.41	659.00	63.00	n/a	n/a	n/a	13326.45
3	720.35	657.00	63.00	n/a	n/a	n/a	13222.22
Contrast Strecth							
1	503.93	484.00	19.00	n/a	n/a	n/a	17.75
2	530.16	513.00	17.00	n/a	n/a	n/a	18.27
3	483.18	466.00	17.00	n/a	n/a	n/a	17.72
Gamma Correction							
1	557.23	539.00	18.00	644.54	398.82	66.38	24.38
2	508.41	492.00	16.00	685.26	389.16	76.37	24.46
3	517.40	499.00	18.00	501.52	277.10	63.41	24.64
Brightness Correction							
1	496.14	476.00	20.00	254.23	226.05	13.72	28.26
2	499.93	480.00	19.00	256.20	227.16	13.88	27.30
3	504.92	485.00	19.00	262.18	233.76	14.09	26.93
Contrast Correction							
1	515.24	478.00	37.00	257.99	235.03	18.70	28.82
2	497.37	479.00	18.00	266.51	233.03	19.07	28.83
3	535.69	514.00	18.00	268.33	234.90	18.79	28.00
Saturation Correction							
1	832.01	773.00	59.00	283.93	242.02	14.00	576.90
2	816.62	781.00	35.00	281.50	239.37	12.93	607.90
3	779.98	745.00	34.00	280.56	237.23	13.05	579.11

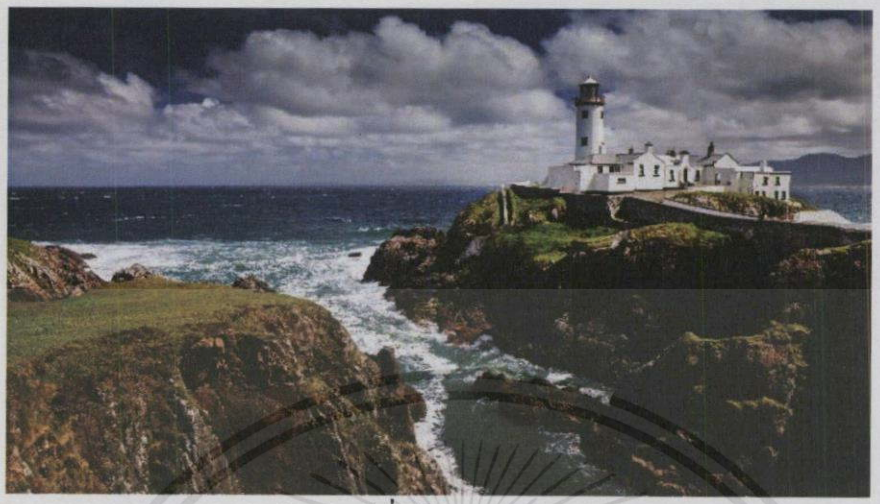
ตาราง 4.11 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	829.64	769.00	60.00	n/a	n/a	n/a	27.76
2	719.90	697.00	22.00	n/a	n/a	n/a	27.71
3	793.43	772.00	21.00	n/a	n/a	n/a	27.71
YCbCr Level Linear							
1	775.50	733.00	42.00	n/a	n/a	n/a	407.93
2	760.31	738.00	22.00	n/a	n/a	n/a	407.14
3	709.56	687.00	22.00	n/a	n/a	n/a	407.89
HSL Level Linear							
1	813.24	756.00	57.00	n/a	n/a	n/a	576.34
2	804.99	766.00	38.00	n/a	n/a	n/a	340.60
3	785.94	748.00	37.00	n/a	n/a	n/a	574.66
Blur							
1	909.28	795.00	114.00	759.55	530.28	68.89	1672.92
2	908.80	795.00	113.00	693.96	458.08	73.63	1672.79
3	941.40	826.00	115.00	611.97	377.24	71.52	1668.49
Edge							
1	886.19	791.00	95.00	598.83	272.07	79.64	790.47
2	847.31	780.00	67.00	520.84	290.94	66.11	788.60
3	873.76	807.00	66.00	521.70	282.69	66.51	790.36
Sharpen							
1	884.07	782.00	102.00	519.02	283.29	72.63	811.67
2	900.03	824.00	76.00	508.61	272.66	67.83	764.10
3	912.52	836.00	76.00	530.69	296.49	65.83	761.46
Sobel							
1	805.04	761.00	44.00	521.44	276.32	74.97	n/a
2	765.14	731.00	34.00	526.07	285.20	72.10	n/a
3	792.30	758.00	34.00	509.65	278.94	68.09	n/a

ตาราง 4.11 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	881.28	792.00	99.00	856.22	275.20	73.80	1670.52
2	903.09	804.00	99.00	658.97	272.47	71.41	1669.12
3	889.83	791.00	98.00	594.39	277.09	73.54	1670.30
Gaussian Blur							
1	963.24	817.00	146.00	263.85	230.22	18.74	769.74
2	981.82	836.00	145.00	60.51	228.04	17.17	763.34
3	967.28	822.00	145.00	282.46	232.12	20.52	770.12
Mean Filter							
1	903.69	827.00	76.00	507.73	276.87	67.16	767.71
2	878.41	801.00	77.00	507.68	278.78	64.88	766.11
3	881.24	805.00	76.00	504.37	277.05	64.06	772.76
Median Filter							
1	1219.34	796.00	423.00	n/a	n/a	n/a	2323.47
2	1245.21	825.00	420.00	n/a	n/a	n/a	2327.33
3	1213.30	794.00	419.00	n/a	n/a	n/a	2370.53
Conservative Smoothing							
1	894.69	830.00	64.00	n/a	n/a	n/a	477.04
2	858.21	792.00	66.00	n/a	n/a	n/a	509.69
3	841.34	777.00	64.00	n/a	n/a	n/a	492.25
Texturing							
1	874.19	854.00	16.00	n/a	n/a	n/a	95.11
2	887.86	869.00	14.00	n/a	n/a	n/a	94.24
3	846.42	829.00	15.00	n/a	n/a	n/a	96.91
Texture Merge							
1	826.22	795.00	25.00	n/a	n/a	n/a	598.68
2	876.70	848.00	26.00	n/a	n/a	n/a	600.22
3	861.52	833.00	24.00	n/a	n/a	n/a	609.70
Texture Filter							
1	861.25	825.00	24.00	n/a	n/a	n/a	619.13
2	889.60	848.00	25.00	n/a	n/a	n/a	615.16
3	863.02	836.00	25.00	n/a	n/a	n/a	613.96

4.2.4.2.3.2 รูปที่2



รูป4.12 รูปที่ใช้ในการทดลอง

ตาราง 4.12 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	620.49	600.00	20.00	130.83	119.04	8.98	13.36
2	520.84	500.00	20.00	131.23	119.68	8.60	12.50
3	612.63	592.00	20.00	133.97	122.14	8.94	13.05
Sepia							
1	552.64	534.00	18.00	n/a	n/a	n/a	53.72
2	565.64	543.00	22.00	n/a	n/a	n/a	54.00
3	570.15	551.00	19.00	n/a	n/a	n/a	53.68
Hue Modifier							
1	508.61	481.00	27.00	176.40	131.80	13.49	493.11
2	491.41	467.00	24.00	180.74	132.38	12.95	489.03
3	611.04	583.00	28.00	176.62	132.18	12.94	488.69
Rotate Channel							
1	448.81	428.00	14.00	154.41	122.04	17.07	15.50
2	517.52	502.00	15.00	154.00	121.23	18.17	16.48
3	541.01	526.00	14.00	154.41	122.36	17.08	16.19

ตาราง 4.12 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	571.72	553.00	18.00	146.56	117.91	13.49	21.49
2	530.13	512.00	18.00	147.47	117.57	14.30	21.80
3	515.36	498.00	17.00	152.17	121.39	12.83	21.44
Color Reduction							
1	596.02	526.00	70.00	n/a	n/a	n/a	13696.07
2	553.14	488.00	65.00	n/a	n/a	n/a	13714.35
3	472.25	407.00	65.00	n/a	n/a	n/a	13679.39
Contrast Strecth							
1	438.65	417.00	21.00	n/a	n/a	n/a	16.96
2	416.36	398.00	18.00	n/a	n/a	n/a	15.81
3	449.69	431.00	18.00	n/a	n/a	n/a	15.74
Gamma Correction							
1	416.80	396.00	20.00	398.35	168.64	64.11	23.47
2	418.84	400.00	18.00	405.60	175.27	61.70	24.28
3	501.81	483.00	18.00	399.56	169.85	65.74	23.79
Brightness Correction							
1	425.47	405.00	20.00	147.88	119.47	13.05	27.74
2	428.79	412.00	16.00	149.71	120.62	11.04	27.48
3	411.13	393.00	18.00	150.16	119.13	14.59	27.30
Contrast Correction							
1	420.35	399.00	21.00	164.69	131.45	18.36	27.81
2	404.08	387.00	17.00	162.98	130.02	18.07	27.83
3	418.91	401.00	17.00	162.75	127.73	18.83	27.78
Saturation Correction							
1	439.08	402.00	37.00	232.87	130.52	14.18	571.35
2	454.99	420.00	34.00	181.37	135.20	14.06	566.63
3	427.94	394.00	33.00	181.77	134.59	13.86	572.62

ตาราง 4.12 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	435.04	412.00	23.00	n/a	n/a	n/a	27.93
2	422.52	407.00	15.00	n/a	n/a	n/a	27.80
3	468.00	450.00	18.00	n/a	n/a	n/a	27.40
YCbCr Level Linear							
1	424.66	402.00	22.00	n/a	n/a	n/a	399.82
2	403.56	386.00	17.00	n/a	n/a	n/a	412.96
3	437.06	418.00	16.00	n/a	n/a	n/a	398.84
HSL Level Linear							
1	443.82	420.00	23.00	n/a	n/a	n/a	574.65
2	408.75	388.00	20.00	n/a	n/a	n/a	579.50
3	415.92	393.00	22.00	n/a	n/a	n/a	576.94
Blur							
1	538.10	418.00	120.00	408.74	172.78	70.16	1725.81
2	528.38	416.00	112.00	463.62	224.13	71.75	1704.70
3	510.45	398.00	112.00	565.74	172.86	70.90	1715.04
Edge							
1	558.31	489.00	69.00	495.85	169.63	65.90	799.96
2	621.75	555.00	66.00	405.27	173.31	64.88	794.87
3	605.87	541.00	64.00	609.38	374.42	65.80	797.11
Sharpen							
1	640.16	562.00	78.00	538.59	305.30	67.02	769.63
2	624.98	549.00	75.00	489.70	172.55	70.86	769.60
3	612.39	539.00	73.00	406.44	171.25	70.95	769.47
Sobel							
1	526.37	487.00	39.00	510.44	275.47	66.83	n/a
2	554.11	519.00	35.00	565.21	170.51	66.91	n/a
3	516.96	482.00	34.00	504.81	178.28	69.29	n/a

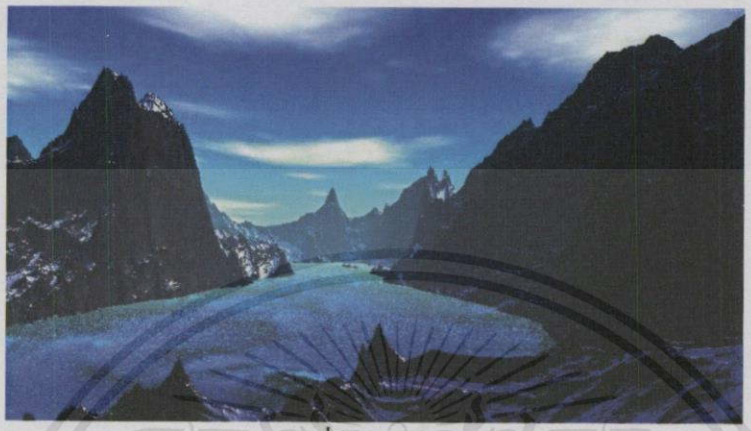
ตาราง 4.12 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	666.34	569.00	97.00	406.42	196.85	71.94	1705.54
2	617.74	521.00	96.00	411.03	170.45	69.99	1701.35
3	637.68	540.00	97.00	407.34	171.46	70.76	1711.33
Gaussian Blur							
1	699.63	553.00	146.00	150.62	119.69	17.26	766.70
2	708.89	562.00	146.00	153.42	120.42	17.88	772.76
3	702.29	556.00	146.00	153.26	119.82	18.19	770.69
Mean Filter							
1	628.53	550.00	78.00	485.41	232.35	64.97	783.04
2	714.35	638.00	76.00	403.30	169.16	67.88	763.33
3	622.66	547.00	75.00	409.17	173.75	70.12	769.75
Median Filter							
1	1001.00	578.00	423.00	n/a	n/a	n/a	2601.17
2	955.66	535.00	420.00	n/a	n/a	n/a	2604.60
3	1008.22	586.00	422.00	n/a	n/a	n/a	2643.47
Conservative Smoothing							
1	530.38	462.00	68.00	n/a	n/a	n/a	495.05
2	551.39	485.00	66.00	n/a	n/a	n/a	457.28
3	588.05	520.00	68.00	n/a	n/a	n/a	459.69
Texturing							
1	638.56	615.00	20.00	n/a	n/a	n/a	94.86
2	578.63	561.00	15.00	n/a	n/a	n/a	100.03
3	621.65	604.00	13.00	n/a	n/a	n/a	95.44
Texture Merge							
1	860.15	825.00	24.00	n/a	n/a	n/a	595.98
2	889.60	848.00	25.00	n/a	n/a	n/a	594.78
3	859.42	821.00	25.00	n/a	n/a	n/a	599.10
Texture Filter							
1	870.25	841.00	25.00	n/a	n/a	n/a	601.46
2	844.66	814.00	26.00	n/a	n/a	n/a	606.76
3	859.56	821.00	25.00	n/a	n/a	n/a	601.65

4.2.4.3 ภาพขนาด 7680*4320 (8K HD)

4.2.4.3.1 จำนวนเทรดน้อย

4.2.4.3.1.1 รูปที่1



รูป4.13 รูปที่ใช้ในการทดลอง

ตาราง 4.13 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	2348.67	2311.00	37.00	611.02	562.21	37.28	52.50
2	2032.28	1997.00	35.00	601.18	559.71	29.95	49.29
3	2114.06	2080.00	34.00	597.36	555.75	30.08	50.11
Sepia							
1	2480.04	2435.00	45.00	n/a	n/a	n/a	196.41
2	2535.96	2498.00	37.00	n/a	n/a	n/a	193.24
3	2472.68	2433.00	39.00	n/a	n/a	n/a	191.91
Hue Modifier							
1	2142.53	2100.00	42.00	806.45	640.48	48.67	1918.46
2	2125.29	2082.00	43.00	775.67	606.49	50.82	1917.92
3	2088.83	2050.00	38.00	786.92	605.11	61.34	1920.87
Rotate Channel							
1	2471.39	2433.00	39.00	697.25	569.84	68.26	61.59
2	2319.78	2280.00	40.00	693.73	566.05	68.36	61.60
3	2455.05	2419.00	40.00	695.89	567.43	64.76	61.54

ตาราง 4.13 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	2474.74	2433.00	41.00	707.14	600.21	48.72	81.29
2	2319.78	2280.00	39.00	662.79	556.55	47.44	82.00
3	2455.05	2419.00	36.00	662.21	556.83	47.86	81.78
Color Reduction							
1	2738.63	2654.00	84.00	n/a	n/a	n/a	n/a
2	2580.19	2492.00	88.00	n/a	n/a	n/a	n/a
3	2636.11	2547.00	89.00	n/a	n/a	n/a	n/a
Contrast Strecth							
1	2519.85	2479.00	40.00	n/a	n/a	n/a	72.36
2	2570.63	2531.00	39.00	n/a	n/a	n/a	65.10
3	2676.40	2561.00	115.00	n/a	n/a	n/a	68.48
Gamma Correction							
1	2501.46	2466.00	35.00	1661.96	754.63	247.98	91.42
2	2144.47	2102.00	42.00	1661.55	742.58	245.84	90.59
3	2069.33	2037.00	32.00	1973.18	744.87	244.85	94.03
Brightness Correction							
1	2178.16	2138.00	40.00	690.70	581.50	52.38	114.19
2	2065.62	2031.00	34.00	673.40	566.91	48.94	105.30
3	2113.96	2082.00	31.00	670.85	561.73	51.19	105.24
Contrast Correction							
1	2119.53	2079.00	40.00	730.17	604.18	68.02	106.58
2	2242.72	2201.00	41.00	779.01	613.56	77.24	104.07
3	2339.69	2297.00	42.00	733.06	602.51	72.60	103.10
Saturation Correction							
1	2068.88	2016.00	52.00	922.70	757.49	53.07	2257.17
2	2196.37	2140.00	56.00	827.88	616.85	52.03	2264.91
3	2384.91	2329.00	55.00	777.69	612.55	51.14	2260.45

ตาราง 4.13 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	2344.76	2304.00	40.00	n/a	n/a	n/a	104.26
2	2441.48	2399.00	42.00	n/a	n/a	n/a	104.63
3	2323.28	2286.00	37.00	n/a	n/a	n/a	105.79
YCbCr Level Linear							
1	2300.58	2256.00	44.00	n/a	n/a	n/a	1602.93
2	2363.72	2319.00	44.00	n/a	n/a	n/a	1606.54
3	2409.14	2365.00	44.00	n/a	n/a	n/a	1610.52
HSL Level Linear							
1	2374.57	2328.00	46.00	n/a	n/a	n/a	2252.33
2	2540.39	2497.00	43.00	n/a	n/a	n/a	2252.96
3	2227.59	2188.00	39.00	n/a	n/a	n/a	2260.32
Blur							
1	2740.79	2614.00	126.00	1913.86	752.70	287.69	6710.00
2	2803.15	2680.00	123.00	1735.94	757.60	307.66	6712.89
3	2731.63	2610.00	121.00	1736.67	790.33	290.33	6712.51
Edge							
1	2818.94	2768.00	50.00	1938.26	756.84	255.37	3161.13
2	2665.29	2611.00	54.00	1702.72	767.35	266.93	3167.57
3	2766.25	2718.00	48.00	2174.04	761.01	266.04	3170.78
Sharpen							
1	2686.08	2624.00	62.00	1856.86	760.80	320.57	3042.84
2	2754.81	2697.00	57.00	1747.13	819.34	263.59	3054.05
3	2626.73	2572.00	54.00	1696.94	762.89	262.19	3043.33
Sobel							
1	2407.00	2348.00	59.00	1762.45	820.20	269.65	n/a
2	2531.24	2478.00	53.00	1694.85	754.42	269.42	n/a
3	2477.54	2423.00	54.00	1729.28	765.22	293.65	n/a

ตาราง 4.13 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	2874.69	2765.00	109.00	1747.52	754.26	289.88	6710.09
2	2908.53	2802.00	106.00	1908.85	740.81	494.10	6706.97
3	2785.59	2682.00	103.00	1976.56	746.87	564.92	6872.42
Gaussian Blur							
1	2993.47	2815.00	178.00	694.09	566.18	64.30	3027.23
2	2763.80	2537.00	226.00	965.74	562.07	73.85	3099.79
3	2811.76	2675.00	136.00	713.43	560.70	61.46	3027.67
Mean Filter							
1	2966.65	2884.00	82.00	1877.95	765.84	429.84	3068.32
2	2618.30	2570.00	48.00	1676.06	751.21	266.29	3069.24
3	2186.75	2140.00	46.00	1686.33	753.07	266.67	3050.88
Median Filter							
1	4414.60	3925.00	489.00	n/a	n/a	n/a	9913.79
2	5045.58	4599.00	446.00	n/a	n/a	n/a	9987.80
3	4730.63	4292.00	438.00	n/a	n/a	n/a	9877.06
Conservative Smoothing							
1	2498.67	2387.00	111.00	n/a	n/a	n/a	1895.92
2	2679.59	2588.00	91.00	n/a	n/a	n/a	1899.41
3	2489.04	2398.00	91.00	n/a	n/a	n/a	1898.94
Texturing							
1	2881.01	2838.00	16.00	n/a	n/a	n/a	380.77
2	2845.16	2831.00	15.00	n/a	n/a	n/a	377.68
3	2862.36	2845.00	14.00	n/a	n/a	n/a	375.47
Texture Merge							
1	3291.11	3230.00	26.00	n/a	n/a	n/a	2305.02
2	3380.60	3325.00	25.00	n/a	n/a	n/a	2312.34
3	3390.67	3325.00	24.00	n/a	n/a	n/a	2334.45
Texture Filter							
1	3728.92	3681.00	26.00	n/a	n/a	n/a	2348.90
2	2790.32	2732.00	25.00	n/a	n/a	n/a	2422.92
3	3353.42	3299.00	25.00	n/a	n/a	n/a	2349.96

4.2.4.3.1.2 รูปที่2

รูป4.14 รูปที่ใช้ในการทดลอง

ตาราง 4.14 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	2471.16	2431.00	40.00	509.28	462.02	33.41	47.43
2	2473.84	2432.00	41.00	505.58	464.36	29.98	49.08
3	2509.26	2473.00	36.00	504.08	462.92	29.64	48.12
Sepia							
1	2505.43	2436.00	69.00	n/a	n/a	n/a	179.18
2	2487.93	2446.00	41.00	n/a	n/a	n/a	182.36
3	2459.30	2417.00	42.00	n/a	n/a	n/a	181.10
Hue Modifier							
1	2546.43	2480.00	66.00	925.14	514.08	288.68	1865.82
2	2377.78	2326.00	51.00	675.01	506.74	50.28	1864.93
3	459.41	2413.00	46.00	1474.41	505.81	49.25	1864.95
Rotate Channel							
1	2419.69	2380.00	39.00	662.94	532.89	68.24	61.04
2	2485.28	2443.00	42.00	596.66	470.32	64.92	61.78
3	2369.55	2330.00	39.00	598.62	473.00	67.03	60.26

ตาราง 4.14 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	2492.48	2447.00	45.00	573.31	465.39	50.18	78.88
2	2419.35	2418.00	45.00	575.34	464.50	53.08	80.32
3	2459.78	2417.00	42.00	568.45	461.80	48.82	78.21
Color Reduction							
1	2494.44	2387.00	107.00	n/a	n/a	n/a	n/a
2	2379.74	2295.00	84.00	n/a	n/a	n/a	n/a
3	2478.61	2393.00	85.00	n/a	n/a	n/a	n/a
Contrast Strecth							
1	2230.76	2180.00	50.00	n/a	n/a	n/a	76.67
2	2361.45	2180.00	41.00	n/a	n/a	n/a	68.48
3	2252.79	2209.00	43.00	n/a	n/a	n/a	69.36
Gamma Correction							
1	2396.64	2356.00	40.00	1764.98	662.81	435.34	92.32
2	2504.99	2464.00	40.00	1715.39	805.94	250.93	90.69
3	2457.27	2416.00	41.00	2229.12	668.85	256.65	90.93
Brightness Correction							
1	2869.08	2788.00	81.00	578.87	465.04	55.29	105.12
2	2636.02	2594.00	42.00	568.77	465.61	48.66	103.68
3	2566.40	2528.00	38.00	576.59	459.78	48.63	104.15
Contrast Correction							
1	2466.42	2406.00	60.00	635.13	503.21	72.43	107.03
2	2843.17	2795.00	48.00	627.47	501.04	68.15	102.93
3	2634.48	2599.00	35.00	630.79	502.93	69.10	104.68
Saturation Correction							
1	2897.19	2834.00	63.00	945.29	531.08	281.46	2091.05
2	2647.48	2600.00	47.00	671.03	509.33	52.55	2091.05
3	2142.58	2076.00	66.00	843.89	507.84	60.85	2093.26

ตาราง 4.14 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	2040.50	1997.00	43.00	n/a	n/a	n/a	105.16
2	1973.02	1937.00	36.00	n/a	n/a	n/a	105.06
3	1995.34	1959.00	36.00	n/a	n/a	n/a	104.22
YCbCr Level Linear							
1	2351.64	2296.00	55.00	n/a	n/a	n/a	1573.53
2	2457.54	2416.00	41.00	n/a	n/a	n/a	1591.97
3	2399.42	2357.00	42.00	n/a	n/a	n/a	1594.88
HSL Level Linear							
1	2436.15	2375.00	61.00	n/a	n/a	n/a	2121.39
2	2352.36	2306.00	46.00	n/a	n/a	n/a	2111.75
3	2397.70	2349.00	48.00	n/a	n/a	n/a	2152.67
Blur							
1	2667.45	2510.00	157.00	1729.51	759.51	290.68	6680.80
2	2699.36	2584.00	115.00	1813.63	846.62	285.29	6670.41
3	2737.92	2606.00	131.00	2039.28	674.04	283.63	6670.75
Edge							
1	2764.37	2710.00	54.00	1987.99	673.42	267.42	3172.45
2	2601.69	2546.00	55.00	1756.83	680.83	388.66	3179.75
3	2757.90	2704.00	54.00	1693.59	716.46	272.80	3181.47
Sharpen							
1	2703.57	2708.75	88.00	1660.28	674.95	276.46	3020.56
2	2611.79	2553.00	58.00	1603.95	657.93	260.18	3208.31
3	2708.75	2658.00	50.00	1609.13	642.85	255.22	3021.39
Sobel							
1	2360.37	2283.00	77.00	3041.14	1839.33	268.04	n/a
2	2320.47	2266.00	54.00	1698.08	654.23	269.42	n/a
3	2325.18	2272.00	53.00	1624.80	667.25	293.65	n/a

ตาราง 4.14 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	2693.95	2603.00	118.00	1747.52	754.26	289.88	6710.09
2	2563.28	2375.00	203.00	1908.85	740.81	377.27	6706.97
3	2898.52	2683.00	103.00	1976.56	746.87	275.43	6872.42
Gaussian Blur							
1	2624.81	2378.00	246.00	613.75	494.43	61.28	3197.78
2	2848.46	2710.00	138.00	595.84	467.51	66.21	3088.85
3	2777.35	2726.00	156.00	589.48	467.51	69.03	3093.57
Mean Filter							
1	2863.59	2764.00	99.00	2123.32	662.37	270.52	3063.58
2	2199.78	2149.00	50.00	1658.23	719.42	273.05	3145.53
3	2425.14	2372.00	53.00	1623.21	673.06	266.23	3105.98
Median Filter							
1	4998.42	4532.00	466.00	n/a	n/a	n/a	9150.82
2	4925.67	4480.00	445.00	n/a	n/a	n/a	9185.75
3	4631.44	4180.00	451.00	n/a	n/a	n/a	9271.33
Conservative Smoothing							
1	2436.13	2333.00	103.00	n/a	n/a	n/a	1835.50
2	2527.26	2442.00	85.00	n/a	n/a	n/a	1793.62
3	2416.27	2329.00	87.00	n/a	n/a	n/a	1818.16
Texturing							
1	2881.01	2838.00	16.00	n/a	n/a	n/a	397.96
2	2845.16	2831.00	15.00	n/a	n/a	n/a	423.14
3	2862.36	2845.00	14.00	n/a	n/a	n/a	400.65
Texture Merge							
1	3736.18	3689.00	24.00	n/a	n/a	n/a	2312.64
2	2655.35	2604.00	25.00	n/a	n/a	n/a	2306.44
3	2815.00	2765.00	24.00	n/a	n/a	n/a	2307.61
Texture Filter							
1	3291.11	3230.00	26.00	n/a	n/a	n/a	2049.62
2	3380.60	3325.00	25.00	n/a	n/a	n/a	2351.54
3	3390.67	3325.00	24.00	n/a	n/a	n/a	2351.01

4.2.4.3.2 จำนวนเทรดปานกลาง

4.2.4.3.2.1 รูปที่1

รูป4.15 รูปที่ใช้ในการทดลอง

ตาราง 4.15 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	2441.46	2402.00	39.00	611.02	562.21	37.28	52.50
2	2328.62	2282.00	46.00	601.18	559.71	29.95	49.29
3	2447.79	2405.00	42.00	597.36	555.75	30.08	50.11
Sepia							
1	2432.93	2390.00	42.00	n/a	n/a	n/a	196.41
2	2457.18	2415.00	42.00	n/a	n/a	n/a	193.24
3	2313.96	2269.00	44.00	n/a	n/a	n/a	191.91
Hue Modifier							
1	2434.79	2379.00	55.00	806.45	640.48	48.67	1918.46
2	2434.60	2386.00	48.00	775.67	606.49	50.82	1917.92
3	2440.58	2390.00	50.00	786.92	605.11	61.34	1920.87
Rotate Channel							
1	2417.29	2374.00	43.00	697.25	569.84	68.26	61.59
2	2363.35	2321.00	42.00	693.73	566.05	68.36	61.60
3	2412.38	2374.00	38.00	695.89	567.43	64.76	61.54

ตาราง 4.15 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	2324.01	2275.00	49.00	707.14	600.21	48.72	81.29
2	2439.63	2396.00	43.00	662.79	556.55	47.44	82.00
3	2398.48	2357.00	41.00	662.21	556.83	47.86	81.78
Color Reduction							
1	2493.50	2374.00	84.00	n/a	n/a	n/a	n/a
2	2344.08	2255.00	88.00	n/a	n/a	n/a	n/a
3	2417.54	2326.00	89.00	n/a	n/a	n/a	n/a
Contrast Strecth							
1	2284.68	2231.00	53.00	n/a	n/a	n/a	72.36
2	2252.14	2211.00	41.00	n/a	n/a	n/a	65.10
3	2276.03	2233.00	43.00	n/a	n/a	n/a	68.48
Gamma Correction							
1	2408.22	2365.00	43.00	1661.96	754.63	247.98	91.42
2	2679.11	2648.00	31.00	1661.55	742.58	245.84	90.59
3	2520.75	2489.00	31.00	1973.18	744.87	244.85	94.03
Brightness Correction							
1	2624.96	2580.00	44.00	690.70	581.50	52.38	114.19
2	2059.90	2024.00	35.00	673.40	566.91	48.94	105.30
3	1998.12	1959.00	39.00	670.85	561.73	51.19	105.24
Contrast Correction							
1	2082.46	2030.00	52.00	730.17	604.18	68.02	106.58
2	2152.03	2107.00	45.00	779.01	613.56	77.24	104.07
3	2525.55	2482.00	43.00	733.06	602.51	72.60	103.10
Saturation Correction							
1	2461.70	2390.00	71.00	922.70	757.49	53.07	2257.17
2	2438.14	2382.00	56.00	827.88	616.85	52.03	2264.91
3	2373.33	2313.00	60.00	777.69	612.55	51.14	2260.45

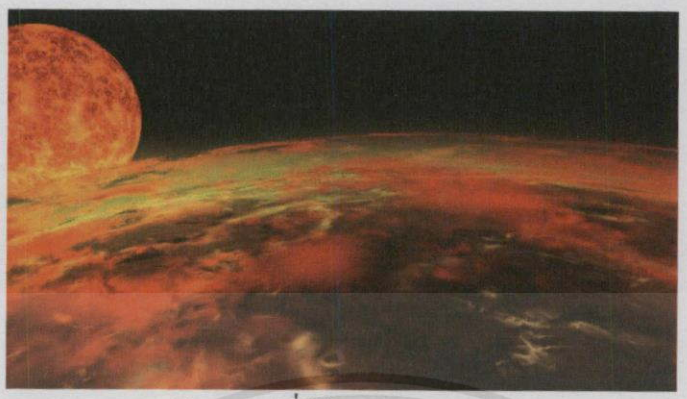
ตาราง 4.15 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	2504.71	2447.00	57.00	n/a	n/a	n/a	104.26
2	2244.75	2205.00	39.00	n/a	n/a	n/a	104.63
3	2469.33	2431.00	38.00	n/a	n/a	n/a	105.79
YCbCr Level Linear							
1	2291.40	2231.00	60.00	n/a	n/a	n/a	1602.93
2	2395.63	2347.00	48.00	n/a	n/a	n/a	1606.54
3	2274.85	2228.00	46.00	n/a	n/a	n/a	1610.52
HSL Level Linear							
1	2347.79	2296.00	51.00	n/a	n/a	n/a	2252.33
2	2357.08	2309.00	48.00	n/a	n/a	n/a	2252.96
3	2345.92	2304.00	41.00	n/a	n/a	n/a	2260.32
Blur							
1	2670.35	2516.00	154.00	1913.86	752.70	287.69	6710.00
2	2718.06	2587.00	131.00	1735.94	757.60	307.66	6712.89
3	2676.18	2551.00	125.00	1736.67	790.33	290.33	6712.51
Edge							
1	2682.36	2576.00	106.00	1938.26	756.84	255.37	3161.13
2	2604.90	2519.00	85.00	1702.72	767.35	266.93	3167.57
3	2690.99	2602.00	88.00	2174.04	761.01	266.04	3170.78
Sharpen							
1	2566.75	2454.00	112.00	1856.86	760.80	320.57	3042.84
2	2591.78	2504.00	87.00	1747.13	819.34	263.59	3054.05
3	2465.83	2373.00	92.00	1696.94	762.89	262.19	3043.33
Sobel							
1	2338.74	2268.00	70.00	1762.45	820.20	269.65	n/a
2	2319.74	2265.00	54.00	1694.85	754.42	269.42	n/a
3	2507.60	2452.00	55.00	1729.28	765.22	293.65	n/a

ตาราง 4.15 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	2721.20	2603.00	118.00	1747.52	754.26	289.88	6710.09
2	2578.48	2375.00	203.00	1908.85	740.81	494.10	6706.97
3	2786.16	2683.00	103.00	1976.56	746.87	564.92	6872.42
Gaussian Blur							
1	2855.52	2690.00	165.00	694.09	566.18	64.30	3027.23
2	2832.87	2682.00	150.00	965.74	562.07	73.85	3099.79
3	2289.76	2140.00	149.00	713.43	560.70	61.46	3027.67
Mean Filter							
1	2339.72	2220.00	119.00	1877.95	765.84	429.84	3068.32
2	2711.64	2617.00	94.00	1676.06	751.21	266.29	3069.24
3	2756.59	2661.00	95.00	1686.33	753.07	266.67	3050.88
Median Filter							
1	3082.25	2624.00	458.00	n/a	n/a	n/a	9913.79
2	2950.97	2509.00	441.00	n/a	n/a	n/a	9987.80
3	3059.80	2618.00	441.00	n/a	n/a	n/a	9877.06
Conservative Smoothing							
1	2480.07	2498.64	97.00	n/a	n/a	n/a	1895.92
2	2428.63	2342.00	86.00	n/a	n/a	n/a	1899.41
3	2498.64	2411.00	87.00	n/a	n/a	n/a	1898.94
Texturing							
1	2881.01	2838.00	16.00	n/a	n/a	n/a	380.77
2	2845.16	2831.00	15.00	n/a	n/a	n/a	377.68
3	2862.36	2845.00	14.00	n/a	n/a	n/a	375.47
Texture Merge							
1	3736.18	3689.00	24.00	n/a	n/a	n/a	2305.02
2	2655.35	2604.00	25.00	n/a	n/a	n/a	2312.34
3	2815.00	2765.00	24.00	n/a	n/a	n/a	2334.45
Texture Filter							
1	3353.42	3299.00	25.00	n/a	n/a	n/a	2348.90
2	3323.49	3272.00	26.00	n/a	n/a	n/a	2422.92
3	3374.04	3327.00	26.00	n/a	n/a	n/a	2349.96

4.2.4.3.2.2 รูปที่2



รูป4.16 รูปที่ใช้ในการทดลอง

ตาราง 4.16 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	2274.89	2227.00	47.00	509.28	462.02	33.41	47.43
2	2314.54	2269.00	45.00	505.58	464.36	29.98	49.08
3	2263.76	2222.00	41.00	504.08	462.92	29.64	48.12
Sepia							
1	2308.82	2229.00	79.00	n/a	n/a	n/a	179.18
2	2439.26	2394.00	45.00	n/a	n/a	n/a	182.36
3	2267.94	2226.00	41.00	n/a	n/a	n/a	181.10
Hue Modifier							
1	2377.69	2322.00	55.00	925.14	514.08	288.68	1865.82
2	2443.26	2400.00	43.00	675.01	506.74	50.28	1864.93
3	2471.33	2420.00	51.00	1474.41	505.81	49.25	1864.95
Rotate Channel							
1	2362.27	2327.00	35.00	662.94	532.89	68.24	61.04
2	2373.74	2330.00	43.00	596.66	470.32	64.92	61.78
3	2393.27	2349.00	44.00	598.62	473.00	67.03	60.26

ตาราง 4.16 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	2404.31	2351.00	53.00	573.31	465.39	50.18	78.88
2	2343.99	2310.00	33.00	575.34	464.50	53.08	80.32
3	2345.12	2304.00	41.00	568.45	461.80	48.82	78.21
Color Reduction							
1	2398.59	2243.00	155.00	n/a	n/a	n/a	n/a
2	2571.15	2487.00	84.00	n/a	n/a	n/a	n/a
3	2329.18	2246.00	83.00	n/a	n/a	n/a	n/a
Contrast Strecth							
1	2213.65	2168.00	45.00	n/a	n/a	n/a	76.67
2	2323.25	2279.00	44.00	n/a	n/a	n/a	68.48
3	2549.46	2506.00	43.00	n/a	n/a	n/a	69.36
Gamma Correction							
1	2699.54	2663.00	36.00	1764.98	662.81	435.34	92.32
2	2720.84	2683.00	37.00	1715.39	805.94	250.93	90.69
3	2684.66	2641.00	43.00	2229.12	668.85	256.65	90.93
Brightness Correction							
1	2231.93	2177.00	54.00	578.87	465.04	55.29	105.12
2	1983.68	1948.00	35.00	568.77	465.61	48.66	103.68
3	1989.78	1953.00	36.00	576.59	459.78	48.63	104.15
Contrast Correction							
1	2079.79	2022.00	57.00	635.13	503.21	72.43	107.03
2	2005.45	1963.00	42.00	627.47	501.04	68.15	102.93
3	2246.24	2204.00	42.00	630.79	502.93	69.10	104.68
Saturation Correction							
1	2386.72	2313.00	73.00	945.29	531.08	281.46	2091.05
2	2356.41	2298.00	58.00	671.03	509.33	52.55	2091.05
3	2344.35	2287.00	57.00	843.89	507.84	60.85	2093.26

ตาราง 4.16 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

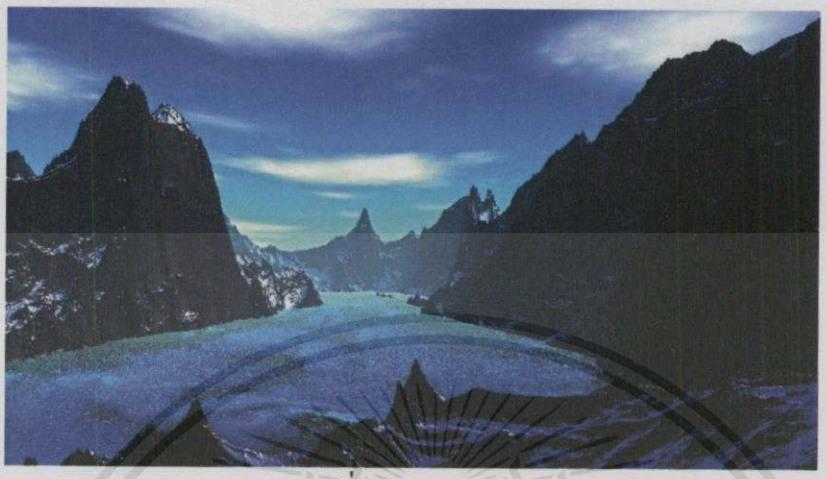
ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	2357.64	2282.00	75.00	n/a	n/a	n/a	105.16
2	2279.73	2237.00	42.00	n/a	n/a	n/a	105.06
3	2353.16	2313.00	40.00	n/a	n/a	n/a	104.22
YCbCr Level Linear							
1	2377.68	2317.00	60.00	n/a	n/a	n/a	1573.53
2	2311.89	2267.00	44.00	n/a	n/a	n/a	1591.97
3	2315.42	2270.00	45.00	n/a	n/a	n/a	1594.88
HSL Level Linear							
1	2334.24	2270.00	64.00	n/a	n/a	n/a	2121.39
2	2469.94	2425.00	44.00	n/a	n/a	n/a	2111.75
3	2302.16	2257.00	45.00	n/a	n/a	n/a	2152.67
Blur							
1	2760.57	2603.00	157.00	1729.51	759.51	290.68	6680.80
2	2671.21	2541.00	130.00	1813.63	846.62	285.29	6670.41
3	2757.21	2626.00	131.00	2039.28	674.04	283.63	6670.75
Edge							
1	2576.44	2477.00	99.00	1987.99	673.42	267.42	3172.45
2	2517.23	2432.00	85.00	1756.83	680.83	388.66	3179.75
3	2575.63	2493.00	82.00	1693.59	716.46	272.80	3181.47
Sharpen							
1	2539.24	2432.00	107.00	1660.28	674.95	276.46	3020.56
2	2583.66	2492.00	91.00	1603.95	657.93	260.18	3208.31
3	2568.77	2475.00	93.00	1609.13	642.85	255.22	3021.39
Sobel							
1	2358.13	2278.00	80.00	3041.14	1839.33	268.04	n/a
2	2417.33	2362.00	55.00	1698.08	654.23	269.42	n/a
3	2365.11	2310.00	55.00	1624.80	667.25	293.65	n/a

ตาราง 4.16 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	3143.55	3006.00	137.00	1747.52	754.26	289.88	6710.09
2	2765.22	2652.00	113.00	1908.85	740.81	377.27	6706.97
3	2615.49	2515.00	100.00	1976.56	746.87	275.43	6872.42
Gaussian Blur							
1	2860.21	2691.00	169.00	613.75	494.43	61.28	3197.78
2	2738.60	2593.00	145.00	595.84	467.51	66.21	3088.85
3	2340.28	2190.00	150.00	589.48	467.51	69.03	3093.57
Mean Filter							
1	2396.93	2270.00	126.00	2123.32	662.37	270.52	3063.58
2	2629.06	2534.0	95.00	1658.23	719.42	273.05	3145.53
3	2779.53	2684.00	95.00	1623.21	673.06	266.23	3105.98
Median Filter							
1	2885.12	2415.00	470.00	n/a	n/a	n/a	9150.82
2	2982.02	2532.00	450.00	n/a	n/a	n/a	9185.75
3	2910.04	2462.00	448.00	n/a	n/a	n/a	9271.33
Conservative Smoothing							
1	2515.68	2415.00	100.00	n/a	n/a	n/a	1835.50
2	2504.48	2417.00	87.00	n/a	n/a	n/a	1793.62
3	2465.06	2372.00	93.00	n/a	n/a	n/a	1818.16
Texturing							
1	2881.01	2838.00	16.00	n/a	n/a	n/a	397.96
2	2855.16	2839.00	15.00	n/a	n/a	n/a	423.14
3	2863.32	2845.00	14.00	n/a	n/a	n/a	400.65
Texture Merge							
1	3728.92	3681.00	26.00	n/a	n/a	n/a	2312.64
2	2790.32	2732.00	25.00	n/a	n/a	n/a	2306.44
3	3353.42	3299.00	25.00	n/a	n/a	n/a	2307.61
Texture Filter							
1	3353.42	3299.00	25.00	n/a	n/a	n/a	2049.62
2	3323.49	3272.00	26.00	n/a	n/a	n/a	2351.54
3	3374.04	3327.00	26.00	n/a	n/a	n/a	2351.01

4.2.4.3.3 จำนวนเทรด 32

4.2.4.3.3.1 รูปที่1



รูป4.17 รูปที่ใช้ในการทดลอง

ตาราง 4.17 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	2499.22	2455.00	44.00	611.02	562.21	37.28	52.50
2	2297.25	2252.00	45.00	601.18	559.71	29.95	49.29
3	2451.24	2420.00	31.00	597.36	555.75	30.08	50.11
Sepia							
1	2292.69	2226.00	66.00	n/a	n/a	n/a	196.41
2	2361.22	2317.00	44.00	n/a	n/a	n/a	193.24
3	2295.13	2251.00	44.00	n/a	n/a	n/a	191.91
Hue Modifier							
1	2343.51	2278.00	65.00	806.45	640.48	48.67	1918.46
2	2469.53	2419.00	50.00	775.67	606.49	50.82	1917.92
3	2281.36	2234.00	47.00	786.92	605.11	61.34	1920.87
Rotate Channel							
1	2447.58	2403.00	44.00	697.25	569.84	68.26	61.59
2	2251.76	2208.00	43.00	693.73	566.05	68.36	61.60
3	2480.57	2440.00	40.00	695.89	567.43	64.76	61.54

ตาราง 4.17 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	2231.29	2164.00	67.00	707.14	600.21	48.72	81.29
2	2419.35	2380.00	39.00	662.79	556.55	47.44	82.00
3	2253.74	2210.00	43.00	662.21	556.83	47.86	81.78
Color Reduction							
1	2510.14	2388.00	122.00	n/a	n/a	n/a	n/a
2	2370.14	2282.00	88.00	n/a	n/a	n/a	n/a
3	2480.50	2394.00	86.00	n/a	n/a	n/a	n/a
Contrast Strecth							
1	2266.08	2207.00	59.00	n/a	n/a	n/a	72.36
2	2185.39	2143.00	42.00	n/a	n/a	n/a	65.10
3	2285.97	2241.00	44.00	n/a	n/a	n/a	68.48
Gamma Correction							
1	2668.23	2624.00	44.00	1661.96	754.63	247.98	91.42
2	2675.39	2634.00	31.00	1661.55	742.58	245.84	90.59
3	2511.92	2473.00	38.00	1973.18	744.87	244.85	94.03
Brightness Correction							
1	2690.57	2655.00	35.00	690.70	581.50	52.38	114.19
2	2678.67	2647.00	31.00	673.40	566.91	48.94	105.30
3	2719.23	2680.00	39.00	670.85	561.73	51.19	105.24
Contrast Correction							
1	2431.65	2384.00	47.00	730.17	604.18	68.02	106.58
2	2431.65	2082.00	37.00	779.01	613.56	77.24	104.07
3	2154.51	2118.00	36.00	733.06	602.51	72.60	103.10
Saturation Correction							
1	2191.82	2116.00	75.00	922.70	757.49	53.07	2257.17
2	2108.83	2055.00	53.00	827.88	616.85	52.03	2264.91
3	2479.07	2421.00	58.00	777.69	612.55	51.14	2260.45

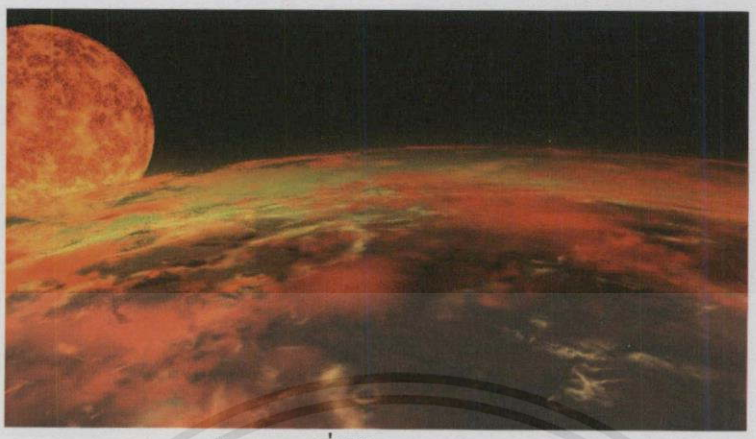
ตาราง 4.17 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	2352.07	2267.00	85.00	n/a	n/a	n/a	104.26
2	2454.78	2410.00	44.00	n/a	n/a	n/a	104.63
3	2280.46	2239.00	44.00	n/a	n/a	n/a	105.79
YCbCr Level Linear							
1	2431.99	2361.00	70.00	n/a	n/a	n/a	1602.93
2	2463.87	2418.00	45.00	n/a	n/a	n/a	1606.54
3	2395.39	2351.00	44.00	n/a	n/a	n/a	1610.52
HSL Level Linear							
1	2463.99	2403.00	60.00	n/a	n/a	n/a	2252.33
2	2492.13	2429.00	63.00	n/a	n/a	n/a	2252.96
3	2450.08	2386.00	64.00	n/a	n/a	n/a	2260.32
Blur							
1	2703.50	2526.00	177.00	1913.86	752.70	287.69	6710.00
2	2770.25	2637.00	133.00	1735.94	757.60	307.66	6712.89
3	2613.92	2478.00	135.00	1736.67	790.33	290.33	6712.51
Edge							
1	2768.01	2699.00	99.00	1938.26	756.84	255.37	3161.13
2	2740.08	2664.00	76.00	1702.72	767.35	266.93	3167.57
3	2645.94	2553.00	92.00	2174.04	761.01	266.04	3170.78
Sharpen							
1	2656.06	2532.00	124.00	1856.86	760.80	320.57	3042.84
2	2527.52	2425.00	102.00	1747.13	819.34	263.59	3054.05
3	2651.10	2555.00	96.00	1696.94	762.89	262.19	3043.33
Sobel							
1	2310.88	2244.00	66.00	1762.45	820.20	269.65	n/a
2	2410.92	2356.00	54.00	1694.85	754.42	269.42	n/a
3	2378.65	2326.00	52.00	1729.28	765.22	293.65	n/a

ตาราง 4.17 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	2693.95	2603.00	118.00	1747.52	754.26	289.88	6710.09
2	2563.28	2375.00	203.00	1908.85	740.81	494.10	6706.97
3	2898.52	2683.00	103.00	1976.56	746.87	564.92	6872.42
Gaussian Blur							
1	2614.4	2418.00	196.00	694.09	566.18	64.30	3027.23
2	2966.32	2809.00	157.00	965.74	562.07	73.85	3099.79
3	2726.00	2726.00	156.00	713.43	560.70	61.46	3027.67
Mean Filter							
1	2734.30	2606.00	128.00	1877.95	765.84	429.84	3068.32
2	2842.62	2753.00	89.00	1676.06	751.21	266.29	3069.24
3	2634.29	2745.00	89.00	1686.33	753.07	266.67	3050.88
Median Filter							
1	2564.35	2121.00	443.00	n/a	n/a	n/a	9913.79
2	2885.92	2453.00	432.00	n/a	n/a	n/a	9987.80
3	2960.43	2520.00	440.00	n/a	n/a	n/a	9877.06
Conservative Smoothing							
1	2440.44	2336.00	104.00	n/a	n/a	n/a	1895.92
2	2481.11	2387.00	94.00	n/a	n/a	n/a	1899.41
3	2439.27	2354.00	85.00	n/a	n/a	n/a	1898.94
Texturing							
1	2881.01	2838.00	16.00	n/a	n/a	n/a	380.77
2	2845.16	2831.00	15.00	n/a	n/a	n/a	377.68
3	2862.36	2845.00	14.00	n/a	n/a	n/a	375.47
Texture Merge							
1	3353.42	3299.00	25.00	n/a	n/a	n/a	2305.02
2	3323.49	3272.00	26.00	n/a	n/a	n/a	2312.34
3	3374.04	3327.00	26.00	n/a	n/a	n/a	2334.45
Texture Filter							
1	3291.11	3230.00	26.00	n/a	n/a	n/a	2348.90
2	3380.60	3325.00	25.00	n/a	n/a	n/a	2422.92
3	3390.67	3325.00	24.00	n/a	n/a	n/a	2349.96

4.2.4.3.3.2 รูปที่2



รูป4.18 รูปที่ใช้ในการทดลอง

ตาราง 4.18 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Grayscale							
1	2440.31	2395.00	45.00	509.28	462.02	33.41	47.43
2	2487.89	2441.00	46.00	505.58	464.36	29.98	49.08
3	2410.97	2366.00	44.00	504.08	462.92	29.64	48.12
Sepia							
1	2423.31	2368.00	55.00	n/a	n/a	n/a	179.18
2	2383.49	2340.00	43.00	n/a	n/a	n/a	182.36
3	2314.37	2273.00	41.00	n/a	n/a	n/a	181.10
Hue Modifier							
1	2347.49	2291.00	56.00	925.14	514.08	288.68	1865.82
2	2336.08	2283.00	53.00	675.01	506.74	50.28	1864.93
3	2381.33	2339.00	42.00	1474.41	505.81	49.25	1864.95
Rotate Channel							
1	2220.51	2178.00	42.00	662.94	532.89	68.24	61.04
2	2357.15	2322.00	35.00	596.66	470.32	64.92	61.78
3	2292.32	2255.00	37.00	598.62	473.00	67.03	60.26

ตาราง 4.18 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Invert							
1	2433.65	2381.00	52.00	573.31	465.39	50.18	78.88
2	2222.51	2183.00	39.00	575.34	464.50	53.08	80.32
3	2235.61	2195.00	40.00	568.45	461.80	48.82	78.21
Color Reduction							
1	2555.74	2451.00	104.00	n/a	n/a	n/a	n/a
2	2450.38	2367.00	83.00	n/a	n/a	n/a	n/a
3	2433.34	2348.00	85.00	n/a	n/a	n/a	n/a
Contrast Strecth							
1	2402.49	2336.00	66.00	n/a	n/a	n/a	76.67
2	2281.99	2241.00	40.00	n/a	n/a	n/a	68.48
3	2396.08	2357.00	39.00	n/a	n/a	n/a	69.36
Gamma Correction							
1	2615.42	2576.00	39.00	1764.98	662.81	435.34	92.32
2	2290.73	2248.00	42.00	1715.39	805.94	250.93	90.69
3	2550.15	2520.00	30.00	2229.12	668.85	256.65	90.93
Brightness Correction							
1	2559.79	2513.00	46.00	578.87	465.04	55.29	105.12
2	2608.31	2575.0	33.00	568.77	465.61	48.66	103.68
3	2575.07	2543.00	32.00	576.59	459.78	48.63	104.15
Contrast Correction							
1	2144.78	2043.00	101.00	635.13	503.21	72.43	107.03
2	1964.51	1931.00	33.00	627.47	501.04	68.15	102.93
3	1983.13	1950.00	33.00	630.79	502.93	69.10	104.68
Saturation Correction							
1	2103.09	2027.00	76.00	945.29	531.08	281.46	2091.05
2	2357.47	2301.00	56.00	671.03	509.33	52.55	2091.05
3	2275.25	2214.00	61.00	843.89	507.84	60.85	2093.26

ตาราง 4.18 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
RGB Level Linear							
1	2646.32	2564.00	82.00	n/a	n/a	n/a	105.16
2	2312.36	2269.00	43.00	n/a	n/a	n/a	105.06
3	2375.93	2332.00	43.00	n/a	n/a	n/a	104.22
YCbCr Level Linear							
1	2254.71	2192.00	62.00	n/a	n/a	n/a	1573.53
2	2339.78	2295.00	44.00	n/a	n/a	n/a	1591.97
3	2294.35	2250.00	44.00	n/a	n/a	n/a	1594.88
HSL Level Linear							
1	2499.57	2409.00	90.00	n/a	n/a	n/a	2121.39
2	2470.57	2409.00	61.00	n/a	n/a	n/a	2111.75
3	2433.59	2375.00	58.00	n/a	n/a	n/a	2152.67
Blur							
1	2736.04	2583.00	153.00	1729.51	759.51	290.68	6680.80
2	2625.97	2491.00	134.00	1813.63	846.62	285.29	6670.41
3	2719.64	2580.00	139.00	2039.28	674.04	283.63	6670.75
Edge							
1	2513.60	2405.00	108.00	1987.99	673.42	267.42	3172.45
2	2715.27	2619.00	96.00	1756.83	680.83	388.66	3179.75
3	2341.50	2250.00	91.00	1693.59	716.46	272.80	3181.47
Sharpen							
1	2785.11	2660.00	125.00	1660.28	674.95	276.46	3020.56
2	2642.70	2538.00	104.00	1603.95	657.93	260.18	3208.31
3	2703.61	2602.00	101.00	1609.13	642.85	255.22	3021.39
Sobel							
1	2457.15	2375.00	82.00	3041.14	1839.33	268.04	n/a
2	2261.83	2206.00	55.00	1698.08	654.23	269.42	n/a
3	2339.72	2286.00	53.00	1624.80	667.25	293.65	n/a

ตาราง 4.18 เปรียบเทียบเวลาที่ใช้ระหว่าง Aforge , Directcompute และ CUDA (ms) (ต่อ)

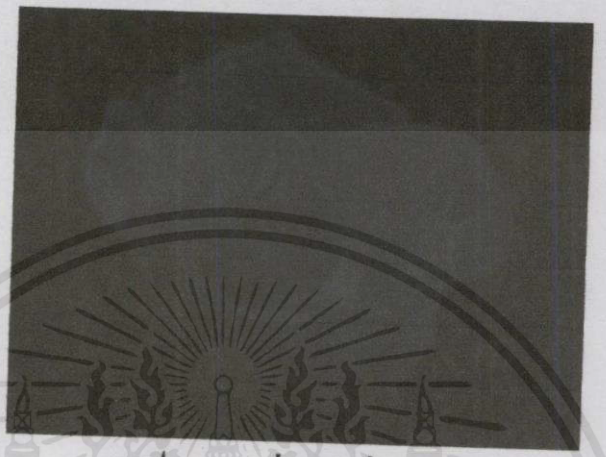
ครั้งที่	Direct compute	Load& Write time	Compute	Cuda	Load time	Compute	Aforge
Motion Blur							
1	2804.86	2696.00	108.00	1747.52	754.26	289.88	6710.09
2	2134.34	1936.00	198.00	1908.85	740.81	377.27	6706.97
3	2693.68	2584.00	109.00	1976.56	746.87	275.43	6872.42
Gaussian Blur							
1	2553.64	2404.00	149.00	613.75	494.43	61.28	3197.78
2	2773.75	2589.00	184.00	595.84	467.51	66.21	3088.85
3	3244.09	3080.00	164.00	589.48	467.51	69.03	3093.57
Mean Filter							
1	2644.12	2527.00	117.00	2123.32	662.37	270.52	3063.58
2	2620.00	2534.00	86.00	1658.23	719.42	273.05	3145.53
3	2630.23	2537.00	93.00	1623.21	673.06	266.23	3105.98
Median Filter							
1	2413.78	1946.00	467.00	n/a	n/a	n/a	9150.82
2	2978.68	2528.00	450.00	n/a	n/a	n/a	9185.75
3	2781.28	2336.00	445.00	n/a	n/a	n/a	9271.33
Conservative Smoothing							
1	2452.86	2355.00	97.00	n/a	n/a	n/a	1835.50
2	2383.03	2279.00	104.00	n/a	n/a	n/a	1793.62
3	2466.19	2379.00	87.00	n/a	n/a	n/a	1818.16
Texturing							
1	2881.01	2838.00	16.00	n/a	n/a	n/a	397.96
2	2845.16	2831.00	15.00	n/a	n/a	n/a	423.14
3	2862.36	2845.00	14.00	n/a	n/a	n/a	400.65
Texture Merge							
1	3728.92	3681.00	26.00	n/a	n/a	n/a	2312.64
2	2790.32	2732.00	25.00	n/a	n/a	n/a	2306.44
3	3353.42	3299.00	25.00	n/a	n/a	n/a	2307.61
Texture Filter							
1	3736.18	3689.00	24.00	n/a	n/a	n/a	2049.62
2	2655.35	2604.00	25.00	n/a	n/a	n/a	2351.54
3	2815.00	2765.00	24.00	n/a	n/a	n/a	2351.01

4.3 ความแตกต่างของผลลัพธ์

วิธีการในการเปรียบเทียบผลลัพธ์นั้นจะทำการนำผลลัพธ์ที่ได้มาทำการลบกัน ดังนี้

4.3.1 Directcompute กับ Aforge

4.3.1.1 Grayscale

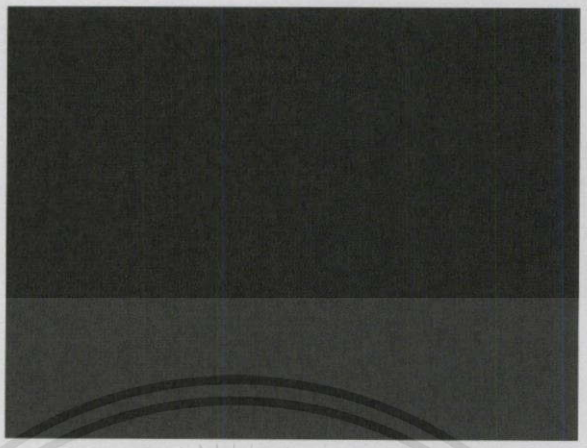


รูป 4.19 ผลต่างระหว่างภาพ

4.3.1.2 Sepia

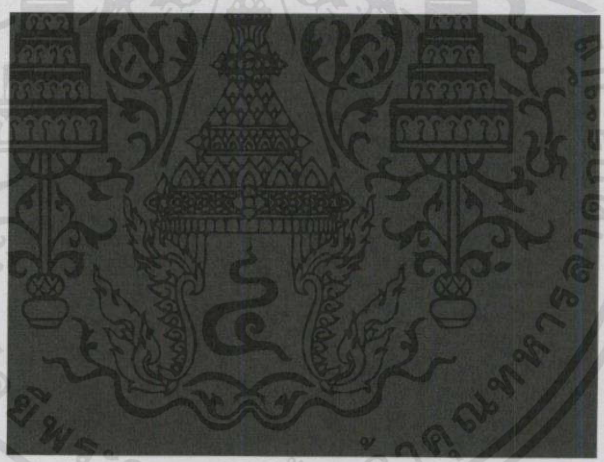
รูป 4.20 ผลต่างระหว่างภาพ

4.3.1.3 Rotate Channel



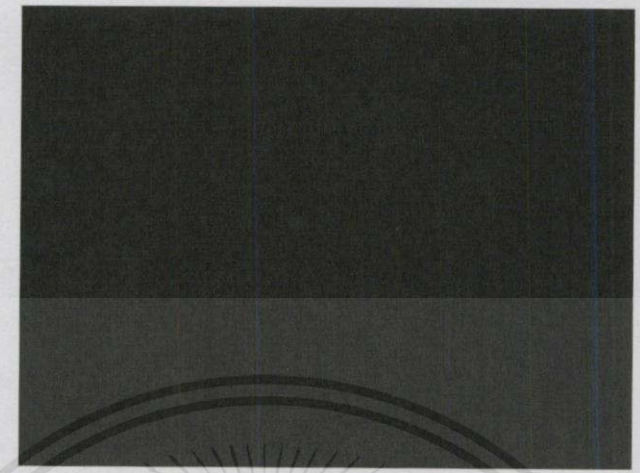
รูป 4.21 ผลต่างระหว่างภาพ

4.3.1.4 Invert



รูป 4.22 ผลต่างระหว่างภาพ

4.3.1.5 Brightness



รูป 4.23 ผลต่างระหว่างภาพ

4.3.1.6 Gamma

รูป 4.24 ผลต่างระหว่างภาพ

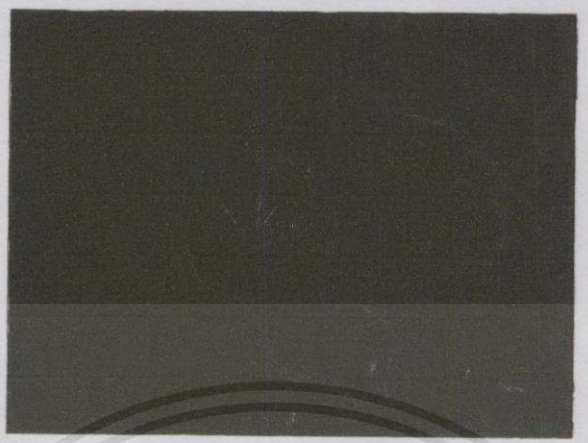
4.3.1.7 Saturation

รูป 4.25 ผลต่างระหว่างภาพ

4.3.1.8 Hue

รูป 4.26 ผลต่างระหว่างภาพ

4.3.1.9 Edge

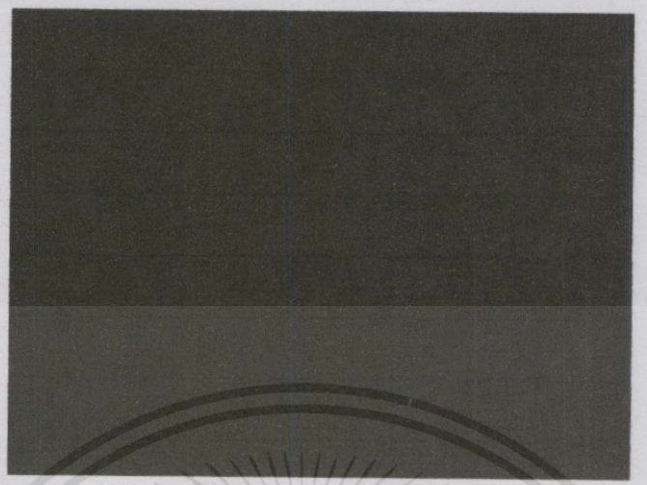


รูป 4.27 ผลต่างระหว่างภาพ

4.3.1.10 Blur

รูป 4.28 ผลต่างระหว่างภาพ

4.3.1.11 Gaussian Blur

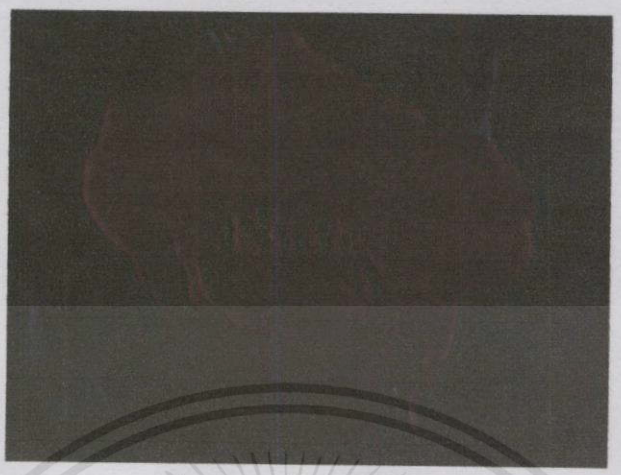


รูป 4.29 ผลต่างระหว่างภาพ

4.3.1.12 Sharpen

รูป 4.30 ผลต่างระหว่างภาพ

4.3.1.13 Motion Blur

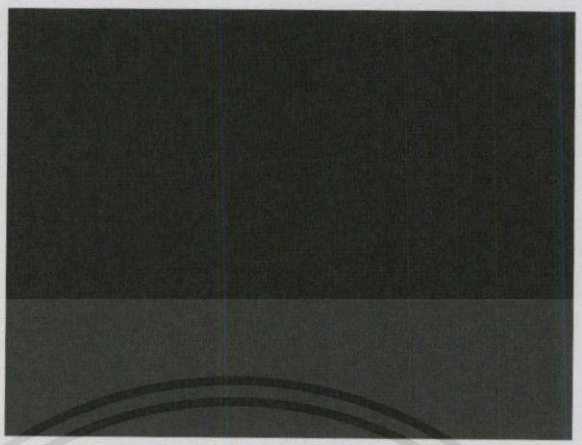


รูป 4.31 ผลต่างระหว่างภาพ

4.3.1.14 Mean

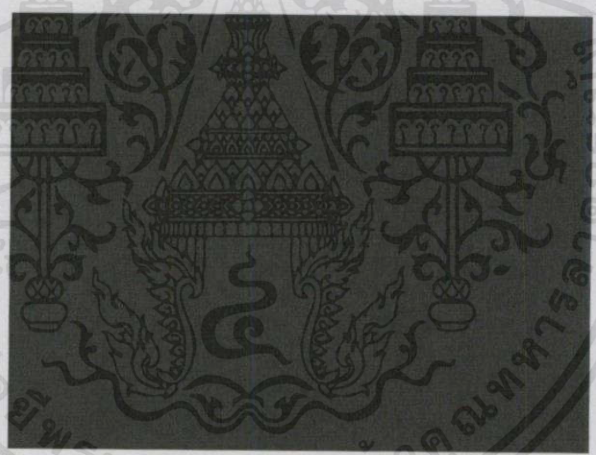
รูป 4.32 ผลต่างระหว่างภาพ

4.3.1.15 Median



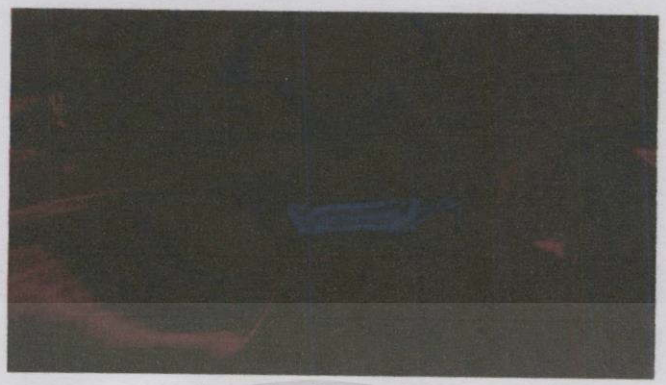
รูป 4.33 ผลต่างระหว่างภาพ

4.3.1.16 Conservative Smoothing



รูป 4.34 ผลต่างระหว่างภาพ

4.3.1.17 Color Reduction



รูป 4.35 ผลต่างระหว่างภาพ

4.3.1.18 Texturing

รูป 4.36 ผลต่างระหว่างภาพ

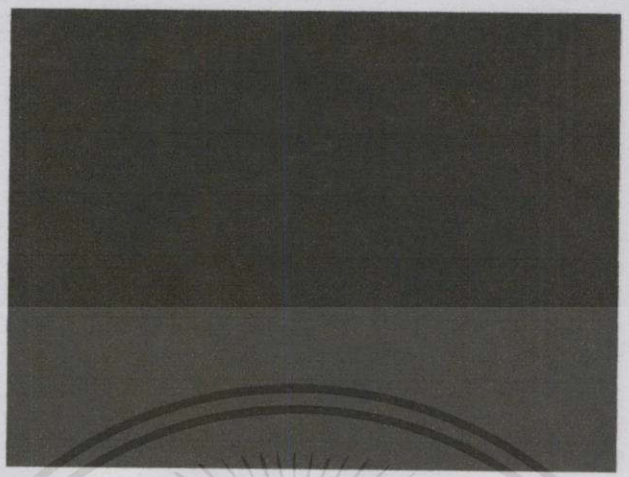
4.3.1.19 Texture Merging

รูป 4.37 ผลต่างระหว่างภาพ

4.3.1.20 Texture Filtering

รูป 4.38 ผลต่างระหว่างภาพ

4.3.1.21 RGB Level

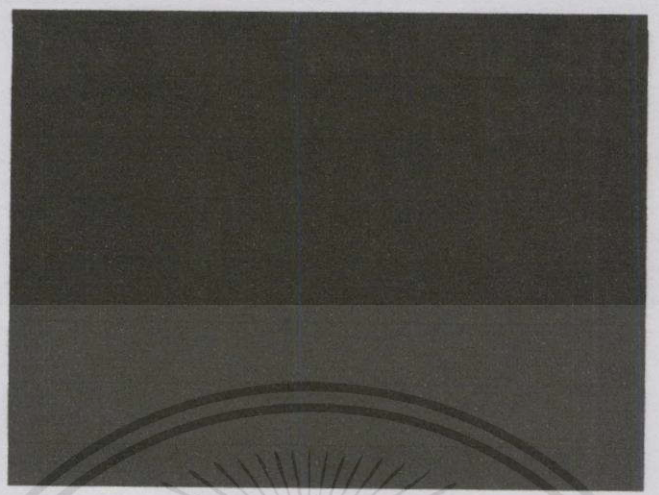


รูป 4.39 ผลต่างระหว่างภาพ

4.3.1.22 YCbCr Level

รูป 4.40 ผลต่างระหว่างภาพ

4.3.1.23 HSL Level

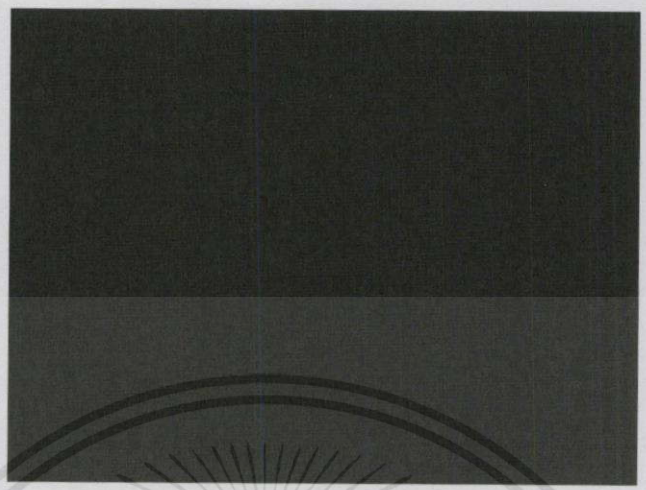


รูป 4.41 ผลต่างระหว่างภาพ

4.3.1.24 Contrast

รูป 4.42 ผลต่างระหว่างภาพ

4.3.1.25 Contrast Strecth

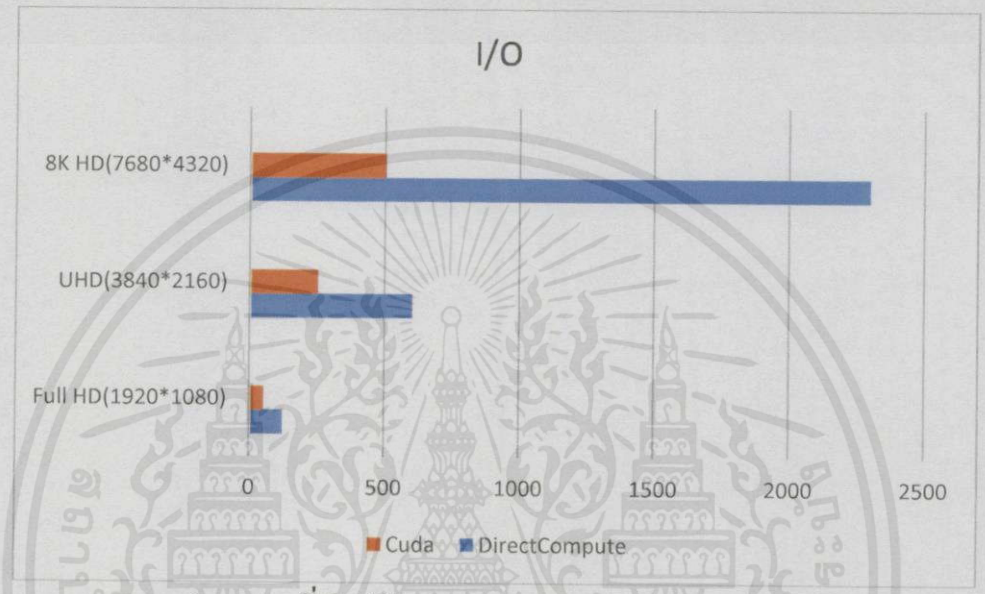


รูป 4.43 ผลต่างระหว่างภาพ

บทที่ 5

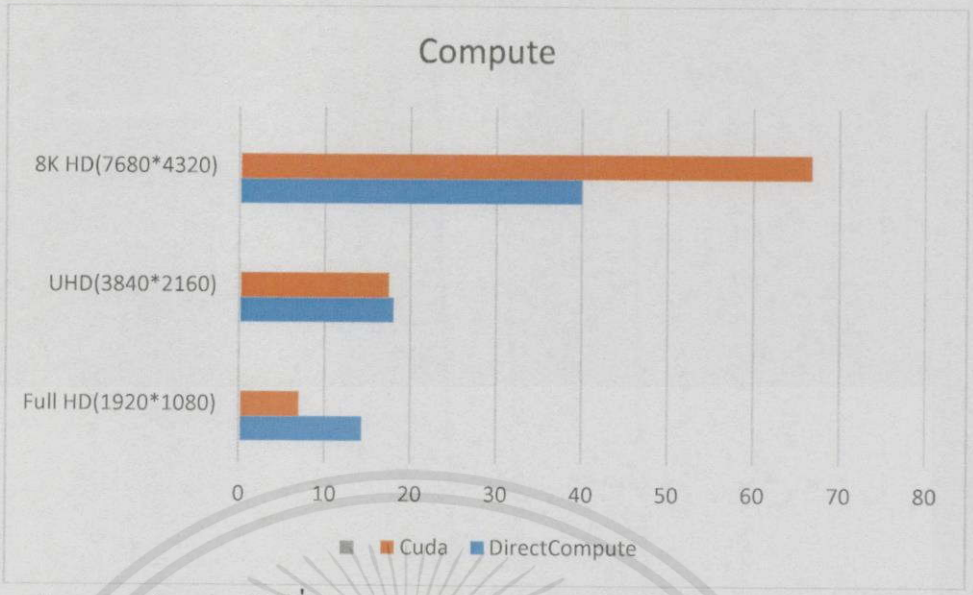
สรุปผลการทดลอง

5.1 ประสิทธิภาพของชิ้นงาน



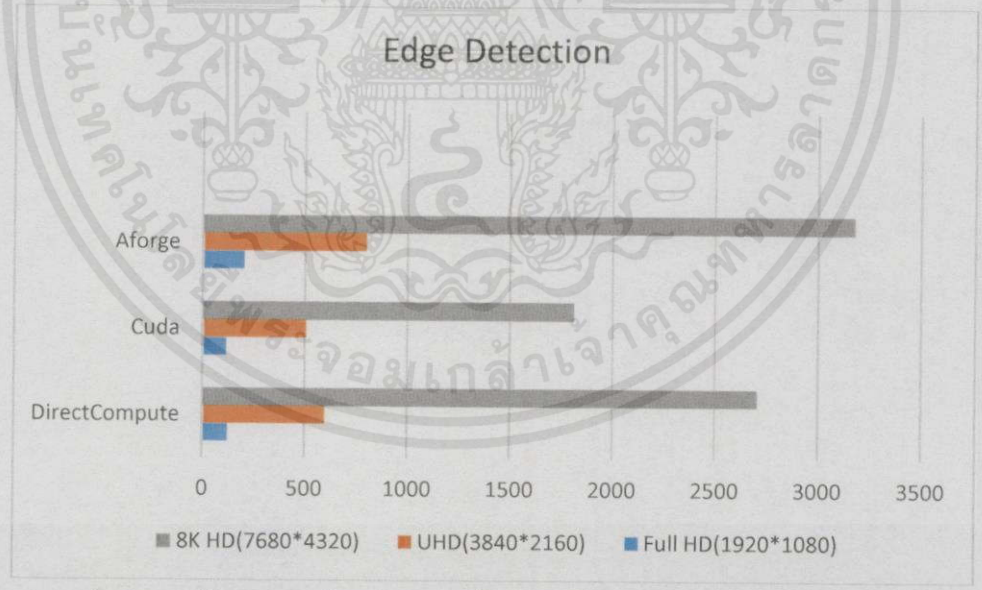
รูป 5.1 เวลาที่เรียกใช้ I/O ของ Cuda และ DirectCompute

จากการทดลองจะเห็นได้ว่า DirectCompute ใช้เวลาในการ I/O มากกว่าทำผลลัพธ์ที่ได้ออกมาไม่ดีเท่าที่ควรจะเป็น แต่ถ้าดูที่การประมวลผลอย่างเดียวบางฟังก์ชั่น DirectCompute ก็สามารถที่จะทำการประมวลผลได้เร็วกว่าที่ความละเอียดสูงๆตามรูป 5.2



รูป 5.2 การประมวลฟังก์ชั่น Rotate Chanenel ระหว่าง Cuda และ DirectCompute

แต่เมื่อดูที่การทำงานโดยรวมจะเห็นว่าการประมวลของDirectComputeก็สามารถทำได้ใกล้เคียงกับ Cuda และสามารถทำได้เร็วกว่า Aforge ในฟังก์ชั่นที่มีความซับซ้อนเยอะ อย่างเช่น Edge Detection ตามรูป 5.3



รูป 5.3 การประมวลผลโดยรวมระหว่าง Aforge Cuda และ DirectCompute

โดยในส่วนของจำนวนเทรดในการแบ่งภาพนั้นจะมีประสิทธิภาพมากขึ้นถ้าแบ่งให้เหมาะสมกับขนาดภาพซึ่งไม่จำเป็นว่าถ้าใช้จำนวนเทรดมากจะมีความเร็วในการประมวลผลมากกว่า ดังนั้นควรจะใช้เทรดให้เหมาะสมจึงจะได้ประสิทธิภาพที่ดี

5.2 ความแตกต่างของผลลัพธ์

ความแตกต่างของรูปที่ได้นั้นอาจจะมาจากการใช้อัลกอริทึมในการคำนวนต่างกันทำให้ได้ผลลัพธ์ไม่ตรงกัน และ อาจจะมาจากการปัดเศษในการคำนวนของ GPU และ CPU ต่างกันทำให้ภาพที่ได้มีความคาดเคลื่อนได้

ส่วนที่บางฟังก์ชั่นนั้นทำงานได้ช้ากว่า CPU เช่น Rotate Channel นั่นมาจากมีวิธีการคำนวนที่ง่าย ทำให้ GPU ไม่สามารถประมวลตามได้ทันเนื่องจากต้องมีการ Initial Device ต่างๆก่อน และไม่เสียในการ ถ่ายโอนข้อมูลไปมาด้วย

5.3 แนวทางการพัฒนาต่อ

จากผลการทดลองจะเห็นตัวแนวทางที่กลุ่มข้าพเจ้าได้พัฒนาขึ้นมานั้นจะมี overhead จากการโหลดภาพและเขียนภาพทำให้ประสิทธิภาพออกมาไม่ดีเท่าที่ควร

กลุ่มที่จะนำไปทำต่อควรจะศึกษาหาวิธีที่จะลด overhead จากการโหลดภาพ และ เขียนภาพรวมถึงการแบ่งเทรดให้เหมาะสม เพื่อที่จะได้มีประสิทธิภาที่ดีขึ้น

บรรณานุกรม

Doron Feinstein June. 2013. HLSL Development Cookbook. United Kingdom: PACKT Publishing

Microsoft Corporation. 2010. DirectCompute Lecture Series.[Online]. Available: https://channel9.msdn.com/Tags/directcompute-lecture-series

Microsoft Corporation. 2015. Reference for HLSL.[Online]. Available: https://msdn.microsoft.com/th-th/enus/library/windows/desktop/bb509638(v=vs.85).aspx

Microsoft Corporation. 2015. Programming Guide for HLSL.[Online]. Available: https://msdn.microsoft.com/en-us/library/windows/desktop/bb509635(v=vs.85).aspx

Nvidia Corporation. 2015. HLSL Developer forum.[Online]. Available: https://developer.nvidia.com/directcompute

Alexandre Mutel. 2012. SharpDX Documentation.[Online]. Available: http://sharpdx.org/documentation

EmguCV. 2008. Tutorial & Documentation.[Online]. Available: http://www.emgu.com/wiki/index.php/Tutorial

ภาคผนวก

โปรแกรม HLSL ในส่วนการประมวลผลภาพ

โปรแกรม 1.1 Gray scale

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
  float4 sample = input[dispatchThreadId.xy];
  float3 target = (sample.r+sample.g+sample.b)/3;
  output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรม 1.2 Sepia

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float3 target;
     target.r =
(sample.r*.393)+(sample.g*.769)+(sample.b*.189) ;
     target.g = (sample.r*.349) + (sample.g*.686) +
(sample.b*.168);
     target.b = (sample.r*.272) + (sample.g*.534) +
(sample.b*.131);
     output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรม 1.3 Hue-Modifier

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float3 hsl = 0;
     float u, v;
     u = -min(sample.r, min(sample.g, sample.b));
     v = max(sample.r, max(sample.g, sample.b));
     hsl.z = (v - u) * 0.5;
     float C = v + u;
     if (C != 0)
     {
          hsl.y = C / (1 - abs(2 * hsl.z - 1));
          float3 Delta = (hsl.z - sample.rgb) / C;
               Delta.rgb -= Delta.brg;
          Delta.rg += float2(2, 4);
          if (sample.r >= hsl.z)
               hsl.x = Delta.b;
          else if (sample.g >= hsl.z)
               hsl.x = Delta.r;
          else
               hsl.x = Delta.g;
          hsl.x = frac(hsl.x / 6);
     }
     h;
     float hh = (float)h / 360;
     hsl.x = hh;
     float ic = (1 - abs(2 * hsl.z - 1)) * hsl.y;
     float3 rgb, dev;
     rgb.r = abs(hsl.x * 6 - 3) - 1;
     rgb.g = 2 - abs(hsl.x * 6 - 2);
     rgb.b = 2 - abs(hsl.x * 6 - 4);
     dev = saturate(rgb);
     float3 target;
     target = (dev - 0.5)*ic + hsl.z;
     output[dispatchThreadId.xy] = float4(target, sample.a);
}
```

โปรแกรม 1.4 Rotate Chanel

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float3 target;
     target.r = sample.g;
     target.g = sample.b;
     target.b = sample.r;
     output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรมที่ 1.5 Invert Color

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float3 target;
     target.r = (sample.a-sample.r);
     target.g = (sample.a-sample.g);
     target.b = (sample.a-sample.b);
     output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรมที่ 1.6 Color Reduction

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
  float4 sample = input[dispatchThreadId.xy];
  float3 target;
  float temp;

//blue
  if (sample.b > 0 && sample.b <= 0.0625)
  {
       target.b = 0.3125;
  }
 else if (sample.b > 0.0625 && sample.b <= 0.125)
  {
       target.b = 0.09375;
  }
  else if (sample.b > 0.125 && sample.b <= 0.1875)
  {
       target.b = 0.15625;
  }
  else if (sample.b > 0.1875 && sample.b <= 0.25)
  {
       target.b = 0.21875;
  }
  else if (sample.b > 0.25 && sample.b <= 0.3125)
  {
       target.b = 0.28125;
  }
  else if (sample.b > 0.3125 && sample.b <= 0.375)
  {
       target.b = 0.34375;
  }
  else if (sample.b > 0.375 && sample.b <= 0.4375)
  {
       target.b = 0.40625;
  }
  else if (sample.b > 0.4375 && sample.b <= 0.5)
  {
       target.b = 0.46875;
  }
```

โปรแกรมที่ 1.6 Color Reduction (ต่อ)

```
}
  else if (sample.b > 0.625 && sample.b <= 0.6825)
  {
       target.b = 0.65625;
  }
  else if (sample.b > 0.6825 && sample.b <= 0.75)
  {
       target.b = 0.71875;
  }
  else if (sample.b > 0.75 && sample.b <= 0.8125)
  {
       target.b = 0.78125;
  }
  else if (sample.b > 0.8125 && sample.b <= 0.875)
  {
       target.b= 0.84375;
  }
  else if (sample.b > 0.875 && sample.b <= 0.9375)
  {
       target.b = 0.90625;
  }
  else if (sample.b > 0.9375)
  {
       target.b = 0.96875;
  }
  temp = (target.b - sample.b);
  target.b = target.b + temp;

  //red
  if (sample.r > 0 && sample.r <= 0.0625)
  {
       target.r = 0.03125;
  }
  else if (sample.r > 0.0625 && sample.r <= 0.125)
  {
       target.r = 0.09375;
  }
  else if (sample.r > 0.1875 && sample.r <= 0.25)
  {
       target.b = 0.21875;
  }
  else if (sample.r > 0.3125 && sample.r <= 0.375)
  {
       target.r = 0.34375;
  }
```

โปรแกรมที่ 1.6 Color Reduction (ต่อ)

```
else if (sample.r > 0.375 && sample.r <= 0.4375)
  {
       target.r = 0.40625;
  }
  else if (sample.r > 0.4375 && sample.r <= 0.5)
  {
       target.r = 0.46875;
  }
  else if (sample.r > 0.5 && sample.r <= 0.5625)
  {
       target.r = 0.53125;
  }
  else if (sample.r > 0.5625 && sample.r <= 0.625)
  {
       target.r = 0.59375;
  }
  else if (sample.r > 0.625 && sample.r <= 0.6825)
  {
       target.r = 0.65625;
  }
  else if (sample.r > 0.6825 && sample.r <= 0.75)
  {
       target.r = 0.71875;
  }
  else if (sample.r > 0.75 && sample.r <= 0.8125)
  {
       target.r = 0.78125;
  }
  else if (sample.r > 0.8125 && sample.r <= 0.875)
  {
       target.r = 0.84375;
  }
  else if (sample.r > 0.875 && sample.r <= 0.9375)
  {
       target.r = 0.90625;
  }
  else if (sample.r > 0.9375)
  {
       target.r = 0.96875;
  }
  temp = (target.r - sample.r);
  target.r = target.r + temp;
```

โปรแกรมที่ 1.6 Color Reduction (ต่อ)

```
//green
  if (sample.g > 0 && sample.g <= 0.0625)
  {
       target.g = 0.03125;
  }
  else if (sample.g > 0.0625 && sample.g <= 0.125)
  {
       target.g = 0.09375;
  }
  else if (sample.g > 0.125 && sample.g <= 0.1875)
  {
       target.g = 0.15625;
  }
  else if (sample.g > 0.1875 && sample.g <= 0.25)
  {
       target.g = 0.21875;
  }
  else if (sample.g > 0.25 && sample.g <= 0.3125)
  {
       target.g = 0.28125;
  }
  else if (sample.g > 0.3125 && sample.g <= 0.375)
  {
       target.g = 0.34375;
  }
  else if (sample.g > 0.375 && sample.g <= 0.4375)
  {
       target.g = 0.40625;
  }
  else if (sample.g > 0.4375 && sample.g <= 0.5)
  {
       target.g = 0.46875;
  }
  else if (sample.g > 0.5 && sample.g <= 0.5625)
  {
       target.g = 0.53125;
     }
  else if (sample.g > 0.5625 && sample.g <= 0.625)
  {
       target.g = 0.59375;
  }
```

โปรแกรมที่ 1.6 Color Reduction (ต่อ)

```
    else if (sample.g > 0.625 && sample.g <= 0.6825)
  {
      target.g = 0.65625;
  }
  else if (sample.g > 0.6825 && sample.g <= 0.75)
  {
      target.g = 0.71875;
  }
  else if (sample.g > 0.75 && sample.g <= 0.8125)
  {
      target.g = 0.78125;
  }
  else if (sample.g > 0.8125 && sample.g <= 0.875)
  {
      target.g = 0.84375;
  }
  else if (sample.g > 0.875 && sample.g <= 0.9375)
  {
      target.g = 0.90625;
  }
  else if (sample.g > 0.9375)
  {
      target.g = 0.96875;
  }
  temp = (target.g - sample.g);
  target.g = target.g + temp;

  output[dispatchThreadId.xy] = float4(target.rgb, sample.a);
}
```

โปรแกรมที่ 1.7 Brightness

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
  brg;
  float f = (float)brg;
  float f1 = f / 255;
  float4 sample = input[dispatchThreadId.xy];
  float3 target = (float3)(sample.rgb +f1);
  output[dispatchThreadId.xy] = float4(target, sample.a);
}
```

โปรแกรมที่ 1.8 Contrast

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
  co;
  float cc = ((float)co/127);
  float c = cc * 10;
  float4 sample = input[dispatchThreadId.xy];
  float3 target;
  target.r = ((sample.r - 0.5)*(c)+0.5);
  target.g = ((sample.g - 0.5)*(c)+0.5);
  target.b = ((sample.b - 0.5)*(c)+0.5);
  output[dispatchThreadId.xy] = float4(target, sample.a);
}
```

โปรแกรมที่ 1.9 RGB Level

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float kr = 0, br = 0, kg = 0, bg = 0, kb = 0, bb = 0;
     float3 target;
     if (mir != mar)
     {
          kr = (omar - omir) / (mar - mir);
          br = omir - kr*mir;
     }
     if (sample.r >= mar)
          target.r = omar;
     else if (sample.r <= mir)
          target.r = omir;
     else
          target.r = (kr*sample.r + br);
     if (mig != mag)
     {
          kg = (omag - omig) / (mag - mig);
          bg = omig - kg*mig;
     }
     if (sample.g >= mag)
          target.g = omag;
     else if (sample.g <= mig)
          target.g = omig;
     else
          target.g = (kg*sample.g + bg);
     if (mib != mab)
     {
          kb = (omab - omib) / (mab - mib);
          bb = omib - kb*mib;
     }
```

โปรแกรมที่ 1.9 RGB Level (ต่อ)

```
      if (sample.b >= mab)
            target.b = omab;
      else if (sample.b <= mib)
            target.b = omib;
      else
            target.b = (kb*sample.b + bb);
      output[dispatchThreadId.xy] =
float4(target.rgb,sample.a);
}
```

โปรแกรมที่ 1.10 YCbCr

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define THREADSX 32
#define THREADSY 32
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float ky = 0, by = 0;
     float kcb = 0, bcb = 0;
     float kcr = 0, bcr = 0;
     float y = (sample.r*0.2989 + sample.g*0.5865 +
sample.b*0.1145);
     float cb = (sample.r*(-.01687 )+ sample.g*(-0.3313) +
sample.b*0.500);
     float cr = (sample.r*(0.500) + sample.g*(-0.4184) +
sample.b*(-0.0816));
     float miy = 0, may = 1;
     float micb = (-0.276), macb = 0.163;
     float micr = (-0.202), macr = 0.5;
```

โปรแกรมที่ 1.10 YCbCr (ต่อ)

```
if (may != miy)
     {
          ky = (1 - 0) / (may - miy);
          by = (0) - ky*miy;
     }
     if (y >= may)
          y = 1;
     else if (y <= miy)
          y = 0;
     else
          y = ky*y + by;
     if (macb != micb)
     {
          kcb = (0.5 - (-0.5)) / (macb - micb);
          bcb = (-0.5) - kcb*micb;
     }
     if (cb >= macb)
          cb = 0.5;
     else if (cb <= micb)
          cb = -0.5;
     else
          cb = kcb*cb + bcb;
     if (macr != micr)
     {
          kcr = (0.5 - (-0.5)) / (macr - micr);
          bcr = (-0.5) - kcr*micr;
     }
     if (cr >= macr)
          cr = 0.5;
     else if (cr <= micr)
          cr = -0.5;
     else
          cr = kcr*cr + bcr;
     float3 target;
     target.r = y+0*cb+1.4022*cr;
     target.g = y - (0.3456*cb) - (cr*0.7145);
     target.b = y * 1 + cb*1.7710 + cr * 0;
     output[dispatchThreadId.xy] =
float4(target.rgb,sample.a);
}
```

โปรแกรมที่ 1.11 HSL Level

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float kl = 0, bl = 0;
     float ks = 0, bs = 0;
     float3 hsl = 0;
     float u, v;
     u = -min(sample.r, min(sample.g, sample.b));
     v = max(sample.r, max(sample.g, sample.b));
     hsl.z = (v - u) * 0.5;
     float C = v + u;
     if (C != 0)
     {
          hsl.y = C / (1 - abs(2 * hsl.z - 1));
          float3 Delta = (hsl.z - sample.rgb) / C;
               Delta.rgb -= Delta.brg;
          Delta.rg += float2(2, 4);
          if (sample.r >= hsl.z)
               hsl.x = Delta.b;
          else if (sample.g >= hsl.z)
               hsl.x = Delta.r;
          else
               hsl.x = Delta.g;
          hsl.x = frac(hsl.x / 6);
     } if (mal != mil)
     {
          kl = (omal - omil) / (mal - mil);
          bl = omil - kl*mil;
     }
     if (mas != mis)
     {
          ks = (omas - omis) / (mas - mis);
          bs = omis - ks * mis;
     }
```

โปรแกรมที่ 1.11 HSL Level (ต่อ)

```
if (hsl.z >= mal)
        hsl.z = omal;
    else if (hsl.z <= mil)
        hsl.z = omil;
    else
        hsl.z = kl * hsl.z + bl;
    if (hsl.y >= mas)
        hsl.y = omas;
    else if (hsl.y <= mis)
        hsl.y = omis;
    else
        hsl.y = ks * hsl.y + bs;
    float ic = (1 - abs(2 * hsl.z - 1)) * hsl.y;
    float3 rgb, dev;
    rgb.r = abs(hsl.x * 6 - 3) - 1;
    rgb.g = 2 - abs(hsl.x * 6 - 2);
    rgb.b = 2 - abs(hsl.x * 6 - 4);
    dev = saturate(rgb);
    float3 target;
    target = (dev - 0.5)*ic + hsl.z;
    output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรมที่ 1.12 Saturation

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float3 hsl = 0;
     float u, v;
     u = -min(sample.r, min(sample.g, sample.b));
     v = max(sample.r, max(sample.g, sample.b));
     hsl.z = (v - u) * 0.5;
     float C = v + u;
     if (C != 0)
     {
          hsl.y = C / ( 1-abs(2* hsl.z -1));
          hsl.y = hsl.y - 0.5;
          float3 Delta = (hsl.z - sample.rgb) / C;
               Delta.rgb -= Delta.brg;
          Delta.rg += float2(2, 4);
          if (sample.r >= hsl.z)
               hsl.x = Delta.b;
          else if (sample.g >= hsl.z)
               hsl.x = Delta.r;
          else
               hsl.x = Delta.g;
          hsl.x = frac(hsl.x / 6);
     }
     s;
     float ss = (float)s;
     hsl.y = hsl.y+ss;
     float ic = (1-abs(2* hsl.z -1)) * hsl.y;
     float3 rgb,dev;
     rgb.r =  abs(hsl.x * 6 - 3) - 1;
     rgb.g = 2 - abs(hsl.x * 6 - 2);
     rgb.b = 2 - abs(hsl.x * 6 - 4);
     dev = saturate(rgb);
     float3 target;
     target = (dev - 0.5)*ic +hsl.z;
     output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรมที่ 1.13 Contrast Stretch

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
  float4 sample = input[dispatchThreadId.xy];
  float3 target;
  target.r = ((sample.r - 0.5) * 1.5) + 0.5;
  target.g = ((sample.g - 0.5) * 1.5) + 0.5;
  target.b = ((sample.b - 0.5) * 1.5) + 0.5;
  output[dispatchThreadId.xy] = float4(target, sample.a);
}
```

โปรแกรมที่ 1.15 Gamma

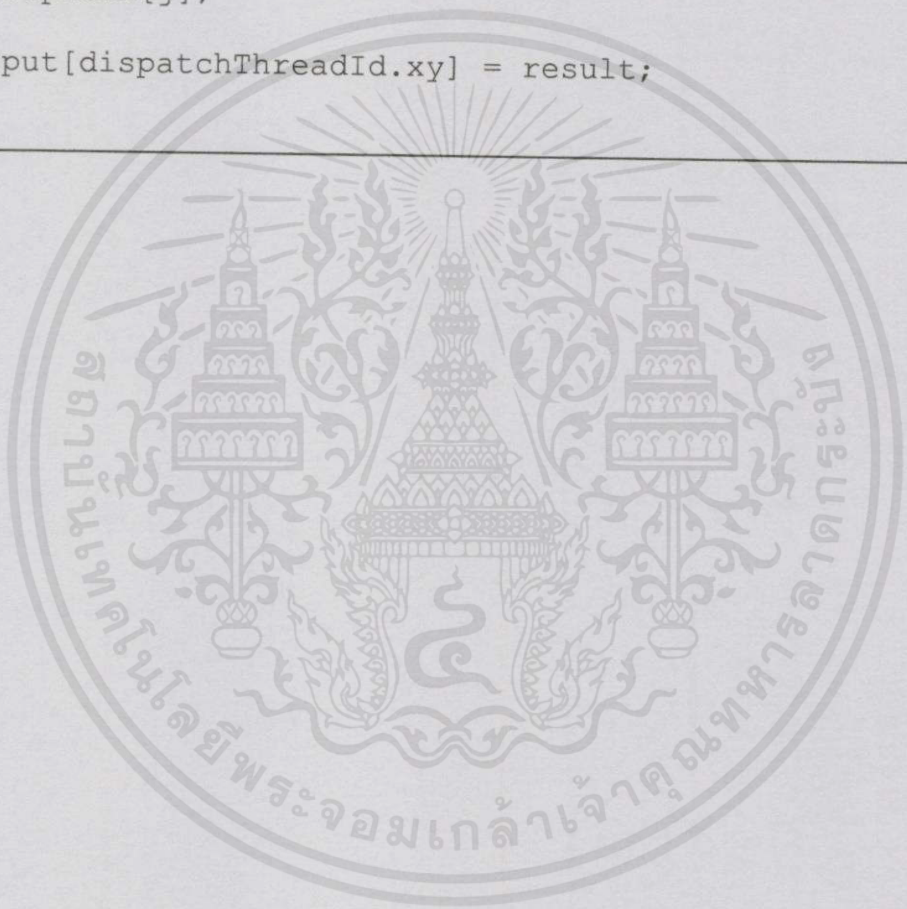
```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void DesaturateCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
    g;
    float gg = (float)g;
    float4 sample = input[dispatchThreadId.xy];
    float3 target = (float3)pow(sample.rgb, (1/gg));
    output[dispatchThreadId.xy] = float4(target,sample.a);
}
```

โปรแกรมที่ 1.16 Blur Horizontal

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 9
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemX[GROUPSIZE + (THREADSY
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 0, 0.2, 0, 0.2,
0.2, 0.2, 0, 0.2, 0 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterHorizontalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.y * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemX[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.x < FILTERRADIUS)
     {
          int x = dispatchThreadId.x - FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(x, 0), dispatchThreadId.y)];
     }
     if (groupThreadId.x >= THREADSX - FILTERRADIUS)
     {
          int x = dispatchThreadId.x + FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(x, input.Length.x - 1),dispatchThreadId.y)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemX[j];
```

โปรแกรมที่ 1.16 Blur Horizontal (ต่อ)

```
GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemX[j];
     }
     output[dispatchThreadId.xy] = result;
}
```



โปรแกรมที่ 1.17 Blur Vertical

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 9
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemY[GROUPSIZE + (THREADSX
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = {
0,0.2,0,0.2,0.2,0.2,0,0.2,0};
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterVerticalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.x * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemY[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.y < FILTERRADIUS)
     {
          int y = dispatchThreadId.y - FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(y, 0), dispatchThreadId.x)];
     }

     if (groupThreadId.y >= THREADSY - FILTERRADIUS)
     {
          int y = dispatchThreadId.y + FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(y, input.Length.y - 1), dispatchThreadId.x)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
               FilterGroupMemY[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.18 Edge Horizontal

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 3
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemX[GROUPSIZE + (THREADSY
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 1,-1.99999,1 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterHorizontalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.y * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemX[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.x < FILTERRADIUS)
     {
          int x = dispatchThreadId.x - FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(x, 0), dispatchThreadId.y)];
     }
     if (groupThreadId.x >= THREADSX - FILTERRADIUS)
     {
          int x = dispatchThreadId.x + FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(x, input.Length.x - 1), dispatchThreadId.y)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
               FilterGroupMemX[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.19 Edge Vertical

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 3

#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemY[GROUPSIZE + (THREADSX
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 0,-1.99999,0 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterVerticalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.x * 2
*
          FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemY[offsetGroupIndex] =
          input[min(dispatchThreadId.xy, input.Length.xy -
1)];
     if (groupThreadId.y < FILTERRADIUS)
     {
          int y = dispatchThreadId.y - FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex - FILTERRADIUS] =
               input[int2(max(y, 0), dispatchThreadId.x)];
     }
     if (groupThreadId.y >= THREADSY - FILTERRADIUS)
     {
          int y = dispatchThreadId.y + FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex + FILTERRADIUS] =
               input[int2(min(y, input.Length.y - 1),
               dispatchThreadId.x)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
```

โปรแกรมที่ 1.19 Edge Vertical (ต่อ)

```
for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
               FilterGroupMemY[j];
     }

     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.20 Sharpen Horizontal

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 3
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemX[GROUPSIZE + (THREADSY
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { -2,5,-2};
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterHorizontalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.y * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemX[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.x < FILTERRADIUS)
     {
          int x = dispatchThreadId.x - FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(x, 0), dispatchThreadId.y)];
     }
     if (groupThreadId.x >= THREADSX - FILTERRADIUS)
     {
          int x = dispatchThreadId.x + FILTERRADIUS;
```

โปรแกรมที่ 1.20 Sharpen Horizontal (ต่อ)

```
{
          int x = dispatchThreadId.x + FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(x, input.Length.x - 1), dispatchThreadId.y)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemX[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.21 Sharpen Vertical

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 3
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemY[GROUPSIZE + (THREADSX
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { -1, 5, -1 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterVerticalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.x * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemY[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.y < FILTERRADIUS)
     {
          int y = dispatchThreadId.y - FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(y, 0), dispatchThreadId.x)];
     }
     if (groupThreadId.y >= THREADSY - FILTERRADIUS)
     {
          int y = dispatchThreadId.y + FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(y, input.Length.y - 1), dispatchThreadId.x)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemY[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.22 Sobel

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void SobelCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_DispatchThreadID)
{
     float3 v0 = input[dispatchThreadId.xy + float2(-1.0, -
1.0)].rgb;
     float3 v1 = input[dispatchThreadId.xy + float2(0.0, -
1.0)].rgb;
     float3 v2 = input[dispatchThreadId.xy + float2(+1.0, -
1.0)].rgb;
     float3 v3 = input[dispatchThreadId.xy + float2(-1.0,
0.0)].rgb;
     float3 v4 = input[dispatchThreadId.xy + float2(0.0,
0.0)].rgb;
     float3 v5 = input[dispatchThreadId.xy + float2(+1.0,
0.0)].rgb;
     float3 v6 = input[dispatchThreadId.xy + float2(-1.0,
+1.0)].rgb;
     float3 v7 = input[dispatchThreadId.xy + float2(0.0,
+1.0)].rgb;
     float3 v8 = input[dispatchThreadId.xy + float2(+1.0,
+1.0)].rgb;
     float3 gx = (v0 + (2 * v1) + v2) - (v6 + (2 * v7) + v8);
     float3 gy = (v2 + (2 * v4) + v8) - (v0 + (2 * v3) + v5);
     float3 target = sqrt(gx*gx + gy*gy);
     output[dispatchThreadId.xy] = float4(target, 1);
}
```

โปรแกรมที่ 1.23 Gaussian Horizontal

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 9
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemX[GROUPSIZE + (THREADSY
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 0.08167442,
0.1016454, 0.1188356, 0.1305153, 0.1346584, 0.1305153,
0.1188356, 0.1016454, 0.08167442 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterHorizontalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.y * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemX[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.x < FILTERRADIUS)
     {
          int x = dispatchThreadId.x - FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(x, 0), dispatchThreadId.y)];
     }
     if (groupThreadId.x >= THREADSX - FILTERRADIUS)
     {
          int x = dispatchThreadId.x + FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(x, input.Length.x - 1), dispatchThreadId.y)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
          int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemX[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.24 Gaussian Vertical

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 9
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemY[GROUPSIZE + (THREADSX
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 0.08167442,
0.1016454, 0.1188356, 0.1305153, 0.1346584, 0.1305153,
0.1188356, 0.1016454, 0.08167442 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterVerticalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.x * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemY[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.y < FILTERRADIUS)
     {
          int y = dispatchThreadId.y - FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(y, 0), dispatchThreadId.x)];
     }
     if (groupThreadId.y >= THREADSY - FILTERRADIUS)
     {
          int y = dispatchThreadId.y + FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(y, input.Length.y - 1), dispatchThreadId.x)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
     int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemY[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.25 Mean Horizontal

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 3
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemX[GROUPSIZE + (THREADSY
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 0.4, 0.2, 0.4 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterHorizontalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{

     uint offsetGroupIndex = groupIndex + (groupThreadId.y * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemX[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.x < FILTERRADIUS)
     {
          int x = dispatchThreadId.x - FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(x, 0), dispatchThreadId.y)];
     }
     if (groupThreadId.x >= THREADSX - FILTERRADIUS)
     {
          int x = dispatchThreadId.x + FILTERRADIUS;
          FilterGroupMemX[offsetGroupIndex + FILTERRADIUS]
=input[int2(min(x, input.Length.x - 1), dispatchThreadId.y)];
     }
     GroupMemoryBarrierWithGroupSync();
     float4 result = float4(0, 0, 0, 0);
     int centerPixel = offsetGroupIndex;
     [unroll]
```

โปรแกรมที่ 1.25 Mean Horizontal (ต่อ)

```
for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemX[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.26 Mean Vertical

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
#define FILTERTAP 3
#define FILTERRADIUS ((FILTERTAP-1)/2)
#define GROUPSIZE (THREADSX * THREADSY)
groupshared float4 FilterGroupMemY[GROUPSIZE + (THREADSX
*FILTERRADIUS * 2)];
static const float BlurKernel[FILTERTAP] = { 0.4,0.2,0.4 };
[numthreads(THREADSX, THREADSY, 1)]
void BlurFilterVerticalCS(uint groupIndex: SV_GroupIndex,
  uint3 groupId : SV_GroupID,
  uint3 groupThreadId : SV_GroupThreadID,
  uint3 dispatchThreadId : SV_dispatchThreadID)
{
     uint offsetGroupIndex = groupIndex + (groupThreadId.x * 2
* FILTERRADIUS) + FILTERRADIUS;
     FilterGroupMemY[offsetGroupIndex] =
input[min(dispatchThreadId.xy, input.Length.xy - 1)];
     if (groupThreadId.y < FILTERRADIUS)
     {
          int y = dispatchThreadId.y - FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex - FILTERRADIUS] =
input[int2(max(y, 0), dispatchThreadId.x)];
     }
     if (groupThreadId.y >= THREADSY - FILTERRADIUS)
     {
          int y = dispatchThreadId.y + FILTERRADIUS;
          FilterGroupMemY[offsetGroupIndex + FILTERRADIUS] =
input[int2(min(y, input.Length.y - 1), dispatchThreadId.x)];
     }
     GroupMemoryBarrierWithGroupSync();
```

โปรแกรมที่ 1.26 Mean Vertical (ต่อ)

```
float4 result = float4(0, 0, 0, 0);
     int centerPixel = offsetGroupIndex;
     [unroll]
     for (int i = -FILTERRADIUS; i <= FILTERRADIUS; ++i)
     {
          int j = centerPixel + i;
          result += BlurKernel[i + FILTERRADIUS] *
FilterGroupMemY[j];
     }
     output[dispatchThreadId.xy] = result;
}
```

โปรแกรมที่ 1.27 Median

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void Median3x3TapSinglePassCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_DispatchThreadID)
{
     float3 v[9];
     float3 temp;
     v[0] = input[dispatchThreadId.xy + float2(-1.0, -
1.0)].rgb;
     v[1] = input[dispatchThreadId.xy + float2(0.0, -
1.0)].rgb;
     v[2] = input[dispatchThreadId.xy + float2(+1.0, -
1.0)].rgb;
     v[3] = input[dispatchThreadId.xy + float2(-1.0,
0.0)].rgb;
     v[4] = input[dispatchThreadId.xy + float2(0.0, 0.0)].rgb;
     v[5] = input[dispatchThreadId.xy + float2(+1.0,
0.0)].rgb;
     v[6] = input[dispatchThreadId.xy + float2(-1.0,
+1.0)].rgb;
     v[7] = input[dispatchThreadId.xy + float2(0.0,
+1.0)].rgb;
     v[8] = input[dispatchThreadId.xy + float2(+1.0,
+1.0)].rgb;

     int count, n = 9, m;

     for (count = 0; count < n - 1; ++count)
          for (m = 0; m < n - count - 1; ++m){
               if (v[m].r > v[m + 1].r){
                    temp.r = v[m].r;
                    v[m].r = v[m + 1].r;
                    v[m + 1].r = temp.r;
               }
          }
     for (count = 0; count < n - 1; ++count)
          for (m = 0; m < n - count - 1; ++m){
               if (v[m].g > v[m + 1].g){
                    temp.g = v[m].g;
                    v[m].g = v[m + 1].g;
                    v[m + 1].g = temp.g;
```

โปรแกรมที่ 1.27 Median (ต่อ)

```
        }
            }

        for (count = 0; count < n - 1; ++count)
            for (m = 0; m < n - count - 1; ++m){
                if (v[m].b > v[m + 1].b){
                    temp.b = v[m].b;
                    v[m].b = v[m + 1].b;
                    v[m + 1].b = temp.b;
                }
            }

        output[dispatchThreadId.xy] = float4(v[4].rgb, 1);
}
```

โปรแกรมที่ 1.28 Conservative

```
Texture2D<float4> input : register(t0);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void SinglePassCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_DispatchThreadID)
{
     float3 v[25];
     int i;
     float3 min;
     float3 max;
     float3 temp;

     v[0] = input[dispatchThreadId.xy + float2(-2.0,
+2.0)].rgb;
     v[1] = input[dispatchThreadId.xy + float2(-1.0,
+2.0)].rgb;
     v[2] = input[dispatchThreadId.xy + float2(0.0,
+2.0)].rgb;
     v[3] = input[dispatchThreadId.xy + float2(+1.0,
+2.0)].rgb;
     v[4] = input[dispatchThreadId.xy + float2(+2.0,
+2.0)].rgb;
     v[5] = input[dispatchThreadId.xy + float2(-2.0,
1.0)].rgb;
     v[6] = input[dispatchThreadId.xy + float2(-1.0,
+1.0)].rgb;
     v[7] = input[dispatchThreadId.xy + float2(0.0,
+1.0)].rgb;
     v[8] = input[dispatchThreadId.xy + float2(+1.0,
+1.0)].rgb;
     v[9] = input[dispatchThreadId.xy + float2(+2.0,
+1.0)].rgb;
     v[10] = input[dispatchThreadId.xy + float2(-2.0,
0.0)].rgb;
     v[11] = input[dispatchThreadId.xy + float2(-1.0,
0.0)].rgb;
     v[12] = input[dispatchThreadId.xy + float2(0.0,
0.0)].rgb;
     v[13] = input[dispatchThreadId.xy + float2(+1.0,
0.0)].rgb;
     v[14] = input[dispatchThreadId.xy + float2(+2.0,
0.0)].rgb;
```

โปรแกรมที่ 1.28 Conservative (ต่อ)

```
v[15] = input[dispatchThreadId.xy + float2(-2.0, -1.0)].rgb;
     v[16] = input[dispatchThreadId.xy + float2(-1.0, -
1.0)].rgb;
     v[17] = input[dispatchThreadId.xy + float2(0.0, -
1.0)].rgb;
     v[18] = input[dispatchThreadId.xy + float2(+1.0, -
1.0)].rgb;
     v[19] = input[dispatchThreadId.xy + float2(+2.0, -
1.0)].rgb;
     v[20] = input[dispatchThreadId.xy + float2(-2.0, -
2.0)].rgb;
     v[21] = input[dispatchThreadId.xy + float2(-1.0, -
2.0)].rgb;
     v[22] = input[dispatchThreadId.xy + float2(0.0, -
2.0)].rgb;
     v[23] = input[dispatchThreadId.xy + float2(+1.0, -
2.0)].rgb;
     v[24] = input[dispatchThreadId.xy + float2(+2.0, -
2.0)].rgb;

     min.r = v[0].r;
     min.g = v[0].r;
     min.b = v[0].g;
     max.r = v[0].g;
     max.g = v[0].b;
     max.b = v[0].b;

     temp.r = v[12].r;
     temp.g = v[12].g;
     temp.b = v[12].b;

     //red
     for (i = 0; i < 24; i++){
          if (i != 12){
               if (min.r > v[i].r){
                    min.r = v[i].r;
               }
          }
     }
     for (i = 0; i < 24; i++){
          if (i != 12){
               if (max.r < v[i].r){
                    max.r = v[i].r;
               }
```

โปรแกรมที่ 1.28 Conservative (ต่อ)

```
if (v[12].r > max.r){
          v[12].r = max.r;
     }

     else if (v[12].r < min.r){
          v[12].r = min.r;
     }
     //green
     for (i = 0; i < 24; i++){
          if (i != 12){
               if (min.g > v[i].g){
                    min.g = v[i].g;
               }
          }
     }
     for (i = 0; i < 24; i++){
          if (i != 12){
               if (max.g < v[i].g){
                    max.g = v[i].g;
               }
          }
     }

     if (v[12].g > max.g){
          v[12].g = max.g;
     }

     else if (v[12].g < min.g){
          v[12].g = min.g;
     }
     for (i = 0; i < 24; i++){
          if (i != 12){
               if (max.g < v[i].g){
                    max.g = v[i].g;
               }
          }
     }

     if (v[12].g > max.g){
          v[12].g = max.g;
     }
```

โปรแกรมที่ 1.28 Conservative (ต่อ)

```
else if (v[12].g < min.g){
          v[12].g = min.g;
     }

     if (v[12].b > max.b){
          v[12].b = max.b;
     }

     else if (v[12].b < min.b){
          v[12].b = min.b;
     }
     output[dispatchThreadId.xy] = float4(v[12].rgb, 1);
}
```

โปรแกรมที่ 1.29 Texturer

```
Texture2D<float4> input : register(t0);
Texture2D<float4> input1: register(t1);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void TextureCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_dispatchThreadID)
{
     p;
     f;
     float4 pp = p;
     float4 ff = f;
     float4 sample = input[dispatchThreadId.xy];
     float4 sample1 = input1[dispatchThreadId.xy];
    float3 target = ((sample*pp)+(sample*ff*sample1));
     output[dispatchThreadId.xy] = float4(target, sample.a);
}
```

โปรแกรมที่ 1.29 Textured Merge

```
Texture2D<float4> input : register(t0);
Texture2D<float4> input1: register(t1);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void TextureCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_dispatchThreadID)
{
     float4 sample = input[dispatchThreadId.xy];
     float4 sample1 = input1[dispatchThreadId.xy];
     float3 hsl = 0;
     float u, v;
     u = -min(sample.r, min(sample.g, sample.b));
     v = max(sample.r, max(sample.g, sample.b));
     hsl.z = (v - u) * 0.5;
     float C = v + u;
     if (C != 0)
     {
          hsl.y = C / (1 - abs(2 * hsl.z - 1));
          float3 Delta = (hsl.z - sample.rgb) / C;
               Delta.rgb -= Delta.brg;
          Delta.rg += float2(2, 4);
          if (sample.r >= hsl.z)
               hsl.x = Delta.b;
          else if (sample.g >= hsl.z)
               hsl.x = Delta.r;
          else
               hsl.x = Delta.g;
          hsl.x = frac(hsl.x / 6);
     }
     h;
     float hh = (float)h / 360;
     hsl.x = hh;
     float ic = (1 - abs(2 * hsl.z - 1)) * hsl.y;
     float3 rgb, dev;
     rgb.r = abs(hsl.x * 6 - 3) - 1;
     rgb.g = 2 - abs(hsl.x * 6 - 2);
     rgb.b = 2 - abs(hsl.x * 6 - 4);
     dev = saturate(rgb);
```

โปรแกรมที่ 1.29 Textured Merge (ต่อ)

```
float3 target1;
     target1 = (dev - 0.5)*ic + hsl.z;
     float3 hsl1 = 0;
     float u1, v1;
     u1 = -min(sample.r, min(sample.g, sample.b));
     v1 = max(sample.r, max(sample.g, sample.b));
     hsl1.z = (v1 - u1) * 0.5;
     float C1 = v1 + u1;
     if (C1 != 0)
     {
          hsl1.y = C1 / (1 - abs(2 * hsl1.z - 1));
          float3 Delta1 = (hsl1.z - sample.rgb) / C1;
               Delta1.rgb -= Delta1.brg;
          Delta1.rg += float2(2, 4);
          if (sample.r >= hsl1.z)
               hsl1.x = Delta1.b;
          else if (sample.g >= hsl1.z)
               hsl1.x = Delta1.r;
          else
          hsl1.x = Delta1.g;
          hsl1.x = frac(hsl1.x / 6);
     }
     h1;
     float hh1 = (float)h1 / 360;
     hsl1.x = hh1;
     float ic1 = (1 - abs(2 * hsl1.z - 1)) * hsl1.y;
     float3 rgb1, dev1;
     rgb1.r = abs(hsl1.x * 6 - 3) - 1;
     rgb1.g = 2 - abs(hsl1.x * 6 - 2);
     rgb1.b = 2 - abs(hsl1.x * 6 - 4);
     dev1 = saturate(rgb1);
     float3 target2;
     target2 = (dev1 - 0.5)*ic1 + hsl1.z;
    float3 target = (sample1*target2)+(target1)*(1.0-target2);
     output[dispatchThreadId.xy] = float4(target, sample.a);
}
```

โปรแกรมที่ 1.30 Textured Filter

```
Texture2D<float4> input : register(t0);
Texture2D<float4> input1: register(t1);
RWTexture2D<float4> output : register(u0);
[numthreads(THREADSX, THREADSY, 1)]
void TextureCS(uint groupIndex: SV_GroupIndex,
     uint3 groupId : SV_GroupID,
     uint3 groupThreadId : SV_GroupThreadID,
     uint3 dispatchThreadId : SV_dispatchThreadID){
     float4 sample = input[dispatchThreadId.xy];
     float4 sample1 = input1[dispatchThreadId.xy];
          float3 hsl = 0;
          float u, v;
     u = -min(sample.r, min(sample.g, sample.b));
     v = max(sample.r, max(sample.g, sample.b));
     hsl.z = (v - u) * 0.5;
     float C = v + u;
     if (C != 0){
          hsl.y = C / (1 - abs(2 * hsl.z - 1));
          float3 Delta = (hsl.z - sample.rgb) / C;
               Delta.rgb -= Delta.brg;
          Delta.rg += float2(2, 4);
          if (sample.r >= hsl.z)
               hsl.x = Delta.b;
          else if (sample.g >= hsl.z)
               hsl.x = Delta.r;
          else
               hsl.x = Delta.g;
          hsl.x = frac(hsl.x / 6);}h;
     float hh = (float)h / 360;
     hsl.x = hh;
     float ic = (1 - abs(2 * hsl.z - 1)) * hsl.y;
     float3 rgb, dev;
     rgb.r = abs(hsl.x * 6 - 3) - 1;
     rgb.g = 2 - abs(hsl.x * 6 - 2);
     rgb.b = 2 - abs(hsl.x * 6 - 4);
     dev = saturate(rgb);
     float3 target1;
     target1 = (dev - 0.5)*ic + hsl.z;
     float3 target = ((0.01*(sample*sample1)) + (target1*(1 -
sample1)) + 0 + target1);
     output[dispatchThreadId.xy] = float4(target, sample.a);
}
```