การสร้างภาพเชิงปริมาตรบนระบบคลัสเตอริ่งโดยวิธีการแปลงเฉือนและบิด

FAST VOLUME RENDERING ON CLUSTERING SYSTEM USING A SHEAR-WARP FACTORIZATION

นพรัตน์ พันธุ์เสนา

NOPPARAT PANTSAENA

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พศ. 2547

ISBN 974-9709-73-X

FAST VOLUME RENDERING ON CLUSTERING SYSTEM USING A SHEAR-WARP FACTORIZATION

NOPPARAT PANTSAENA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRONIC ENGINEERING

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2004

ISBN 974-9709-73-X

หัวข้อวิทยานิพนธ์	**การสร้างภาพเชิงปริมาตรบนระบบคลัสเตอริ่ง โดยวิธีการแปลงเฉือนและบิด**
นักศึกษา	**นายนพรัตน์ พันธุ์เสนา**
รหัสประจำตัว	45061129
ปริญญา	**วิศวกรรมศาสตรมหาบัณฑิต**
สาขาวิชา	**วิศวกรรมอิเล็กทรอนิกส์**
พ.ศ.	2547
อาจารย์ผู้ควบคุมวิทยานิพนธ์	**รศ.ดร. มนัส สังวรศิลป์**
	ผศ.ดร. ชูชาติ ปิณฑวิรุจน์

บทคัดย่อ

การสร้างภาพเชิงปริมาตรเป็นเทคนิคที่ทำให้สามารถขยายขอบเขตการมองเห็นโครงสร้างภายในของวัตถุได้ดี แต่ขั้นตอนในการคำนวณมีความซับซ้อนและต้องการเวลาในการสร้างภาพนาน อีกทั้งยังต้องการความสามารถของคอมพิวเตอร์ความเร็วสูงเพื่อใช้ในการคำนวณ แต่คอมพิวเตอร์ความเร็วสูงนั้นราคาแพงมาก ดังนั้นในวิทยานิพนธ์ฉบับนี้จึงได้นำเอาหลักการของระบบประมวลผลแบบขนานบนคอมพิวเตอร์หลายเครื่องที่เรียกว่าระบบคลัสเตอริ่ง และระเบียบวิธีการสร้างภาพเชิงปริมาตรแบบการแปลงเฉือนและบิดมาประยุกต์ใช้ เพื่อให้การสร้างภาพเชิงปริมาตรทำได้รวดเร็วที่สุด สำหรับกระบวนการแบ่งงานในระบบประมวลผลแบบขนาน ใช้วิธีการที่เรียกว่า Work Pool เพื่อกระจายงานไปยังคอมพิวเตอร์เครื่องต่าง ๆ ในระบบ โดยปริมาณงานที่แต่ละเครื่องได้รับจะขึ้นอยู่กับความเร็วในการประมวลผลของแต่ละเครื่อง ทำให้การทำงานรวมของระบบทั้งหมดเสร็จภายในเวลาใกล้เคียงกัน งานวิจัยได้มุ่งความสนใจไปที่ปริมาตรที่มีขนาดใหญ่มาก โดยปริมาตรของข้อมูลภาพเริ่มต้นคือ 587x341x1877 ว็อกเซล เป็นข้อมูลภาพระดับเทา 8 บิต และผลลัพธ์ที่ได้เป็นภาพสี 24 บิต จากผลการทดลองระบบการคำนวณในวิทยานิพนธ์นี้สามารถทำให้การคำนวณรวดเร็วขึ้น 6.97 เท่า เมื่อเปรียบเทียบกับวิธีการสร้างภาพเชิงปริมาตรแบบฉายแสงบนคอมพิวเตอร์ขนานระบบเดียวกัน และทำให้การคำนวณรวดเร็วขึ้น 38.81 เท่า เมื่อเทียบกับการประมวลผลบนคอมพิวเตอร์หน่วยประมวลผลเดี่ยวโดยใช้วิธีฉายแสง

Thesis Title	Fast Volume Rendering on Clustering System using A Shear-Warp Factorization
Student	Mr. Nopparat Pantsaena
Student ID.	45061129
Degree	Master of Engineering
Programme	Electronic Engineering
Year	2004
Thesis Advisor	Assoc.Prof.Dr. Manas Sangworasil
	Asst.Prof.Dr. Chuchart Pintavirooj

ABSTRACT

The volume rendering has the advantage of visualizing the internal structure of an object and being high image quality. However, the time consuming and the complexity of the rendering process are the crucial problems. As a result, the rendering process requires very high performance of the computer. Therefore, in this thesis, concepts of parallel programming method and shear-warp factorization algorithm, are employed to speed up a medical volume rendering process. The scheduling process in the clustering system is improved. The appropriate amount of work is distributed to each processor in the clustering system using a work-pool scheduling scheme. In this scheme, even though the work load for each node in clustering system is uneven, the variation of the processing time for each node is insignificant. In the experiment, the result image is a whole body rendered from large volumetric data. The input image is 8-bit gray scale and volume's size is 587x341x1877 voxels whereas the output image is 24-bit colors. Our proposed system works successfully with decreasing the rendering time up to 6.97 times compared with the ray casting on the clustering system and 38.81 times compared with the ray casting on the single machine.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี เนื่องมากจากความช่วยเหลือจากบุคคลหลายฝ่ายด้วยกัน

ขอบคุณ ผศ.ดร. จีรพร ศรีสวัสดิ์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ได้ช่วยอธิบายในส่วนทฤษฎีของ Parallel processing และช่วยตรวจทานการทดลองในวิทยานิพนธ์เล่มนี้

ขอบคุณ ผศ.ดร.วรา วราวิทย์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ คุณชมพล งามผิว คุณสุริยะ อุรุเอกโอฬาร ที่ได้ช่วยอธิบายเรื่องของ Parallel computing ให้เข้าใจมากยิ่งขึ้น

ขอบคุณ คุณนนท์ บัณฑิตวงษ์ และอาจารย์เกษมสุข เสพสิริสุข ที่ได้ช่วยอธิบายในเรื่องพื้นฐานเกี่ยวกับการประมวลผลขนาน และทฤษฎีการสร้างภาพเชิงปริมาตร ที่สำคัญพี่ทั้งสองทำให้ผมไม่ต้องเริ่มต้นการทำวิทยานิพนธ์นี้จากศูนย์

ขอบคุณ คุณธเนศ ชูวิเศษวณิชย์ ที่ได้ช่วยปรับปรุงรูปแบบของวิทยานิพนธ์เล่มนี้ให้ถูกต้องและสามารถแก้ไขได้ง่ายมากยิ่งขึ้น

ขอบคุณคุณวรัญญา พิรุณสาร ผู้ที่คอยเตือนในเรื่องของตารางเวลาต่างๆ และอื่น ๆ อีกมาก

สุดท้ายขอขอบพระคุณพ่อและแม่ ที่ให้ชีวิต ความรัก ความอบอุ่น การอบรมสั่งสอน กำลังใจ และอื่น ๆ อีกมากมาย ซึ่งเป็นสิ่งที่ทำให้ลูกคนนี้มีทั้ง IQ และ EQ สามารถอยู่ในสังคมได้อย่างมีความสุข

นพรัตน์ พันธุ์เสนา

พ.ค. 2547

สารบัญ

หน้า

บทคัดย่อ I

ABSTRACT II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง X

สารบัญรูป XI

บทที่ 1 บทนำ 1

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา 2

1.3 ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย 2

1.4 ขอบเขตการวิจัยและเค้าโครงวิทยานิพนธ์ 2

บทที่ 2 หลักการสร้างภาพเชิงปริมาตรเบื้องต้น 4

2.1 บทนำ 4

2.2 ความรู้เบื้องต้นเกี่ยวกับภาพเชิงพื้นผิวและภาพเชิงปริมาตร 4

2.2.1 ภาพเชิงพื้นผิว 4

2.2.2 ภาพเชิงปริมาตร 5

2.2.3 สมการการสร้างภาพเชิงปริมาตร 7

2.2.4 สมการการสร้างภาพเชิงปริมาตรแบบไม่ต่อเนื่อง 10

2.3 วิธีการสร้างภาพเชิงปริมาตร 11

2.3.1 วิธีการแบบลำดับภาพ (Image Order Algorithm) 12

2.3.2 วิธีการแบบลำดับวัตถุ (Object Order Algorithm) 13

2.4 สรุป 13

สารบัญ (ต่อ)

หน้า

บทที่ 3 การสร้างภาพเชิงปริมาตรแบบลำดับภาพ 14
3.1 บทนำ 14
3.2 การสร้างภาพโดยใช้วิธีการฉายแสง (Ray Casting) 14
3.3 การอินเทอร์โพเลชั่น (Interpolation) 17
3.3.1 การอินเทอร์โพเลชั่นแบบเนียเรสเนเบอร์ (Nearest Neighbor) 18
3.3.2 การอินเทอร์โพเลชั่นแบบเชิงเส้น (Linear Interpolation) 18
3.3.3 การอินเทอร์โพเลชั่นแบบคิวบิคคอนโวลูชั่น (Cubic Convolution) 20
3.4 การแบ่งกลุ่มข้อมูล 22
3.4.1 เกรเดียนท์ (Gradient) 22
3.4.2 นอร์มอลเวคเตอร์ (Normal Vector) 24
3.4.3 ฮีสโตรแกรม (Histogram) 24
3.4.4 ฟังก์ชั่นถ่ายโอน (Transfer Function) 25
3.4.5 การให้สี 27
3.5 การส่องสว่าง และ การให้แสงเงา 28
3.5.1 แสงแวดล้อม (Ambient Light) 28
3.5.2 การสะท้อนแบบกระจาย (Diffuse Reflection) 29
3.5.3 การสะท้อนแบบกระจก (Specular Reflection) 29
3.6 การประกอบภาพ 31
3.6.1 การประกอบภาพแบบหน้าไปหลัง 31
3.6.2 การประกอบภาพแบบหลังไปหน้า 33
3.7 สรุป 34

บทที่ 4 การสร้างภาพเชิงปริมาตรแบบลำดับวัตถุ 35
4.1 บทนำ 35
4.2 เทคนิคการเพิ่มความเร็วในการสร้างภาพเชิงปริมาตร 35
4.2.1 การใช้รูปแบบโครงสร้างข้อมูลแบบพิเศษ (Special Data Structure) 35
4.2.2 การสิ้นสุดลำแสงก่อน (Early Ray Termination) 37

สารบัญ (ต่อ)

หน้า

4.3 การสร้างภาพเชิงปริมาตรแบบเร็วโดยใช้วิธีการแปลงเฉือนและบิด 38
4.4 หลักการแปลงเฉือนและบิด 40
4.4.1 ระบบพิกัดที่เกี่ยวข้อง 41
4.4.2 การหาแกนมุมมองหลัก 42
4.4.3 การแปลงไปยังพิกัดวัตถุมาตรฐาน 44
4.4.4 เมตริกซ์การเฉือนและบิด 45
4.4.5 การฉายภาพไปยังระนาบภาพระหว่างกลาง 46
4.4.6 เมตริกซ์การแปลงที่สมบูรณ์ 48
4.5 การประกอบภาพไปยังภาพระหว่างกลาง 49
4.5.1 การประกอบภาพแบบเร็วโดยใช้การเข้ารหัสแบบรันเลงท์ (Run-length) 49
4.5.2 การแบ่งกลุ่มข้อมูล 51
4.5.3 การอินเทอร์โพเลชั่น 52
4.5.4 การค้นหาค่าสี และแสงเงาจากตาราง 54
4.6 คุณสมบัติที่สำคัญของวิธีการสร้างภาพแบบลำดับวัตถุ 57
4.7 สรุป 58

บทที่ 5 โครงสร้างของคอมพิวเตอร์ขนานแบบคลัสเตอร์ 59
5.1 บทนำ 59
5.2 ความสำคัญของคอมพิวเตอร์ความเร็วสูง 59
5.3 ชนิดและโครงสร้างของคอมพิวเตอร์แบบขนาน 60
5.3.1 ระบบแบบหลายหน่วยประมวลผล (Multiprocessor) 61
5.3.2 ระบบแบบหลายเครื่องประมวลผล (Multi computer) 62
5.4 ส่วนประกอบโดยรวมของคอมพิวเตอร์แบบคลัสเตอร์ 67
5.5 การจำแนกคอมพิวเตอร์แบบคลัสเตอร์ 68
5.5.1 จำแนกตามงานที่ประยุกต์ใช้ 68
5.5.2 จำแนกตามลักษณะของเครื่องในระบบ 69

สารบัญ (ต่อ)

หน้า

5.6 โครงสร้างพื้นฐานของระบบคลัสเตอร์ 70
5.6.1 ส่วนประกอบทางฮาร์ดแวร์ 70
5.6.2 ระบบปฏิบัติการ 70
5.6.3 คลัสเตอร์มิดเดิลแวร์ 71
5.6.4 การเชื่อมต่อเครือข่าย 73
5.7 เครื่องมือและโปรแกรมอัตถะประโยชน์ 77
5.7.1 โปรแกรมสื่อสารระหว่างโหนดโดยการส่งข้อความ 77
5.7.2 คอมไพเลอร์ 78
5.7.3 ชุดคำสั่งทางคณิตศาสตร์ 78
5.8 การจัดการระบบคลัสเตอร์ 79
5.9 สรุป 80

บทที่ 6 โปรแกรมแบบขนาน 81
6.1 บทนำ 81
6.2 รูปแบบของการพัฒนาโปรแกรมแบบขนาน 81
6.2.1 การสร้างโปรแกรมแบบขนานโดยอัตโนมัติ 81
6.2.2 การสร้างโปรแกรมโดยใช้ชุดคำสั่งแบบขนาน 82
6.2.3 การสร้างโปรแกรมแบบขนานด้วยตัวเอง 83
6.3 กลวิธีการโปรแกรมแบบขนาน 83
6.4 การออกแบบโปรแกรมขนาน 87
6.4.1 ขั้นตอนการแบ่งงาน 88
6.4.2 ขั้นตอนออกแบบการสื่อสาร 89
6.4.3 ขั้นตอนการรวมกลุ่มงาน 91
6.4.4 ขั้นตอนการกำหนดงาน 92

สารบัญ (ต่อ)

หน้า

6.5 การสร้างภาพเชิงปริมาตรแบบฉายแสงบนระบบคลัสเตอร์ 93
6.5.1 การแบ่งงานในการสร้างภาพเชิงปริมาตร 93
6.5.2 การรวมกลุ่มงาน 94
6.5.3 การสื่อสารระหว่างงานในการสร้างภาพเชิงปริมาตร 94
6.5.4 การกำหนดงานในการสร้างภาพเชิงปริมาตร 94
6.6 สรุป 95

บทที่ 7 การทดลองและผลการทดลอง 96
7.1 บทนำ 96
7.2 การสร้างภาพเชิงปริมาตรโดยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์ 97
7.2.1 การออกแบบโปรแกรมการสร้างภาพแบบขนาน 98
7.2.2 ระบบคลัสเตอร์ที่ใช้ในการทดลอง 102
7.2.3 ข้อมูลเชิงปริมาตรที่ใช้ในการทดลอง 103
7.3 การวัดประสิทธิภาพของการสร้างภาพเชิงปริมาตรบนระบบคลัสเตอร์ 103
7.4 การทดลองและผลการทดลอง 105
7.4.1 ผลการทดลองเปรียบเทียบกับระบบอุดมคติ 105
7.4.2 การทดลองเปรียบเทียบระหว่างการเก็บข้อมูลแบบศูนย์กลางและแบบกระจาย 107
7.4.3 การทดลองเปรียบเทียบระหว่างวิธีการฉายแสงกับการแปลงเฉือนและบิด 110
7.4.4 การทดลองเปรียบเทียบเมื่อขนาดก้อนปริมาตรต่างกัน 112
7.4.5 การทดลองเปรียบเทียบเมื่อค่า Threshold ต่างกัน 114

บทที่ 8 สรุปผลและแนวทางการพัฒนา 118
8.1 สรุปและวิจารณ์ผลการทดลอง 118
8.2 แนวทางการพัฒนา 119

สารบัญ (ต่อ)

หน้า

บรรณานุกรม 120

ภาคผนวก ก. 123

ภาคผนวก ข. 129

ประวัติผู้เขียน 130

สารบัญตาราง

หน้า

ตารางที่ 5.1 เปรียบเทียบความแตกต่างของคอมพิวเตอร์แบบกระจาย (Buyya, 1999)..... 66
ตารางที่ ก.1 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิด
บนระบบคลัสเตอร์ (Cubic Size = 32).....124
ตารางที่ ก.2 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิด
บนระบบคลัสเตอร์ (Cubic Size = 64).....125
ตารางที่ ก.3 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิด
บนระบบคลัสเตอร์ (Cubic Size = 128).....126
ตารางที่ ก.4 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิด
บนระบบคลัสเตอร์ (Cubic Size = 256).....127
ตารางที่ ก.5 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการฉายแสงบนระบบคลัสเตอร์.....128

สารบัญรูป

หน้า

รูปที่ 2.1 ภาพตัดขวางจากเครื่องถ่ายภาพแบบ CT 6
รูปที่ 2.2 ตัวอย่างภาพผลลัพธ์จากการสร้างภาพเชิงปริมาตร 7
รูปที่ 2.3 ภาพแสดงเหตุการณ์จำลองการเกิดภาพในธรรมชาติ 7
รูปที่ 2.4 สถานการณ์จำลองตามสมการการสร้างภาพเชิงปริมาตร 9

รูปที่ 3.1 การสร้างภาพเชิงปริมาตรโดยวิธีการฉายแสง 15
รูปที่ 3.2 ขั้นตอนในการสร้างภาพเชิงปริมาตร 15
รูปที่ 3.3 การฉายแสงในสองมิติโดยแปลงตำแหน่งของระนาบการมอง 16
รูปที่ 3.4 (ก) เคอร์เนลแบบเนียเรสเนเบอร์ (ข) การอินเทอร์โพเลชั่นด้วยวิธีเนียเรสเนเบอร์ 18
รูปที่ 3.5 ความสัมพันธ์แบบเชิงเส้นของข้อมูล 19
รูปที่ 3.6 การอินเทอร์โพเลชั่นในสามมิติ 20
รูปที่ 3.7 ระยะทางระหว่างจุดสี่จุดที่ใช้ในการอินเทอร์โพเลชั่น 21
รูปที่ 3.8 คิวบิคคอนโวลูชั่นในสองมิติ 22
รูปที่ 3.9 แสดงการหาค่าเวกเตอร์ปกติจากเกรเดียนท์ของพื้นผิว 23
รูปที่ 3.10 ฮีสโตแกรมของข้อมูลภาพทางการแพทย์ที่ถ่ายด้วยเครื่อง CT 24
รูปที่ 3.11 ฟังก์ชันถ่ายโอนของ Levoy 26
รูปที่ 3.12 การแยกแยะชนิดของอวัยวะซึ่งมีขอบเขตผสมกัน 26
รูปที่ 3.13 ฟังก์ชันถ่ายโอนของข้อมูลที่มีมากกว่าหนึ่งวัตถุ 27
รูปที่ 3.14 ผลของแสงแวดล้อมที่มีต่อวัตถุ ซึ่งมีค่า k_a ต่าง ๆ กัน 28
รูปที่ 3.15 ผลของการสะท้อนแสงแบบกระจายร่วมกับผลของแสงแวดล้อมที่มีต่อวัตถุที่มีค่า k_d ต่าง ๆ กัน และแสงแวดล้อมมีค่า $k_a = 0.3$ 29
รูปที่ 3.16 องค์ประกอบการสะท้อนแบบกระจก 30
รูปที่ 3.17 ผลของการสะท้อนแสงแบบกระจกร่วมกับผลของการสะท้อนแบบกระจาย และแสงแวดล้อมที่มีต่อวัตถุ 30
รูปที่ 3.18 ลักษณะและทิศทางของการประกอบภาพแบบหน้าไปหลัง 32
รูปที่ 3.19 ลักษณะและทิศทางของการประกอบภาพแบบหลังไปหน้า 33

สารบัญรูป (ต่อ)

หน้า

รูปที่ 4.1 หลักการสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิด 38
รูปที่ 4.2 การฉายภาพแบบขนานโดยใช้การแปลงเฉือนและบิด 39
รูปที่ 4.3 แสดงวิธีการสร้างภาพเชิงปริมาตรโดยใช้การแปลงแบบเฉือนและบิด 40
รูปที่ 4.4 ระบบพิกัดต่าง ๆ ที่ใช้ในการแปลงแบบเฉือนและบิด 41
รูปที่ 4.5 การพิจารณาหาค่าสัมประสิทธิ์สำหรับการแปลงเฉือน 45
รูปที่ 4.6 การแปลงพิกัดภาพระหว่างกลางในกรณีต่าง ๆ 47
รูปที่ 4.7 รหัสรันเลงท์ของข้อมูลแบบไบนารี 50
รูปที่ 4.8 ขั้นตอนการประกอบภาพโดยใช้การเข้ารหัสรันเลงท์และการสิ้นสุดลำแสงก่อน 51
รูปที่ 4.9 แสดงการอินเทอร์โพเลชั่นแบบไบลีเนียร์ในการฉายภาพ 53
รูปที่ 4.10 วิธีการกำหนดค่าสีโดยค้นหาจากตาราง 54
รูปที่ 4.11 ตารางกริดและพีระมิดแปดหน้าที่ถูกใช้เพื่อการเข้ารหัสนอร์มอลเว็ดเตอร์ 56

รูปที่ 5.1 โครงสร้างการทำงานของสถาปัตยกรรมแบบหลายหน่วยประมวลผล 61
รูปที่ 5.2 โครงสร้างการทำงานของสถาปัตยกรรมแบบหลายเครื่องประมวลผล 63
รูปที่ 5.3 บล็อกไดอะแกรมแสดงส่วนประกอบของระบบคลัสเตอร์ 67
รูปที่ 5.4 กราฟแสดงการกระจายงานระหว่างที่ประมวลผลในระบบคลัสเตอร์มิดเดิลแวร์ 73
รูปที่ 5.5 กราฟแสดงความเร็วในการส่งข้อมูลของเครือข่ายของ Quadrics 74
รูปที่ 5.6 กราฟแสดงค่า Latency Time ของเครือข่ายของ Quadrics 74
รูปที่ 5.7 การเชื่อมต่อกับเครือข่ายภายนอกผ่านเกตเวย์ 75
รูปที่ 5.8 การเชื่อมต่อแบบทุกเครื่องเชื่อมต่อกับเครือข่ายภายนอก 76
รูปที่ 5.9 ส่วนประกอบของชุดคำสั่ง PETSc 78

รูปที่ 6.1 ขั้นตอนการทำงานของการสร้างโปรแกรมแบบขนานโดยอัตโนมัติ 82
รูปที่ 6.2 ขั้นตอนการสร้างโปรแกรมโดยใช้ชุดคำสั่งแบบขนาน 83
รูปที่ 6.3 ขั้นตอนการสร้างโปรแกรมแบบขนานด้วยตัวเอง 83
รูปที่ 6.4 ขั้นตอนการออกแบบการทำงานของโปรแกรมขนาน 87
รูปที่ 6.5 การแบ่งขอบเขตหน้าที่ของปัญหา 88

สารบัญรูป (ต่อ)

หน้า

รูปที่ 6.6 ลักษณะการสื่อสารในการคำนวณไฟไนต์ดิฟเฟอเรนต์ในสองมิติ 89
รูปที่ 6.7 การสื่อสารเพื่อรวมค่าโดยงานที่เป็นผู้จัดการ 90
รูปที่ 6.8 ตาราง (Grid) ที่สร้างขึ้นเพื่อวิเคราะห์ไฟไนต์อีลีเมนต์ 90
รูปที่ 6.9 การรวมกลุ่มงานในการประมวลผลแบบขนานเพื่อลดเวลาในการสื่อสาร 91
รูปที่ 6.10 การแบ่งข้อมูลของปัญหาออกเป็นส่วนย่อยตามแนวลำแสง 93
รูปที่ 6.11 การกำหนดงานแบบศูนย์รวมงาน (Work Pool) 94

รูปที่ 7.1 การสร้างภาพเชิงปริมาตรบนระบบคลัสเตอริ่งโดยวิธีการแปลงเฉือนและบิด 98
รูปที่ 7.2 การแบ่งงานเป็นส่วนย่อยโดยพิจารณาที่ภาพผลลัพธ์ในการสร้างภาพแบบฉายแสง 99
รูปที่ 7.3 การแบ่งงานเป็นส่วนย่อยโดยแบ่งที่ปริมาตรโดยตรงในการสร้างภาพแบบแปลงเฉือนและบิด 99
รูปที่ 7.4 วิธีการกำหนดงานแบบ Work Pool 101
รูปที่ 7.5 ภาพที่ใช้ในการทดลอง 103
รูปที่ 7.6 แผนภูมิแสดงเวลาที่ลดลงของการทดลองเปรียบเทียบกับระบบอุดมคติ 105
รูปที่ 7.7 แผนภูมิแสดงอัตราการเพิ่มความเร็วของการทดลองเปรียบเทียบกับระบบอุดมคติ 106
รูปที่ 7.8 แสดงประสิทธิภาพของระบบที่ใช้ทดลองเปรียบเทียบกับระบบอุดมคติ 106
รูปที่ 7.9 ระบบที่มีการเก็บข้อมูลปริมาตรและโปรแกรมการทำงานไว้ที่ศูนย์กลาง 107
รูปที่ 7.10 ระบบที่มีการเก็บข้อมูลปริมาตรและโปรแกรมการทำงานแบบกระจาย 108
รูปที่ 7.11 แผนภูมิแสดงเวลาที่ลดลงเปรียบเทียบสองระบบที่เก็บข้อมูลต่างกัน 109
รูปที่ 7.12 แผนภูมิแสดงอัตราความเร็วที่เพิ่มขึ้นของสองระบบที่เก็บข้อมูลต่างกัน 109
รูปที่ 7.13 แผนภูมิแสดงประสิทธิภาพของสองระบบที่เก็บข้อมูลต่างกัน 109
รูปที่ 7.14 แผนภูมิแสดงเวลาที่ใช้สร้างภาพผลลัพธ์ของสองระบบที่ใช้วิธีการสร้างภาพต่างกัน 111
รูปที่ 7.15 แผนภูมิแสดงอัตราความเร็วที่เพิ่มขึ้นของสองระบบที่ใช้วิธีการสร้างภาพต่างกัน 111
รูปที่ 7.16 แผนภูมิแสดงประสิทธิภาพของระบบเมื่อใช้วิธีการสร้างภาพต่างกัน 112
รูปที่ 7.17 ขนาดของปริมาตรที่แบ่งในการทดลอง 113
รูปที่ 7.18 แผนภูมิแสดงเวลาที่ใช้ในการสร้างภาพเมื่อขนาดของก้อนปริมาตรแตกต่างกัน 113
รูปที่ 7.19 แผนภูมิแสดงเวลาที่ใช้ในการสร้างภาพเมื่อใช้ค่า Threshold ต่างกัน 114

สารบัญรูป (ต่อ)

หน้า

รูปที่ 7.20 ภาพผลลัพธ์ในแต่ละมุมมอง โดยที่ปริมาตรที่ใช้ในการสร้างภาพมีขนาด 587x341x1877 Voxel และภาพผลลัพธ์เป็นภาพสี 24 bit มีขนาด 1920x512 pixels 116
รูปที่ 7.21 ภาพผลลัพธ์ที่เลือกมาเฉพาะบางส่วนเพื่อให้เห็นรายละเอียดมากยิ่งขึ้น 117
รูปที่ 7.22 ภาพผลลัพธ์ในมุมมองอื่น ๆ 117

รูปที่ 8.1 แผนภูมิแสดงการเปรียบเทียบความเร็วในการสร้างภาพเชิงปริมาตรแบบต่าง ๆ 118

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การแสดงผลข้อมูลที่เรียกว่าการสร้างภาพเชิงปริมาตรนั้น เป็นวิธีการที่มีความสำคัญในการทำให้เกิดความเข้าใจในชุดของข้อมูลภาพเชิงปริมาตรได้ง่ายมากยิ่งขึ้น ดังนั้นจึงถูกนำมาประยุกต์ใช้ในงานหลายๆ ด้าน ไม่ว่าจะเป็นงานทางด้านการแพทย์ หรืองานทางด้านวิศวกรรมเครื่องกล กลศาสตร์ของไหล เป็นต้น การสร้างภาพเชิงปริมาตรนั้นจะประกอบด้วยสองขั้นตอนหลัก คือ การจัดการกับชุดของข้อมูลเชิงปริมาตร โดยทำการรีแซมปลิง (Re-Sampling) และการรวมค่าสีเพื่อไปปรากฏยังภาพผลลัพธ์ ซึ่งวิธีการเหล่านี้จะสิ้นเปลืองเวลาในการคำนวณและต้องการความสามารถในประมวลผลของคอมพิวเตอร์ที่สูงมาก ถึงแม้ว่าจะมีงานวิจัยที่เกี่ยวข้องกับงานทางด้านนี้ออกมาอย่างต่อเนื่อง เช่น การสร้างภาพเชิงปริมาตรแบบเฉือนและบิด การสร้างภาพเชิงปริมาตรโดยการแปลงระยะทาง ซึ่งเป็นเทคนิคที่ใช้ความสามารถของหน่วยประมวลผลเดี่ยวความเร็วสูง แต่ทั้งนี้ก็ยังต้องการเวลาช่วงหนึ่งในการสร้างภาพผลลัพธ์ขึ้นมาสำหรับปริมาตรขนาดใหญ่ ปัญหาที่สำคัญอีกอย่างหนึ่งคือ ความจำกัดในเรื่องของหน่วยความจำบนเครื่องคอมพิวเตอร์แบบหน่วยประมวลผลเดี่ยว ซึ่งทำให้ประสิทธิภาพของเครื่องเหล่านี้มีไม่เพียงพอที่จะประมวลผลกับข้อมูลภาพขนาดใหญ่ได้ เช่น ข้อมูลภาพเชิงปริมาตรแบบเต็มทั้งตัวของมนุษย์ เป็นต้น

ดังนั้นวิทยานิพนธ์ฉบับนี้จึงได้หาทางแก้ไขปัญหาต่าง ๆ ที่กล่าวมา โดยได้นำเสนอวิธีการการสร้างภาพเชิงปริมาตรแบบเร็วโดยใช้การแปลงเฉือนและบิดบนคอมพิวเตอร์แบบคลัสเตอร์ เป็นการนำเอาข้อดีของวิธีการสร้างภาพเชิงปริมาตรแบบเร็ว ให้สามารถทำงานบนระบบคลัสเตอร์ซึ่งเป็นระบบคอมพิวเตอร์ขนานแบบความเร็วสูงได้ ทำให้การสร้างภาพในแต่ละมุมมองเป็นไปได้อย่างรวดเร็ว อีกทั้งระบบที่สร้างขึ้นมานี้ยังสามารถรองรับการทำงานกับปริมาตรข้อมูลขนาดใหญ่ได้ เพราะการประมวลผลบนระบบคอมพิวเตอร์ขนาน จะทำการแบ่งงานขนาดใหญ่ออกเป็นงานย่อย ๆ เพื่อแยกกันประมวลผล แล้วค่อยนำผลลัพธ์กลับมารวมกันทีหลัง ดังนั้นไม่ว่างานจะมีขนาดใหญ่แค่ไหน ก็สามารถทำงานได้บนระบบการทำงานแบบนี้

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

- ทำการสร้างคอมพิวเตอร์ประมวลผลขนานแบบคลัสเตอร์ขึ้น ให้สามารถรองรับการทำงานในเกือบทุกรูปแบบ โดยเฉพาะกับการสร้างภาพเชิงปริมาตร
- พัฒนาซอฟต์แวร์ในการสร้างภาพเชิงปริมาตรให้สามารถทำงานได้รวดเร็ว โดยการทำงานแบบขนานบนระบบคลัสเตอร์ มีระบบการแบ่งงานที่ดี สามารถรองรับปริมาตรงานขนาดใหญ่ได้
- ระบบที่สร้างขึ้นมา สามารถที่จะเป็นต้นแบบในการพัฒนาโปรแกรมอื่น ๆ ต่อไป

1.3 ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย

แต่เดิมการสร้างภาพเชิงปริมาตรนั้นจะใช้วิธีการฉายแสง (Ray casting: Levoy M, 1988) บนคอมพิวเตอร์หน่วยประมวลผลเดี่ยว หรือการสร้างภาพเชิงปริมาตรแบบฉายแสงบนคอมพิวเตอร์แบบคลัสเตอร์ ซึ่งวิธีการเหล่านี้สามารถสร้างภาพได้รวดเร็วในระดับหนึ่ง แต่ยังต้องใช้เวลาในการประมวลผลค่อนข้างมาก ดังนั้นหากเราสามารถทำให้วิธีการสร้างภาพเชิงปริมาตรแบบเร็วที่มีประสิทธิภาพสูง สามารถทำงานบนระบบคอมพิวเตอร์แบบคลัสเตอร์ได้ ก็จะทำให้สามารถที่จะสร้างภาพผลลัพธ์ในแต่ละมุมมองได้รวดเร็วมากยิ่งขึ้น

วิธีการสร้างภาพเชิงปริมาตรที่ถูกเลือกนำมาใช้ในวิทยานิพนธ์ฉบับนี้คือ การสร้างภาพเชิงปริมาตรแบบเร็วโดยใช้การแปลงเฉือนและบิด ซึ่งเป็นวิธีการสร้างภาพเชิงปริมาตรสมัยใหม่ที่มีความรวดเร็วสูงมากแม้จะทำงานอยู่บนหน่วยประมวลผลเดี่ยว ดังนั้นเมื่อนำมาทำงานร่วมกับระบบคลัสเตอร์จึงทำให้มีความรวดเร็วเพิ่มมากขึ้นอีก

1.4 ขอบเขตการวิจัยและเค้าโครงวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้มุ่งเน้นที่จะเพิ่มความเร็วของการสร้างภาพเชิงปริมาตรโดยทำการสร้างภาพด้วยวิธีการประมวลผลแบบขนานบนระบบคลัสเตอร์ รายละเอียดต่าง ๆ ภายในวิทยานิพนธ์ได้จัดแบ่งเนื้อหาออกเป็น 8 บทซึ่งแต่ละบทมีหัวข้อและเนื้อหาดังต่อไปนี้

บทที่ 1 บทนำ

กล่าวถึงความเป็นมาและความสำคัญของปัญหา พร้อมทั้งกล่าวถึงวัตถุประสงค์และขอบเขต ของการวิจัยนี้

บทที่ 2 หลักการสร้างภาพเชิงปริมาตรเบื้องต้น

อธิบายถึงหลักการและทฤษฎีที่เกี่ยวข้องกับการสร้างภาพเชิงปริมาตรเบื้องต้น สมการการสร้างภาพ สมการการสร้างภาพแบบไม่ต่อเนื่อง (Discrete) ชนิดและวิธีการในการสร้างภาพเชิงปริมาตรทั้งการสร้างภาพเชิงปริมาตรแบบลำดับภาพ และแบบลำดับวัตถุ

บทที่ 3 การสร้างภาพเชิงปริมาตรแบบลำดับภาพ

ในบทนี้ได้กล่าวถึงการสร้างภาพเชิงปริมาตรโดยการฉายแสงซึ่งเป็นวิธีการสร้างภาพแบบลำดับภาพแบบหนึ่ง โดยจะกล่าวถึงขั้นตอนทั้งหมด ตั้งแต่การเตรียมข้อมูล การอินเทอร์โพเลชั่น การแบ่งกลุ่มข้อมูล การส่องสว่างและให้แสงเงา และการประกอบภาพ

บทที่ 4 การสร้างภาพเชิงปริมาตรแบบลำดับวัตถุ

เป็นการกล่าวถึงการสร้างภาพเชิงปริมาตรแบบเฉือนและบิด ซึ่งเป็นการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุแบบหนึ่ง โดยจะกล่าวถึงข้อดีของการใช้วิธีการสร้างภาพแบบนี้ และอธิบายหลักการทำงานขั้นตอนต่าง ๆ ของการสร้างภาพ

บทที่ 5 โครงสร้างของคอมพิวเตอร์ขนานแบบคลัสเตอร์

อธิบายถึงโครงสร้างการประมวลผลแบบขนาน ทั้งแบบหลายหน่วยประมวลผลและแบบหลายเครื่องประมวลผล การจำแนกชนิดของระบบคลัสเตอร์ ส่วนประกอบของระบบ ทั้งในส่วนพื้นฐาน เครื่องมือและโปรแกรมอัตถะประโยชน์ และการจัดการระบบคลัสเตอร์

บทที่ 6 การโปรแกรมแบบขนาน

กล่าวถึงลักษณะของโปรแกรมแบบขนาน วิธีการสร้างโปรแกรมแบบขนาน กลวิธีการโปรแกรมแบบขนาน การออกแบบโปรแกรมแบบขนาน ซึ่งประกอบด้วยการแบ่งงาน ขั้นตอนการออกแบบการสื่อสาร ขั้นตอนรวมกลุ่มงาน และขั้นตอนการกำหนดงาน หลังจากนั้นได้ทำการยกตัวอย่าง วิธีการออกแบบโปรแกรมการสร้างภาพเชิงปริมาตรโดยใช้วิธีการฉายแสงแบบขนาน

บทที่ 7 การทดลองและผลการทดลอง

เป็นการอธิบายถึงการทดลอง และการสร้างภาพเชิงปริมาตรโดยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์ การออกแบบโปรแกรม การวัดประสิทธิภาพของระบบ และผลการทดลอง

บทที่ 8 สรุปผลและแนวทางการพัฒนา

กล่าวถึงบทสรุปของวิทยานิพนธ์และแนวทางการวิจัยต่อไป

บทที่ 2
หลักการสร้างภาพเชิงปริมาตรเบื้องต้น

2.1 บทนำ

บทนี้จะเป็นการกล่าวถึงหลักการและวิธีการสร้างภาพเชิงปริมาตรเบื้องต้น โดยจะเริ่มจากการชี้ให้เห็นถึงความแตกต่างระหว่างภาพเชิงพื้นผิว และภาพเชิงปริมาตร ว่ามีความแตกต่างและมีการนำไปใช้งานอย่างไร สมการการสร้างภาพเชิงปริมาตร สมการการสร้างภาพเชิงปริมาตรแบบไม่ต่อเนื่อง และสุดท้ายจะได้กล่าวถึงวิธีการสร้างภาพเชิงปริมาตรทั้งในส่วนของวิธีการแบบลำดับภาพ และวิธีการแบบลำดับวัตถุ

2.2 ความรู้เบื้องต้นเกี่ยวกับภาพเชิงพื้นผิวและภาพเชิงปริมาตร

ในหัวข้อนี้ต้องการอธิบายถึงความแตกต่างระหว่างภาพแบบเชิงพื้นผิวและภาพเชิงปริมาตร แต่จะกล่าวถึงรายละเอียดเน้นไปที่ภาพเชิงปริมาตรเป็นส่วนใหญ่

2.2.1 ภาพเชิงพื้นผิว

ภาพที่เราเห็นในธรรมชาตินั้นจะเป็นภาพเชิงพื้นผิวเป็นส่วนใหญ่ เพราะเราสามารถมองเห็นเพียงแค่ภายนอกในส่วนที่เป็นพื้นผิว ไม่สามารถมองทะลุเข้าไปเห็นรายละเอียดภายในวัตถุนั้น ๆ ได้ การเกิดภาพเชิงพื้นผิวในธรรมชาติ จะเกิดขึ้นได้เมื่อแสงเดินทางไปกระทบวัตถุ แล้วสะท้อนกลับออกมา การสะท้อนกลับนี้จะขึ้นอยู่กับคุณสมบัติของพื้นผิววัตถุว่ามีการดูดกลืนแสงได้มากหรือน้อยเพียงใด และค่าของสเปกตรัมหรือค่าความถี่ของสีใดที่จะถูกสะท้อนออกมา ทำให้เรามองเห็นวัตถุมีสีต่าง ๆ

เมื่อเราทำการสร้างภาพเชิงพื้นผิวในระบบคอมพิวเตอร์กราฟิก จะมีวิธีการในการสร้างอยู่หลายรูปแบบทั้งวิธีการฉายภาพตามทิศทางการมอง วิธีการของมาร์ชชิงคิวบ์ (Marching Cubes Algorithm) หรือวิธีการอื่น ๆ แต่ในที่นี้จะไม่กล่าวถึงรายละเอียดของวิธีการต่าง ๆ ของการสร้างภาพเชิงพื้นผิว

การนำเอาภาพเชิงพื้นผิวไปใช้งานหรือประโยชน์ของภาพเชิงพื้นผิวนั้น จะเป็นงานในลักษณะของคอมพิวเตอร์กราฟิกทั่วไป เช่น การทำ 3D Modeling (วิทวัส วิทยชำนาญกุล, 2545) เกมสามมิติต่างๆ ซึ่งเกิดมาจากการสร้างโพลิกอนรูปสามเหลี่ยมหลาย ๆ รูปมาประกอบกัน พร้อมด้วยการทำ Texture mapping และการให้แสงเงา ส่วนงานทางด้านการแพทย์นั้นก็สามารถนำเอาภาพเชิงพื้นผิวไปช่วยในการวิเคราะห์หารอยร้าวของกระดูกได้ เป็นต้น

2.2.2 ภาพเชิงปริมาตร

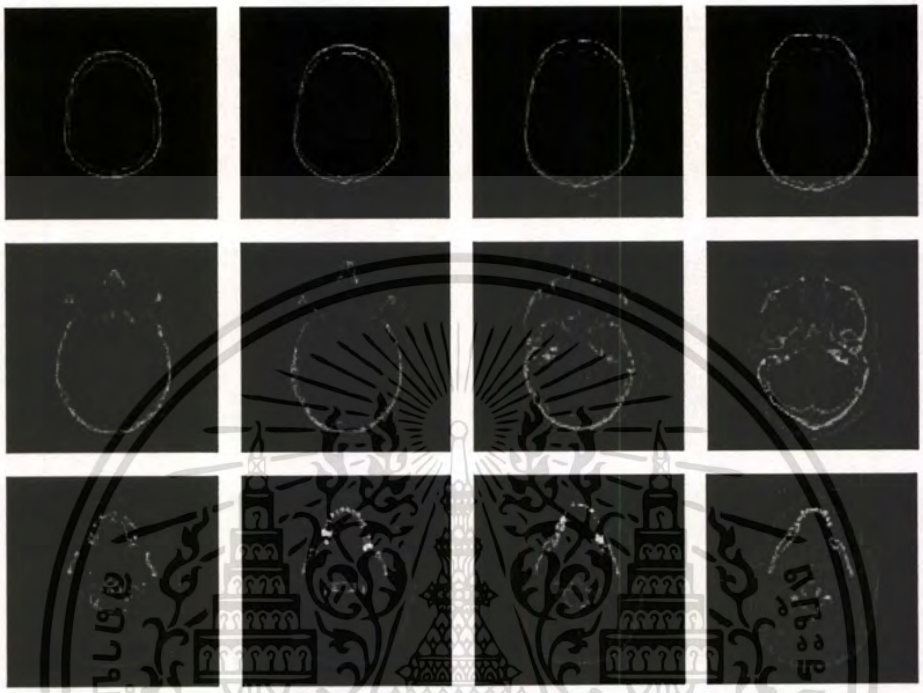
ภาพเชิงปริมาตรเป็นภาพที่สามารถมองเห็นรายละเอียดได้มากกว่ารายละเอียดที่ปรากฏอยู่บนพื้นผิวของวัตถุ เพราะสามารถมองทะลุเข้าไปเห็นรายละเอียดถึงภายในเนื้อของวัตถุได้ ในธรรมชาตินั้นภาพเชิงปริมาตรจะเกิดขึ้นในวัตถุกึ่งโปร่งแสง หรือวัตถุโปร่งใสต่าง ๆ ซึ่งวัตถุประเภทนี้จะมีคุณสมบัติยอมให้แสงเดินทางผ่านเข้าในเนื้อของวัตถุหรือทะลุออกไปในอีกทิศทางหนึ่งได้ ทำให้ภาพที่ปรากฏแก่ตาเป็นภาพที่มีความลึกของวัตถุมากกว่าการสะท้อนของแสงและเงาที่ผิวของวัตถุเพียงอย่างเดียว ตัวอย่างที่สามารถเห็นได้ทั่วไป เช่น เราสามารถมองเห็นภายในเนื้อเยื่อบางส่วนของแมงกะพรุน หรือปลาบางชนิด ที่มีความโปร่งแสงได้ เป็นต้น ซึ่งการที่จะสามารถมองเห็นรายละเอียดได้มากหรือน้อยจะขึ้นอยู่กับสี และความทึบแสง ซึ่งเป็นคุณสมบัติเฉพาะตัวของแต่ละวัตถุว่าจะยอมให้แสงผ่านเข้าไปได้มากน้อยแค่ไหน และสะท้อนแสงออกมาในความถี่ของสีใด

ภาพคอมพิวเตอร์กราฟิกที่เป็นภาพเชิงปริมาตรนั้นก็เช่นกันกับที่อธิบายมาข้างต้น คือ เป็นภาพที่เกิดจากความพยายามในการเลียนแบบภาพที่เกิดขึ้นในธรรมชาติเหล่านี้กับวัตถุชนิดต่าง ๆ เช่น การสร้างภาพเชิงปริมาตร์ภายในร่างกายของมนุษย์ หรือ การสร้างภาพเชิงปริมาตรของเครื่องจักร ซึ่งจะทำให้เห็นส่วนประกอบต่าง ๆ ของร่างกายหรือเครื่องยนต์โดยที่ไม่ต้องผ่าตัด หรือแยกชิ้นส่วนออกมาจริง

แต่เนื่องจากวัตถุเหล่านี้ในสภาพแวดล้อมนั้นไม่ได้โปร่งแสง หรือยอมให้แสงสีขาวโดยทั่ว ๆ ไปผ่านเข้าไปภายในแล้วสะท้อนมาเข้าตาจนเกิดเป็นภาพได้ ดังนั้นจึงต้องมีเครื่องมือที่ช่วยทำให้สามารถแสดงข้อมูลภายในเหล่านี้ออกมา ตัวอย่างเครื่องมือเหล่านี้ เช่น การถ่าย X-Ray, เครื่องถ่าย CT (Computer Tomography), เครื่องถ่ายแบบ MRI (Magnetic Resonance Imaging), Ultra Sound, PET และอื่น ๆ ซึ่งเครื่องมือแต่ละชนิดก็จะมีคุณสมบัติที่แตกต่างกันออกไป เช่นเครื่องถ่าย CT จะให้ภาพในส่วนที่เป็นกระดูกชัดเจน แต่เครื่องถ่าย MRI จะให้ภาพในส่วนของเนื้อเยื่อชัดเจนกว่า เป็นต้น ซึ่งในงานวิจัยนี้ได้ใช้ภาพจากเครื่องถ่ายแบบ CT เป็นภาพเริ่มต้นของการสร้างภาพเชิงปริมาตรของร่างกายมนุษย์

ภาพที่ได้จากเครื่องถ่าย CT นั้นจะเป็นภาพตัดขวาง มีงานวิจัยหลาย ๆ งานวิจัยที่ได้กล่าวถึงวิธีการสร้างภาพตัดขวางเหล่านี้ เช่น การปรับปรุงวิธีการสร้างภาพกลับโดยใช้วิธีการทางพีชคณิต (พิทยา อึงพินิจพงศ์, 2545) แต่ในวิทยานิพนธ์ฉบับนี้ได้เน้นที่การนำข้อมูลเหล่านี้มาใช้ในการสร้างภาพเชิงปริมาตรมากกว่า เมื่อจะทำการสร้างภาพเชิงปริมาตรบนคอมพิวเตอร์จะต้องเตรียมข้อมูลภาพให้เหมาะสมแก่การนำไปคำนวณก่อน โดยจะนำข้อมูลจุดภาพ (Pixel) ไปสร้างเป็นปริมาตรเริ่มต้นเก็บไว้เป็นชุดข้อมูล Array แบบ 3 มิติ หรือรูปแบบอื่น ๆ ที่เหมาะสม เมื่อจัดรูปแบบข้อมูลเหล่านี้เสร็จแล้ว

สมาชิกหรือหนึ่งจุดข้อมูลเหล่านี้จะเรียกว่า วีอกเซล (Voxel) หลังจากนั้นจึงจะนำไปสร้างเป็นภาพเชิงปริมาตรตามขั้นตอน หรือวิธีการต่าง ๆ ต่อไป



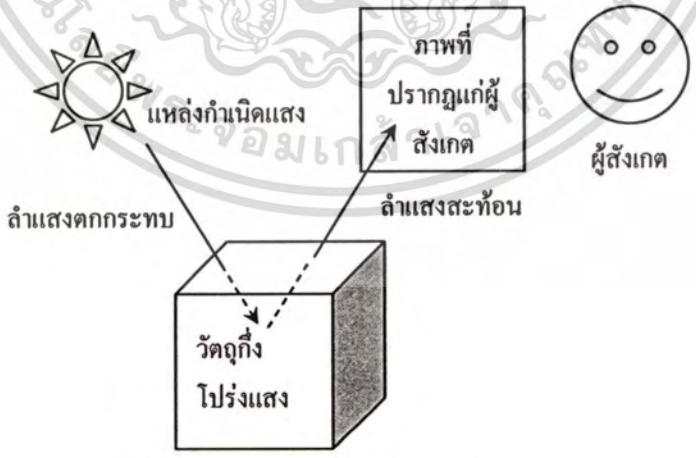
รูปที่ 2.1 ภาพตัดขวางจากเครื่องถ่ายภาพแบบ CT

ในการสร้างภาพเชิงปริมาตรในงานคอมพิวเตอร์กราฟิกนั้น มีวัตถุประสงค์เพื่อการนำเสนอข้อมูลในอีกมุมมองหนึ่ง ซึ่งช่วยขยายขอบเขตการมองเห็นของมนุษย์เป็นสำคัญ ไม่ได้เน้นที่ความเหมือนจริงของภาพให้เหมือนกับภาพถ่ายมากนัก ดังนั้นภาพที่เกิดจากวิธีการนี้จึงสามารถแสดงออกมาได้หลายรูปแบบ หลายสีขึ้นอยู่กับผู้สร้างภาพนั้นขึ้นมา ว่าต้องการที่จะให้เห็นรายละเอียดหรือความแตกต่างของแต่ละเนื้อเยื่อหรืออวัยวะต่าง ๆ มากน้อยแค่ไหน ตัวอย่างภาพผลลัพธ์ของการสร้างภาพเชิงปริมาตร เช่น รูปที่ 2.2 เป็นต้น

รูปที่ 2.2 ตัวอย่างภาพผลลัพธ์จากการสร้างภาพเชิงปริมาตร

2.2.3 สมการการสร้างภาพเชิงปริมาตร

เมื่อแสงเดินทางเข้าไปในปริมาตรเช่น ในวัตถุโปร่งแสงนั้น แสงอาจเกิดการเลี้ยวเบน การหักเห การถูกดูดกลืน ซึ่งเป็นเรื่องคุณสมบัติของแสงและวัตถุทั่วไป แต่วัตถุประสงค์ของการสร้างภาพเชิงปริมาตรนั้นไม่ได้ต้องการจำลองเหตุการณ์ทางฟิสิกส์ของแสง แต่เป็นการที่จะนำเสนอข้อมูลในอีกรูปแบบหนึ่ง เพื่อเป็นการขยายขอบเขตการมองเห็นของมนุษย์ เท่านั้น ดังนั้นการสร้างภาพเชิงปริมาตรนี้จึงละเลยกฎเกณฑ์บางอย่างที่ไม่จำเป็นไป



รูปที่ 2.3 ภาพแสดงเหตุการณ์จำลองการเกิดภาพในธรรมชาติ

พฤติกรรมต่าง ๆ ของแสงได้ถูกอธิบายไว้ในสมการสมดุลพลังงาน (Energy balance equation) ในส่วนของการส่องสว่าง (Radiance) ใช้สัญลักษณ์แทนคือ $L(r,\varpi)$ ซึ่งสมการนี้จะกล่าวถึงความหนาแน่นของพลังงาน (W-Watt) ต่อพื้นที่ (m^2-ตารางเมตร) ต่อมุมแบบกรวย (sr-Steradian) ในทิศทางการเดินของแสง โดยจะมีหน่วยเป็น W/m^2-sr และประกอบด้วยองค์ประกอบสามส่วนดังสมการ

$$\varpi \cdot \nabla L(r,\varpi) = -\phi_t(r)L(r,\varpi) + \varepsilon(r,\varpi) + \int_{S^2} k(r,\varpi' \rightarrow \varpi)L(r,\varpi')d\varpi' \qquad (2.1)$$

โดยที่

$\phi_t(r)$ คือ สัมประสิทธิ์การสูญเสียต่อระยะทางของอนุภาคโฟตอน ทั้งในส่วนการดูดกลืน (Absorbed) และการกระจาย (Scattered) มีหน่วยเป็น m^{-1}

$\varepsilon(r,\varpi)$ คือ ฟังก์ชั่นการแพร่ (Emission function) ของอนุภาคโฟตอนภายในเนื้อของวัตถุ มีหน่วยเป็น W/m^3-sr

$k(r,\varpi' \rightarrow \varpi)$ คือ เคอร์เนลของการกระจาย (Scattering kernel) มีหน่วยเป็น $sr^{-1}m^{-1}$

ϖ' คือ ลำแสงอื่น ๆ ที่กระจายเข้ามาในแนวทางเดินของ ϖ โดยจะทำการอินทิเกรตรวมจากทุก ๆ ทิศทางของแสง โดยมีขนาดของพื้นที่เท่ากับ S^2

สมการที่ 2.1 เป็นสมการเชิงอนุพันธ์อันดับหนึ่ง ซึ่งเป็นรูปแบบเชิงอนุพันธ์ของสมการถ่ายโอน (Differential form of the equation of transfer) ซึ่งสามารถเขียนในรูปแบบของการอินทิเกรตได้เป็น

$$L(r,\varpi) = e^{-\tau(r,r_B)}L_B(r_B,\varpi) + \int_{(r,r_B)} e^{-\tau(r,r')}Q(r',\varpi)dr' \qquad (2.2)$$

โดยที่

$\tau(r,s)$ คือ สัมประสิทธิ์การสูญเสียในรูปของการอินทิเกรต โดยมีระยะทางตั้งแต่ r ถึง s

$\tau(r,s) \equiv \int_{(r,s)} \phi_t(r')dr'$

ระยะทางระหว่างจุดสองจุดแทนด้วย $\Gamma(r,s)$ และกำหนดให้ $e^{-\tau(r,r_B)}$ เป็นตัวประกอบการอินทิเกรตที่แปลงสมการ 2.1 เป็นสมการที่ 2.2

$L_B(r,\varpi)$ คือฟังก์ชั่นที่กำหนดขอบเขตเหนือพื้นที่ผิวรอบ ๆ ปริมาตร โดยที่ r_B คือจุดตัดของพื้นผิวและทางเดินแสง จาก r ในทิศทาง ϖ

$Q(r',\varpi)$ คือ รูปย่อของพจน์การแพร่และการกระจาย

$Q(r,\varpi) \equiv \varepsilon(r,\varpi) + \int_{S^2} k(r,\varpi' \rightarrow \varpi)L(r,\varpi')d\varpi'$

สมการที่ 2.2 เป็นสมการที่ถูกใช้งานอย่างแพร่หลายในการอธิบายคุณสมบัติของแสงต่อการเกิดภาพ หรือการนำไปใช้ในการสร้างภาพ แต่จะเห็นได้ว่าสมการนี้ยังไม่สะดวกในการนำไปใช้นักเพราะยังมีความยุ่งยากอยู่มาก ดังนั้นจึงมีการปรับปรุงสมการนี้ใหม่ โดยการตั้งสมมติฐานเพิ่มเติมอีกสี่ข้อ โดยข้อแรกคือ พิจารณาแสงว่าเป็นเพียงลำแสงเดี่ยว ข้อที่สองคือ ไม่พิจารณาการดูดกลืนและการกระจายในระหว่างแหล่งกำเนิดและวัตถุ ข้อสามตั้งข้อสันนิษฐานว่ามีการดูดกลืนแบบไอโซโทรปิค (Isotropic absorption) และข้อสุดท้ายคือ ลำแสงที่เดินทางเข้ามาในวัตถุมาจากแหล่งกำเนิดแสงระยะอนันต์

จากสมมติฐานทั้งสี่ข้อที่กล่าวมาทำให้สามารถลดพจน์ของสมการ 2.2 เหลือเพียงสมการที่ 2.3 ซึ่งเป็นสมการทั่วไปของการสร้างภาพเชิงปริมาตร (Volume rendering equation)

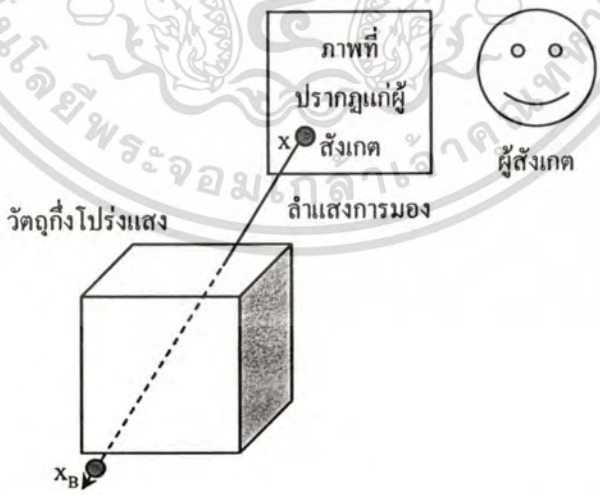
$$L(x) = \int_{x}^{x_B} e^{-\int_{x}^{x'} \phi_t(x'')dx''} \varepsilon(x')dx' \qquad (2.3)$$

โดยที่

$L(x)$ เป็นความเข้มแสงที่ระนาบการมอง

$\varepsilon(x')$ เป็นคุณสมบัติเฉพาะของจุดแสงในแนวลำแสง

$\phi_t(x'')$ เป็นสัมประสิทธิ์การสูญเสียต่อระยะทางของอนุภาคโฟตอน ซึ่งบอกถึงความทึบแสงของวัตถุโดยจะเป็นอัตราส่วนของการลดลงของความสว่างของแสงต่อหนึ่งหน่วยความยาวเนื่องจากการถูกดูดกลืนและการกระจายแสง



รูปที่ 2.4 สถานการณ์จำลองตามสมการการสร้างภาพเชิงปริมาตร

การสร้างภาพเชิงปริมาตรแบบการฉายแสงธรรมดาทั่วไปจะมีพื้นฐานมาจากการใช้สมการการสร้างภาพเชิงปริมาตรนี้ โดยจะทำการฉายทีละลำแสงเข้าไปในปริมาตร แล้วทำการหาค่าผลลัพธ์ในแต่ละแนวลำแสง โดยจะกระทำขนานกันไปเรื่อย ๆ แสดงได้ดังรูปด้านบน

จากสมการ 2.3 นั้นจะเห็นได้ว่าเราสามารถนำเอาสมการการส่องสว่าง (สมการที่ 2.2) มาใช้ได้ง่ายยิ่งขึ้นในรูปแบบของสมการการสร้างภาพเชิงปริมาตร แต่ว่าสมการนี้ยังอยู่ในรูปของการอินทิเกรต ทำให้เมื่อทำการสร้างภาพเชิงปริมาตรด้วยคอมพิวเตอร์ยังเกิดความยุ่งยากอยู่ ดังนั้นจึงต้องมีการประมาณสมการนี้ใหม่อีกครั้งเป็นแบบฟังก์ชั่นไม่ต่อเนื่อง ซึ่งจะได้กล่าวในหัวข้อถัดไป

2.2.4 สมการการสร้างภาพเชิงปริมาตรแบบไม่ต่อเนื่อง

ในหัวข้อที่ผ่านมาได้กล่าวถึงสมการการสมดุลพลังงานไปแล้ว หลังจากนั้นได้ลดรูปของสมการโดยการละเลยกฎเกณฑ์บางอย่างทางฟิสิกส์ไป เพื่อให้สมการมีความง่ายในการนำไปใช้งานในการคำนวณมากขึ้น ทำให้เราได้สมการการสร้างภาพเชิงปริมาตรในสมการที่ 2.3 แต่ว่าสมการนี้ก็ยังนำไปใช้ลำบากหากเป็นการคำนวณบนคอมพิวเตอร์ ดังนั้นจึงต้องทำการประมาณฟังก์ชั่นการอินทิเกรตใหม่ โดยการใช้หลักการทางการคำนวณเชิงเลข เพราะการอินทิเกรทในคอมพิวเตอร์นั้นต้องเป็นลักษณะของการอินทิเกรทแบบไม่ต่อเนื่อง (Discrete Integration) การประมาณค่าของการอินทิเกรทแบบต่อเนื่องในรูปแบบของการอินทิเกรทแบบไม่ต่อเนื่องสามารถทำได้โดยใช้ Riemann sum ทำให้ได้สมการการประกอบภาพเชิงปริมาตร (Volumetric compositing equation) ออกมาดังสมการที่ 2.4

$$\begin{aligned} L(x) &= \sum_{i=0}^{n-1} e^{-\sum_{j=0}^{i-1}\phi_j \Delta x} \cdot \varepsilon_i \Delta x \\ &= \sum_{i=0}^{n-1} \varepsilon_i \Delta x \cdot \prod_{j=0}^{i-1} e^{-\phi_j \Delta x} \end{aligned} \tag{2.4}$$

โดยที่ $\varepsilon_i \equiv \varepsilon(x + i\Delta x)$
$\phi_i \equiv \phi_i(x + i\Delta x)$

จากสมการที่ 2.4 มีการนิยามเพิ่มเติมดังนี้

$\alpha_i \equiv 1 - e^{-\phi_i \Delta x}$	เป็นค่าความทึบแสงของจุดแซมเปิล i
$C_i \equiv (\varepsilon_i / \alpha_i) \cdot \Delta x$	เป็นค่าสีของจุดแซมเปิล i
$c_i \equiv \alpha_i C_i$	ค่าการคูณของค่าสีและความทึบแสง

ค่าการคูณกันล่วงหน้าของค่าสีและความทึบแสงนี้ มีประโยชน์มากในการที่จะรวมแสงแบบ Digital Compositing (Porter and Duff, 1984) เพราะเราสามารถที่จะใช้ตัวดำเนินการติดตามลำแสงสะสมหรือ Over operator ได้ ทำให้การใช้เทคนิคการสิ้นสุดลำแสงก่อน (Early ray termination) ถูกนำมาใช้ได้

$$\begin{aligned} L(x) &= \sum_{i=0}^{n-1} c_i \cdot \prod_{j=0}^{i-1} (1-\alpha_j) \\ &= c_0 + c_1(1-\alpha_0) + c_2(1-\alpha_0)(1-\alpha_1) + \ldots \\ &\quad + c_{n-1}(1-\alpha_0)\ldots(1-\alpha_{n-2}) \\ &= c_0 over \cdot c_1 over \cdot c_2 over \cdots over \cdot c_{n-1} \end{aligned} \tag{2.5}$$

ค่าสีหรือ C_i ในสมการที่ 2.5 นั้น อาจจะกำหนดได้โดยการพิจารณาค่าความทึบแสงเป็นสัดส่วนลดหลั่นกันไป เพราะว่าไม่มีกฎเกณฑ์ทางฟิสิกส์ที่เคร่งครัดในเรื่องเกี่ยวกับการกำหนดค่าสี ตัวอย่างเช่น ถ้าค่าความทึบแสงมากก็อาจให้มีค่าสีมีความเข้มมาก ถ้าหากความทึบแสงน้อยค่าความเข้มของสีก็จะน้อยตามสัดส่วนที่ได้กำหนดไว้ เป็นต้น

จากที่กล่าวมาทั้งหมดในหัวข้อนี้นั้น สามารถสรุปขั้นตอนการสร้างภาพเชิงปริมาตรโดยการใช้สมการการประกอบภาพเชิงปริมาตร หรือสมการที่ 2.5 โดยใช้วิธีการ Ray Casting สามารถสรุปขั้นตอนทั้งหมดได้ดังนี้

1. สำหรับแต่ละจุดภาพ (pixel) ในภาพผลลัพธ์จะทำการฉายแสงเข้าไปในปริมาตรเพื่อสร้างภาพในจุดนั้นๆ
2. ทำการคำนวณค่าความทึบแสง α_i และค่าสี C_i ของแต่ละจุด (voxel) ที่ลำแสงวิ่งผ่าน
3. รวมค่าสีและความทึบแสงเหล่านี้ตามวิธีของสมการที่ 2.5 จะได้จุดภาพ (pixel) ที่ภาพผลลัพธ์หนึ่งจุด
4. กระทำจนครบทุก ๆ จุดภาพ (pixel) ก็จะได้ภาพผลลัพธ์ที่สมบูรณ์

2.3 วิธีการสร้างภาพเชิงปริมาตร

ในหัวข้อนี้จะเป็นการกล่าวถึงขั้นตอนและวิธีการสร้างภาพเชิงปริมาตรแบบต่าง ๆ โดยจะแบ่งแยกออกเป็นสองประเภทชัดเจนคือ วิธีการแบบลำดับภาพและ วิธีการแบบลำดับวัตถุว่ามีลำดับและขั้นตอนแตกต่างกันอย่างไรบ้าง

2.3.1 วิธีการแบบลำดับภาพ (Image Order Algorithm)

วิธีการสร้างภาพเชิงปริมาตรแบบลำดับภาพนั้น ในที่นี้จะยกตัวอย่างของวิธีการสร้างภาพเชิงปริมาตรแบบฉายแสง (Ray Casting Algorithm) ซึ่งเป็นวิธีที่รู้จักกันโดยทั่วไป และสามารถทำความเข้าใจได้ง่าย เพราะขั้นตอนการทำงานจะตรงไปตรงมา ตามสมการการประกอบภาพเชิงปริมาตร หรือสมการที่ 2.5 ที่ได้นำเสนอไปแล้วนั่นเอง ขั้นตอนการทำงานของวิธีการนี้จะทำการคำนวณโดยการฉายแสงทีละเส้นทะลุผ่านลึกเข้าไปในปริมาตร และทำการรวมค่าสีและค่าความทึบแสงตลอดเส้นทางการเดินแสงสำหรับการสร้างภาพหนึ่งจุดบนภาพผลลัพธ์ แล้วทำตามวิธีการเดิมซ้ำตลอดทั้งภาพผลลัพธ์ ดังวิธีการที่แสดงต่อไปนี้

```
1:  for y_i = 1 to ImageHeight                    } เป็นการพิจารณาที่ภาพผลลัพธ์
      for x_i = 1 to ImageWidth                   }
        for z_i = 1 to VolumeDept                 } คำนวณทีละแนวลำแสงลึกเข้าไปในปริมาตร
          for_each x_o in ResamplingFilter(x_i , y_i ,z_i)
5:          for_each y_o in ResamplingFilter(x_i , y_i ,z_i)   } ใช้ Filter แบบสามมิติ
              for_each z_o in ResamplingFilter(x_i , y_i ,z_i)
                add contribution of Voxel[x_o , y_o , z_o] to ImagePixel[x_i , y_i]
```

วงรอบการทำซ้ำด้านนอก 2 วงจะทำซ้ำทุก ๆ พิกเซลของภาพผลลัพธ์ที่ต้องการ ส่วนวงรอบต่อมาจะทำการคำนวณทุกจุดบนทิศทางของมุมมองที่แสงเดินทางผ่านภายในปริมาตรภาพ ส่วนสุดท้ายตั้งแต่บรรทัดที่ 4-6 เป็นสามวงรอบที่ทำการคำนวณกับค่าของวอกเซล (Voxel) ด้วยการใช้ตัวกรองสุ่มตัวอย่าง (Resampling filter) ค่าวอกเซลจะถูกคูณด้วยสัมประสิทธิ์การกรองแล้วทำการรวมไปเป็นภาพผลลัพธ์ในบรรทัดที่ 7

ข้อเสียของวิธีฉายแสง คือ วิธีการนี้มีค่าการสูญเสียจากการเข้าถึงหน่วยความจำสูงมาก (Higher memory overhead) เพราะเมื่อมีการเปลี่ยนมุมมองต้องมีการปรับตำแหน่งของปริมาตรใหม่ หรือไม่ก็ต้องมีเทคนิคหรือวิธีการอื่นที่จะมาช่วยให้ไม่ต้องปรับตำแหน่งของปริมาตรใหม่ทุกครั้ง และวิธีการนี้ไม่สามารถใช้เทคนิคการเข้าถึงข้อมูลตามลำดับในหน่วยความจำหลักหรือฮาร์ดิสก์ได้เมื่อจะทำการคำนวณจะต้องโหลดเอาข้อมูลทั้งหมดมาเก็บไว้ในหน่วยความจำก่อนเสมอ ซึ่งถ้าหากปริมาตรมีขนาดใหญ่ ยิ่งทำให้ต้องใช้หน่วยความความจำจำนวนมากขึ้นตามไปด้วย

2.3.2 วิธีการแบบลำดับวัตถุ (Object Order Algorithm)

วิธีการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุจะพิจารณาปริมาตร และภาพผลลัพธ์ด้วยวิธีการอีกรูปแบบหนึ่ง คือ จะคำนวณที่วอกเซลของปริมาตรก่อนแล้วค่อยฉายกลับไปยังภาพ ด้วยวิธีการหรือแนวคิดนี้ทำให้ขั้นตอนของการรีแซมปลิงลดความยุ่งยากลง และสามารถใช้วิธีการเข้าถึงข้อมูลในฮาร์ดดิสก์ตามลำดับของการจัดเก็บข้อมูลได้ เมื่อทำการคำนวณแผ่นสไลด์ใดเสร็จแล้วก็สามารถปล่อยให้พื้นที่หน่วยความจำสำรองหรือ RAM ว่างได้ หลังจากนั้นค่อยโหลดข้อมูลปริมาตรที่จะทำการคำนวณเข้ามาใหม่ ทำให้วิธีการนี้ช่วยประหยัดหน่วยความจำในระหว่างการคำนวณไปได้มาก ขั้นตอนและวิธีการสามารถแสดงได้ดังนี้

```
1:  for z_o = 1 to VolumeDept
        for y_i = 1 to ImageHeight      } เป็นการพิจารณาทีละ
            for x_i = 1 to ImageWidth   } แผ่นสไลซ์ (สองมิติ)
                for_each y_o in ResamplingFilter(x_i , y_i )   } เป็นการใช้ Filter แบบสองมิติ หรือ
5:                  for_each x_o in ResamplingFilter(x_i , y_i)  } Bi-linear Interpolation
                        add contribution of Voxel[x_o , y_o , z_o] to ImagePixel[x_i , y_i]
```

จากวิธีการที่แสดงนี้ จะเห็นได้ว่าวงรอบการทำงานลดลงไปหนึ่งรอบ ทำให้การคำนวณทำได้รวดเร็วขึ้น และมีคุณสมบัติหลาย ๆ อย่างที่ช่วยให้เวลาในการทำงานลดน้อยลง ซึ่งรายละเอียดเกี่ยวกับวิธีการนี้จะกล่าวถึงโดยละเอียดในบทที่ 4 ต่อไป

2.4 สรุป

ในบทนี้เป็นการกล่าวถึงภาพที่เราเห็นจริงในธรรมชาติ ทั้งในส่วนของภาพเชิงพื้นผิวและภาพเชิงปริมาตร โดยได้อธิบายถึงคุณสมบัติต่าง ๆ ของแสงต่อการเกิดภาพให้ปรากฏต่อสายตา ด้วยสมการการสมดุลพลังงานของแสงต่อหน่วยพื้นที่ (Energy balance equation) หลังจากนั้นจึงแสดงให้เห็นถึงการนำเอาสมการนี้มาใช้ในการสร้างภาพเชิงปริมาตรในระบบคอมพิวเตอร์กราฟิก ซึ่งต้องมีการคำนวณเป็นแบบไม่ต่อเนื่องหรือ Discrete computation ซึ่งสมการที่นำมาใช้ คือ สมการการประกอบภาพเชิงปริมาตร (Volumetric compositing equation)

หลังจากนั้นจึงเป็นการกล่าวนำให้รู้จักถึงขั้นตอนและวิธีการที่ใช้ในการสร้างภาพเชิงปริมาตรสองแบบคือ การสร้างภาพแบบลำดับภาพ และการสร้างภาพแบบลำดับวัตถุ ซึ่งแต่ละชนิดก็มีข้อดีแตกต่างกันออกไป ซึ่งรายละเอียดเหล่านี้จะได้กล่าวถึงในบทต่อ ๆ ไป

บทที่ 3
การสร้างภาพเชิงปริมาตรแบบลำดับภาพ

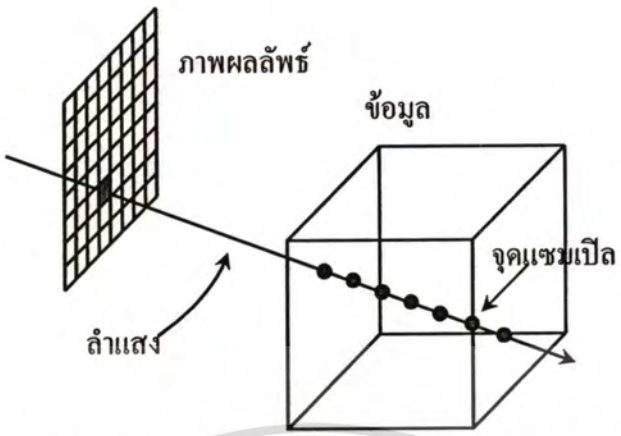
3.1 บทนำ

หลังจากที่ได้กล่าวถึงสมการการสร้างภาพเชิงปริมาตรไปในบทที่แล้วนั้น ในบทนี้จะเป็นการกล่าวถึงรายละเอียดของการสร้างภาพเชิงปริมาตร โดยใช้วิธีการฉายแสง ซึ่งเป็นวิธีการสร้างภาพเชิงปริมาตรแบบลำดับภาพ (Image Order Algorithm) ที่เป็นที่รู้จักกันดีและสามารถทำความเข้าใจได้ง่าย เพราะใช้หลักการของสมการสร้างภาพเชิงปริมาตรอย่างตรงไปตรงมา

ในบทนี้จะเริ่มจากการอธิบายคร่าว ๆ เกี่ยวกับวิธีการสร้างภาพแบบนี้ หลังจากนั้นจะเป็นการกล่าวถึงวิธีการในการประมาณค่าในช่วงหรือการอินเทอร์โพเลชั่น เพื่อหาค่าของจุดข้อมูลที่ทำการแซมปลิง แล้วตามด้วยการแบ่งกลุ่มของข้อมูล เพื่อจะได้กำหนดคุณสมบัติของข้อมูลแต่ละชนิดให้สามารถแสดงออกมาได้ชัดเจน และถูกต้องมากยิ่งขึ้น ลำดับถัดไปคือการกล่าวถึงการส่องสว่างและการให้แสงเงา และสุดท้ายคือ การอธิบายถึงวิธีการประกอบภาพ ดังจะได้กล่าวในรายละเอียดต่อไป

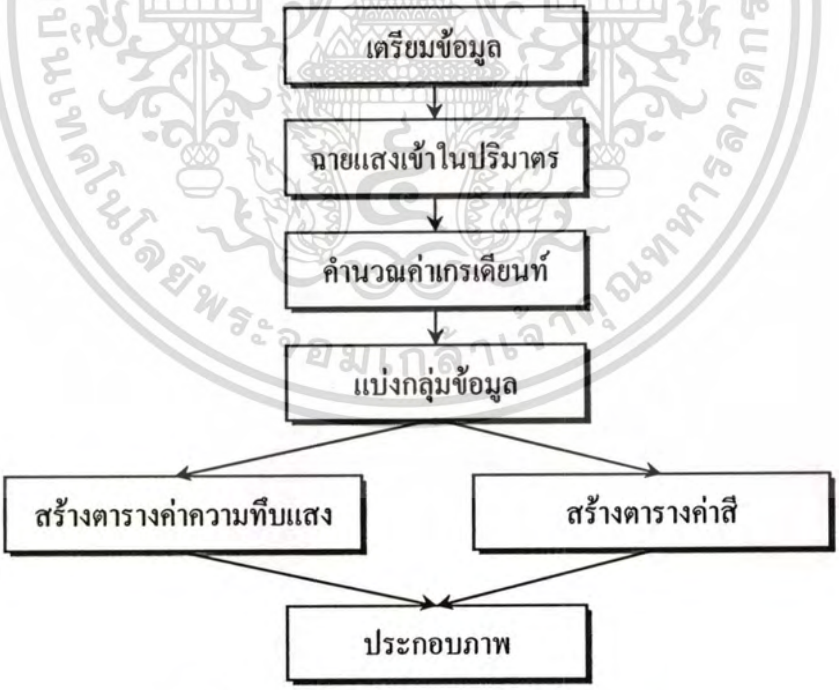
3.2 การสร้างภาพโดยใช้วิธีการฉายแสง (Ray Casting)

การสร้างภาพเชิงปริมาตรโดยวิธีการฉายแสงนี้เป็นอีกวิธีการหนึ่งที่ ต้องการแสดงข้อมูลเชิงปริมาตรให้อยู่ในรูปของข้อมูลภาพสองมิติ ในมุมมองต่าง ๆ ที่กระทำกับปริมาตร โดยที่จะทำการฉายแสงเข้าไปในปริมาตร แบ่งกลุ่มของข้อมูลออกเป็นชนิดต่าง ๆ เช่น เนื้อเยื่อ กระดูก ไขมัน แล้วคำนวณค่าสีและแสงเงาของข้อมูลแต่ละจุดแต่ละชนิด แล้วทำการรวมค่าเหล่านั้นเป็นจุดภาพผลลัพธ์หนึ่งจุด หลังจากนั้นจึงทำตามวิธีการตลอดทั้งภาพผลลัพธ์ ขั้นตอนการทำงานสามารถแสดงได้ดังนี้



รูปที่ 3.1 การสร้างภาพเชิงปริมาตรโดยวิธีการฉายแสง

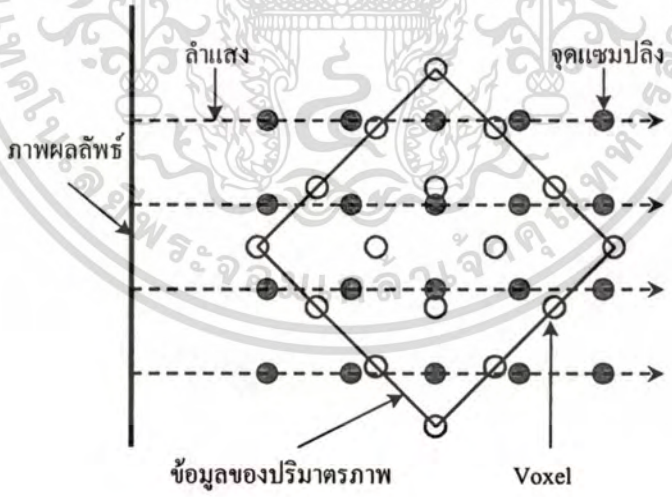
ขั้นตอนทั้งหมดของการสร้างภาพด้วยวิธีนี้นั้นจะแบ่งได้ทั้งหมดห้าขั้นตอน โดยจะประกอบด้วยการเตรียมข้อมูลภาพ การฉายแสงเข้าในปริมาตร การคำนวณค่าเกรเดียนท์ การแบ่งกลุ่มข้อมูล และการประกอบภาพดังแสดงในรูปที่ 3.2



รูปที่ 3.2 ขั้นตอนในการสร้างภาพเชิงปริมาตร

การเตรียมข้อมูลภาพจะถูกทำเพื่อนำภาพจากข้อมูลต้นฉบับมาจัดเรียงเป็นปริมาตร การฉายแสงจะถูกทำเพื่อหาตำแหน่งในปริมาตรข้อมูลภาพเพื่อทำการประมวลผลต่อไป การแบ่งกลุ่มข้อมูลจะทำเพื่อกำหนดค่าสีและความทึบแสงให้กับข้อมูล และการประกอบภาพจะทำเพื่อประกอบค่าสีและค่าความทึบแสงที่ได้เป็นจุดภาพในระนาบการมองเป็นขั้นตอนสุดท้าย

โดยภาพที่ปรากฏบนระนาบการมองจะประกอบไปด้วยจุดภาพ (Pixel) หลายจุด จุดภาพเหล่านี้จะถูกให้ค่าสีด้วยกระบวนการฉายแสง ซึ่งในกระบวนการนี้แสงจะถูกฉายจากจุดภาพในระนาบการมองไปสู่ปริมาตรข้อมูลภาพแล้วผ่านการประมวลผลเพื่อหาค่าสีมาแสดงในจุดภาพนั้น ๆ ดังแสดงในรูปที่ 3.1 แต่ก่อนที่แสงจะถูกฉายอาจต้องมีการหมุนปริมาตรข้อมูลภาพเพื่อให้เห็นปริมาตรวัตถุในมุมมองที่ต้องการ ปกติปริมาตรข้อมูลภาพมักมีขนาดใหญ่ซึ่งอาจประกอบไปด้วยข้อมูลภาพมากกว่าหนึ่งล้านข้อมูล การแปลงตำแหน่งของข้อมูลภาพโดยตรงอาจใช้เวลานานมาก เนื่องจากต้องทำการแปลงตำแหน่งของข้อมูลภาพทุกจุด วิธีการแปลงตำแหน่งแบบนี้เรียกว่าการแปลงตำแหน่งแบบไปหน้า (Forward Transformation) ในทางกลับกันแทนที่จะแปลงตำแหน่งของข้อมูลภาพเพื่อให้เห็นวัตถุในมุมมองที่ต้องการ สามารถแปลงตำแหน่งของระนาบการมองในทิศทางตรงข้ามกับการแปลงตำแหน่งของข้อมูลภาพโดยที่ตำแหน่งของข้อมูลภาพยังคงเดิมเรียกวิธีการแปลงแบบนี้ว่าการแปลงตำแหน่งแบบย้อนกลับ (Inverse Transformation) ซึ่งแสดงในรูปที่ 3.3



รูปที่ 3.3 การฉายแสงในสองมิติโดยแปลงตำแหน่งของระนาบการมอง

ตำแหน่งของจุดภาพในระนาบการมอง (x, y, z) จะถูกเปลี่ยนไปเป็นตำแหน่งที่แท้จริงของข้อมูลภาพในปริมาตร (x', y', z') โดยอาศัยเมตริกการแปลงผกผัน (Inverse Transform Matrix) ดังสมการ 3.1

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = M^{-1} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.1}$$

M^{-1} คือ เมตริกการแปลงผกผัน

จากรูปที่ 3.3 ตำแหน่งของจุดแสงที่ถูกเลือก (Sample Point) อาจไม่ตรงกับตำแหน่งของข้อมูลปริมาตร (Voxel) ที่มีอยู่จริง จึงทำให้จำเป็นต้องมีการคำนวณหาค่าจากข้อมูลที่อยู่รอบ ๆ จุดนั้น ซึ่งเรียกวิธีการนี้ว่า การอินเทอร์โพเลชั่น ซึ่งมีด้วยกันหลายแบบ และจะกล่าวถึงรายละเอียดในหัวข้อถัดไป

3.3 การอินเทอร์โพเลชั่น (Interpolation)

การกระทำทางคณิตศาสตร์เป็นสิ่งที่หลีกเลี่ยงไม่ได้ ในการสร้างภาพเชิงปริมาตร การนำเอาข้อมูลมาใช้ในการสร้างภาพนั้นในบางครั้ง อาจจะมีการเข้าถึงข้อมูลในตำแหน่งที่ข้อมูลไม่มีอยู่ดังรูปที่ 3.3 ดังนั้นจึงต้องมีการคำนวณค่าทดแทนข้อมูลจุดนั้นหรือการอินเทอร์โพเลชั่นเพื่อใช้ในการคำนวณต่อไป การอินเทอร์โพเลชั่นมีอยู่หลายวิธี ซึ่งแต่ละวิธีจะถูกควบคุมโดยเคอร์เนลของการอินเทอร์โพเลชั่น (Interpolation Kernel) และรูปร่างของเคอร์เนลจะเป็นตัวกำหนดค่าสัมประสิทธิ์สำหรับถ่วงน้ำหนักการบวกและคุณภาพของการอินเทอร์โพเลชั่น

การคำนวณค่าในตำแหน่งที่ต้องการอินเทอร์โพเลชั่นนั้นทำได้โดยนำเคอร์เนลการอินเทอร์โพเลชั่นไปคอนโวลูชันกันดังนี้

$$f_n(x) = \sum_{l=-\infty}^{\infty} g_s(l\Delta x) h_n(x - l\Delta x) \tag{3.2}$$

เมื่อ $f_n(x)$ คือผลลัพธ์ที่ได้จากการอินเทอร์โพเลชั่น และ h_n คือเคอร์เนลของการอินเทอร์โพเลชั่น ส่วนกรณี 2 มิตินั้นการอินเทอร์โพเลชั่นจะเป็นดังสมการที่ 3.3

$$f_n(x, y) = \sum_{l=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} g_s(l\Delta x, k\Delta y) h_n(x - l\Delta x, y - k\Delta y) \tag{3.3}$$

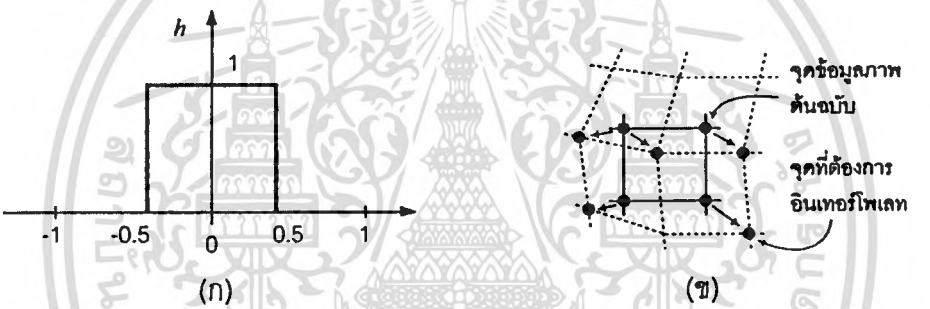
เคอร์เนลการอินเทอร์โพเลชั่นแบบ 1 มิติ สามารถขยายไปใช้ใน 2 และ 3 มิติโดยเริ่มจากการอินเทอร์โพเลชั่นแบบ 1 มิติในแนวแกน x, y และ z

3.3.1 การอินเทอร์โพเลชั่นแบบเนียเรสเนเบอร์ (Nearest Neighbor)

การอินเทอร์โพเลชั่นด้วยวิธีการนี้เป็นวิธีการที่ง่ายที่สุด แต่ให้ผลลัพธ์ที่มีค่าหยาบมาก ลักษณะโดยทั่วไปนั้นจะเป็นการนำค่าที่อยู่ใกล้จุดที่ต้องการอินเทอร์โพเลชั่นมากที่สุดมาเป็นคำตอบ ซึ่งสมการการอินเทอร์โพเลชั่นในหนึ่งและสองมิติ แสดงด้วยสมการ (3.4) และ (3.5) ตามลำดับ รูปร่างของเคอร์เนลสำหรับการอินเทอร์โพเลชั่นชนิดนี้แสดงดังรูปที่ 3.4 จะเห็นได้ว่ามีเพียงค่าเดียวเท่านั้นที่ถูกประมวลผล ดังนั้นการคำนวณในแต่ละวงรอบจึงมีน้อยมาก

$$f(x) = g(round(x)) \tag{3.4}$$

$$f(x,y) = g(round(x), round(y)) \tag{3.5}$$



รูปที่ 3.4 (ก) เคอร์เนลแบบเนียเรสเนเบอร์ (ข) การอินเทอร์โพเลชั่นด้วยวิธีเนียเรสเนเบอร์

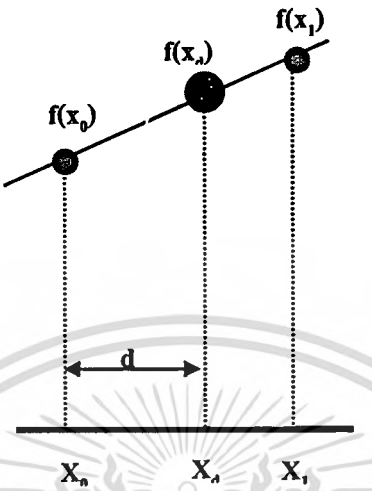
ข้อเสียของการอินเทอร์โพเลชั่นแบบนี้คือผลกระทบจากเอเลียสซิง (Aliasing) และข้อเสียอีกอย่างหนึ่งของการอินเทอร์โพเลชั่นแบบนี้คือการเลื่อนของภาพ ซึ่งการปัดค่าของเนียเรสเนเบอร์เป็นสาเหตุให้ภาพเลื่อนขึ้นไป ½ ถึง 1 พิกเซล ซึ่งมีผลอย่างมากในกรณีที่มีการประมวลผลภาพที่เกี่ยวข้องกับตำแหน่งของวัตถุภายในภาพ แต่ข้อดีของการอินเทอร์โพเลชั่นด้วยวิธีการนี้คือหน่วยประมวลผลไม่ต้องคำนวณค่าต่าง ๆ มาก

3.3.2 การอินเทอร์โพเลชั่นแบบเชิงเส้น (Linear Interpolation)

การอินเทอร์โพเลชั่นแบบเชิงเส้นนี้อาศัยสมการเชิงเส้นมาทำการคำนวณค่าที่ต้องการอินเทอร์โพเลชั่น แสดงดังสมการ 3.6

$$y = mx + c \tag{3.6}$$

สมมติว่าจุดสองจุด $f(x_0)$ และ $f(x_1)$ มีความสัมพันธ์แบบเชิงเส้น ดังแสดงในรูปที่ 3.4



รูปที่ 3.5 ความสัมพันธ์แบบเชิงเส้นของข้อมูล

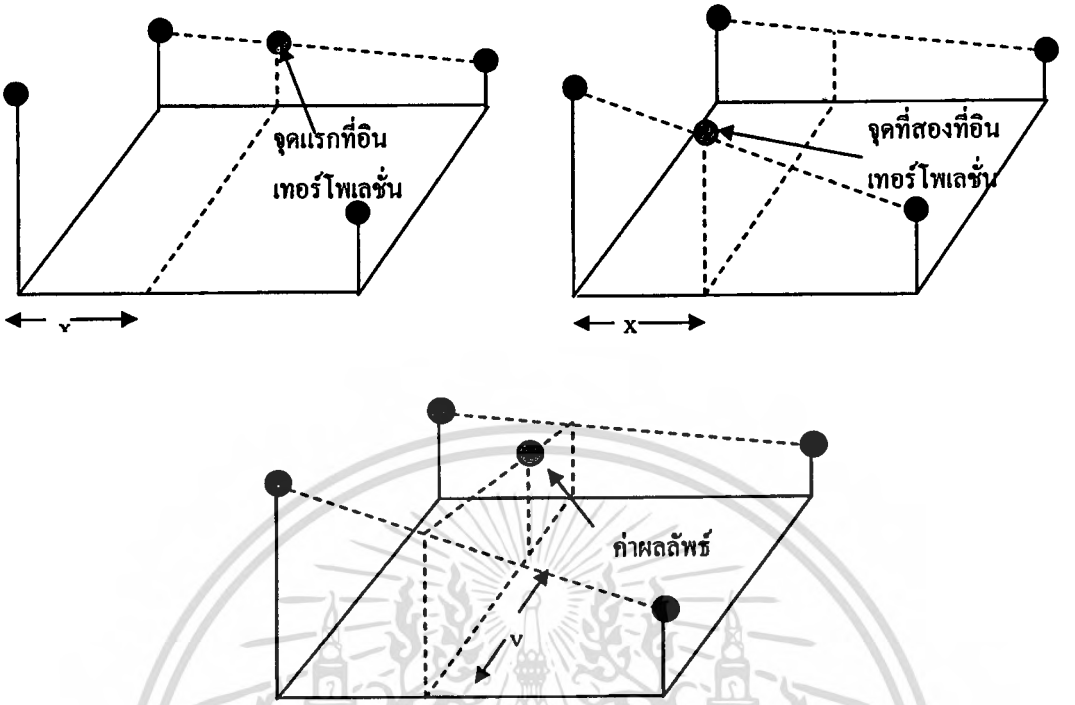
จากรูปที่ 3.5 สามารถเขียนสมการความสัมพันธ์แบบเชิงเส้นของ ข้อมูล $f(x_0)$ และ $f(x_1)$ ได้ดังสมการ 3.7

$$f(x_1)=\left(\frac{f(x_1)-f(x_0)}{x_1-x_0}\right)x_1+f(x_0) \tag{3.7}$$

ดังนั้นค่าของข้อมูลที่ตำแหน่ง x_d สามารถหาได้จากสมการ 3.8

$$f(x_d)=\left(\frac{f(x_1)-f(x_0)}{x_1-x_0}\right)(x_d-x_0)+f(x_0) \tag{3.8}$$

สำหรับข้อมูลในสองมิติ การอินเทอร์โพเลชั่นจะต้องทำสามครั้งคือแกน x สองครั้งและในแนวแกน y หนึ่งครั้ง ซึ่งเรียกการอินเทอร์โพเลชั่นแบบนี้ว่าไบลิเนียร์อินเทอร์โพเลชั่น (Bi-linear interpolation) ดังแสดงในรูปที่ 3.6



รูปที่ 3.6 การอินเทอร์โพเลชั่นในสามมิติ

(ก) ค่าที่ถูกอินเทอร์โพเลชั่นในแนวแกน x เป็นจุดแรก

(ข) ค่าที่ถูกอินเทอร์โพเลชั่นในแนวแกน x เป็นจุดที่สอง

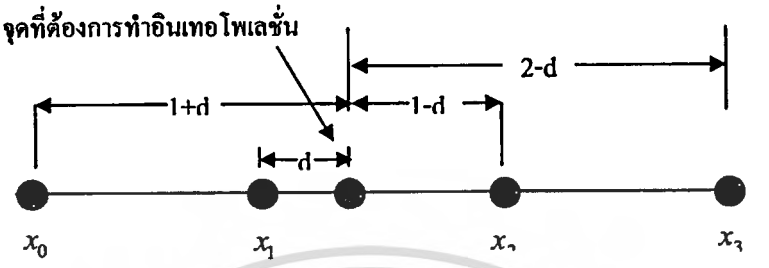
(ค) ค่าที่ถูกอินเทอร์โพเลชั่นในแนวแกน y

3.3.3 การอินเทอร์โพเลชั่นแบบคิวบิคคอนโวลูชั่น (Cubic Convolution)

การอินเทอร์โพเลชั่นด้วยคิวบิกคอนโวลูชั่นจะให้คุณภาพของภาพที่ดีกว่าการอินเทอร์โพเลชั่นแบบเชิงเส้น โดยลักษณะของภาพที่ได้จะมีความคมชัดมากกว่า ถ้าพิจารณาในหนึ่งมิติคิวบิกคอนโวลูชั่นจะใช้ค่าสี่ค่าในการคำนวณแต่ในการอินเทอร์โพเลชั่นแบบเชิงเส้นจะใช้เพียงสองค่า ข้อกำหนดของคิวบิกคอนโวลูชั่นแสดงดังสมการ 3.9

$$f(x)=\begin{cases}(a+2)|x|^3-(a+3)|x|^2+1 & 0\le|x|\le 1\\ a|x|^3-5a|x|^2+8a|x|-4a & 1\le|x|\le 2\\ 0 & 2\le|x|\end{cases} \tag{3.9}$$

ปกติค่า a จะอยู่ในช่วง –3 ถึง 0 ถ้าค่า a เข้าใกล้ 0 จะทำให้ได้ภาพที่มีความเบลอมากขึ้น ถ้า a มีค่าเข้าใกล้ –3 จะทำให้ได้รูปที่มีลักษณะคมขึ้น ในกรณีทั่วๆไปใช้ค่า a เท่ากับ -0.5



รูปที่ 3.7 ระยะทางระหว่างจุดสี่จุดที่ใช้ในการอินเทอร์โพเลชั่น

จากรูปที่ 3.7 เป็นระยะทางระหว่างจุดสี่จุดที่ใกล้ที่สุดซึ่งใช้ในการอินเทอร์โพเลชั่น ถ้าหากรู้ระยะทางนี้จะทำให้การคำนวณง่ายลง กำหนดให้สมการ 3.10 เป็นสมการของคิวบิกคอนโวลูชั่น

$$f(d) = c_0 p_0 + c_1 p_1 + c_2 p_2 + c_3 p_3 \tag{3.10}$$

ค่าของจุดภาพที่ตำแหน่ง x_0, x_1, x_2 และ x_3 คือ p_0, p_1, p_2 และ p_3 ตามลำดับ ถ้าหากทราบสมการของคอนโวลูชันฟังก์ชันก็จะสามารถหาค่าสัมประสิทธิ์ของสมการคอนโวลูชัน c_0, c_1, c_2 และ c_3 ได้ดังนี้

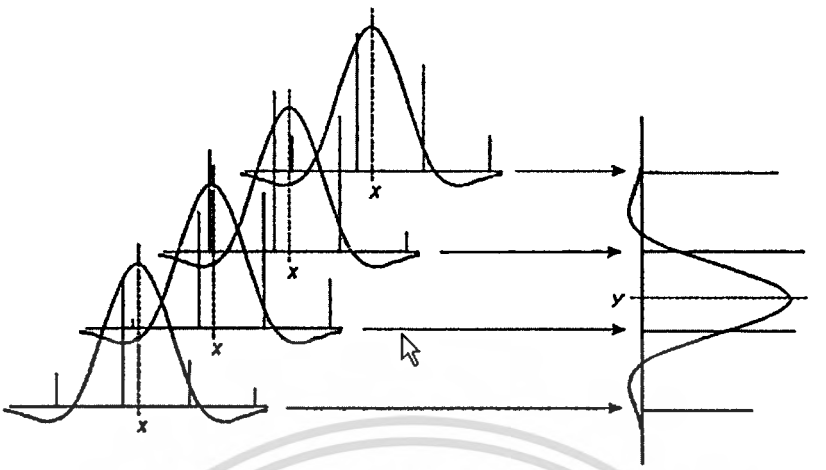
$$c_0 = a(1+d)^3 - 5a(1+d)^2 + 8a(1+d) - 4a = ad^3 - 2ad^2 + ad \tag{3.11}$$

$$c_1 = (a+2)d^3 - (a+3)d^2 + 1 \tag{3.12}$$

$$c_2 = -(a+2)d^3 + (2a+3)d^2 - ad \tag{3.13}$$

$$c_3 = -ad^3 + ad^2 \tag{3.14}$$

คิวบิกคอนโวลูชั่นในหนึ่งมิติสามารถขยายเป็นสองและสามมิติได้เช่นเดียวกับอินเทอร์โพเลชั่นแบบเชิงเส้นดังรูปที่ 3.8



รูปที่ 3.8 คิวบิคคอนโวลูชั่นในสองมิติ

3.4 การแบ่งกลุ่มข้อมูล

การแบ่งกลุ่มข้อมูลจะถูกทำเพื่อให้เราสามารถแยกแยะความแตกต่างภายในวัตถุได้อย่างชัดเจนมากขึ้น เช่น การสร้างภาพเชิงปริมาตรของร่างกายมนุษย์ การแบ่งกลุ่มข้อมูลในการสร้างภาพเชิงปริมาตรจะทำโดยการกำหนดค่าความทึบแสง (Opacity) ให้กับแต่ละจุดภาพ (Voxel) ในปริมาตรของข้อมูล โดยค่าความทึบแสงจะบอกถึงความทึบหรือความสามารถในการดูดกลืนแสงของจุดภาพ ความทึบแสงในการสร้างภาพเชิงปริมาตรจะมีค่าอยู่ในช่วง 0 ถึง 1 จุดภาพที่มีค่าเข้าใกล้ 1 จะเป็นจุดภาพที่มีลักษณะทึบแสง ส่วนจุดภาพที่มีค่าความทึบแสงใกล้ 0 จะเป็นจุดภาพที่มีความโปร่งแสงมาก โครงสร้างที่ต้องการเห็นจะถูกกำหนดให้มีค่าความทึบแสงมาก ส่วนโครงสร้างที่มีความสำคัญน้อยหรือไม่ต้องการเห็นจะถูกกำหนดความทึบแสงให้มีค่าน้อย การดูว่าข้อมูลที่นำมาสร้างภาพมีวัตถุอยู่ในภาพที่ความสว่างใดบ้างสามารถดูได้จากฮีสโตแกรมของภาพภาคตัดขวางของชุดข้อมูลนั้นๆ

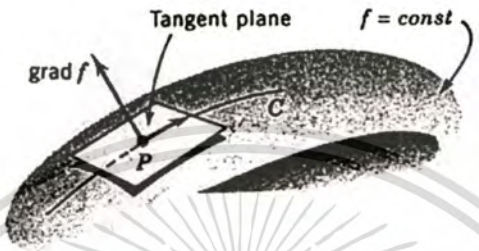
3.4.1 เกรเดียนท์ (Gradient)

ในคอมพิวเตอร์กราฟิก การแบ่งกลุ่มข้อมูลและการให้แสงเงาจำเป็นที่จะต้องรู้เกรเดียนท์และค่าเวคเตอร์ปกติหรือนอร์มอลเวคเตอร์ (Normal Vector) ของพื้นผิวที่แสงส่องไปถึง การหาค่านอร์มอลเวคเตอร์สามารถทำได้โดยหาค่าเกรเดียนท์ของพื้นผิว ซึ่งเกรเดียนท์เป็นตัวดำเนินการ (Operator) ทางคณิตศาสตร์ชนิดหนึ่ง มีรูปแบบดังสมการ 3.15 โดยที่ $f(x, y, z)$ เป็นสเกลลาฟังก์ชันและสามารถหาอนุพันธ์ได้ (Differentiable)

$$grad \quad f = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} + \frac{\partial f}{\partial z} \tag{3.15}$$

เกรเดียนท์จะบอกถึงทิศทางและอัตราเร็วของการเปลี่ยนแปลงของฟังก์ชัน $f(x, y, z)$ ในกรณีที่ฟังก์ชันเป็นพื้นผิว

จากรูปที่ 3.9 ถ้าเกรเดียนท์ของฟังก์ชัน f ที่จุด P ไม่เป็นศูนย์แล้ว จะทำให้สามารถหาค่านอร์มอลเวคเตอร์ที่จุด P นั้น ๆ ได้ ถ้าเกรเดียนท์เป็นศูนย์จะหมายความว่าไม่มีความเปลี่ยนแปลงในพื้นผิวนั้น ๆ



รูปที่ 3.9 แสดงการหาค่าเวกเตอร์ปกติจากเกรเดียนท์ของพื้นผิว

ในกรณีของข้อมูลเชิงปริมาตร เกรเดียนท์จะบอกถึงทิศทางและอัตราเร็วในการเปลี่ยนแปลงของข้อมูลในปริมาตร ถ้าขนาดของเกรเดียนท์มีค่าเป็นศูนย์หมายความว่าไม่มีการเปลี่ยนแปลงในค่าของจุดภาพเทียบกับจุดภาพข้างเคียง การหาค่าเกรเดียนท์สามารถทำได้หลายวิธีซึ่งแต่ละวิธีก็จะมีความถูกต้องและความซับซ้อนของการคำนวณแตกต่างกันไป

แต่วิธีหนึ่งที่ง่ายและไม่ซับซ้อนคือการหาค่าเกรเดียนท์ด้วยตัวประมาณค่าแบบเซ็นทรัลดิฟเฟอเรนซ์ (The Central Difference Gradient Estimator) การประมาณค่าเกรเดียนท์ด้วยวิธีนี้สามารถทำได้ง่าย และรวดเร็ว แต่ก็จะให้คุณภาพของภาพด้อยกว่าวิธีอื่น เนื่องจากว่าในการคำนวณหาค่าเกรเดียนท์ด้วยวิธีนี้ใช้ข้อมูลในปริมาตรข้อมูลภาพเพียงหกจุดมาทำการคำนวณ

ตัวประมาณค่าเกรเดียนท์แบบเซ็นทรัลดิฟเฟอเรนซ์มีสมการดังนี้

$$\begin{aligned}\nabla d(x,y,z) = &\frac{1}{2}[d(x+1,y,z)-d(x-1,y,z)]\bar{i} \\ &+\frac{1}{2}[d(x,y+1,z)-d(x,y-1,z)]\bar{j} \\ &+\frac{1}{2}[d(x,y,z+1)-d(x,y,z-1)]\bar{k}\end{aligned} \tag{3.16}$$

$d(x, y, z)$ คือ ค่าความสว่างของจุดภาพที่ตำแหน่ง (x, y, z) ในปริมาตรข้อมูล

ซึ่งจากสมการที่ 3.16 จะเห็นได้ว่าเกรเดียนท์เป็นเวคเตอร์แบบสามมิติ ที่มีทั้งขนาดและทิศทางในแกน x, y และ z

ค่าเกรเดียนท์จะถูกใช้ในสองขั้นตอนของการสร้างภาพเชิงปริมาตรคือขบวนการแยกแยะชนิดหรือแบ่งกลุ่มของข้อมูลภาพและขบวนการให้แสงเงา ซึ่งในขบวนการการแบ่งกลุ่มข้อมูลสามารถนำเอาค่าขนาดของเกรเดียนท์ไปใช้ได้เลย แต่ในขั้นตอนของการให้แสงเงาจะต้องคำนวณหาค่านอร์มอลเวคเตอร์ก่อน

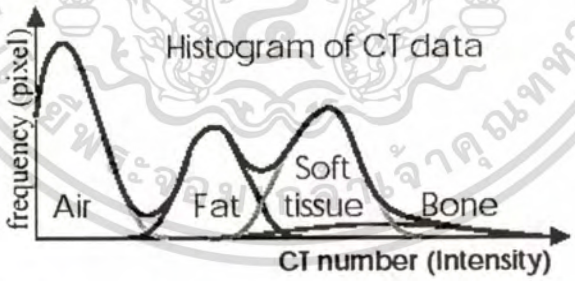
3.4.2 นอร์มอลเวคเตอร์ (Normal Vector)

นอร์มอลเวคเตอร์ในภาพเชิงปริมาตรมีนิยาม คือ เวคเตอร์หนึ่งหน่วยที่ขนานไปกับเกรเดียนท์ที่คำนวณโดยใช้ข้อมูลระดับเทา ถ้ากำหนดให้ $\nabla d(x,y,z)$ แทนเกรเดียนท์ และ n (x, y, z) แทนนอร์มอลเวคเตอร์ จะได้ว่า

$$n(x,y,z) = \frac{\nabla d(x,y,z)}{|\nabla d(x,y,z)|} \tag{3.17}$$

3.4.3 ฮีสโตรแกรม (Histogram)

ฮีสโตรแกรมของภาพจะบอกถึงความถี่ของค่าความสว่างหรือค่าระดับเทา (Gray Level) ที่พบในจุดภาพ ค่าของฮีสโตแกรมจะถูกพล็อตเป็นกราฟสองมิติโดยข้อมูลในแนวแกน x จะเป็นค่าระดับเทาและข้อมูลในแกน y จะเป็นความถี่ของข้อมูลหรือจำนวนจุดที่พบในภาพนั้น ๆ ดังแสดงในรูปที่ 3.10



รูปที่ 3.10 ฮีสโตแกรมของข้อมูลภาพทางการแพทย์ที่ถ่ายด้วยเครื่อง CT

การใช้ฮีสโตแกรมในการสร้างภาพเชิงปริมาตรจะช่วยให้สามารถกำหนดฟังก์ชันการถ่ายโอน (Transfer Function) ได้ง่ายขึ้น ข้อมูลที่มีค่าของฮีสโตแกรมอยู่ในยอดแหลม (Peak) เดียวกันมักจะเป็นวัตถุกลุ่มเดียวกันในข้อมูลของภาพ เช่น ที่ค่าระดับเทาน้อยที่สุดมักจะเป็นอากาศ ส่วนที่ค่าระดับเทามากที่สุดมักจะเป็นกระดูก

3.4.4 ฟังก์ชั่นถ่ายโอน (Transfer Function)

ฟังก์ชันถ่ายโอนมีไว้เพื่อกำหนดค่าความทึบแสงให้กับจุดภาพโดยตัวแปรของฟังก์ชันอาจเป็นค่าความสว่าง หรืออาจเป็นทั้งความสว่างและขนาดของเกรเดียนท์ หรืออาจมีตัวแปรอื่นเข้ามาร่วมด้วยก็ได้ ดังแสดงในสมการ 3.18

$$\alpha_i = O(I_i, |\nabla_i|, ..., ..., ...) \tag{3.18}$$

O คือฟังก์ชันการถ่ายโอนของค่าความทึบแสง

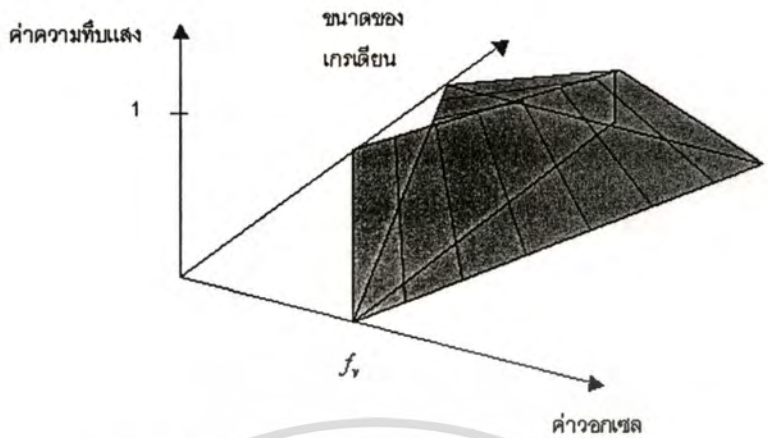
I_i คือค่าความสว่างหรือค่าระดับเทาของจุดภาพ

$|\nabla_i|$ คือขนาดของเกรเดียนท์ของจุดภาพ

ในกรณีของภาพที่ถ่ายจากเครื่อง CT ค่าความสว่างของจุดภาพในแต่ละอวัยวะจะแตกต่างกันอย่างชัดเจน การใช้ค่าความสว่างของจุดภาพเพียงอย่างเดียวก็เพียงพอในการแยกแยะอวัยวะต่างๆออกจากกัน แต่ในกรณีของภาพที่ถ่ายจากเครื่อง MRI อวัยวะที่ต่างกันอาจมีค่าความสว่างเท่ากันได้ การใช้ค่าความสว่างเพียงอย่างเดียวในการแยกแยะภาพของอวัยวะซึ่งถ่ายจากเครื่อง MRI อาจไม่เพียงพอดังนั้นจึงต้องมีการพิจารณาค่าอื่นๆมาร่วมด้วย เช่น ขนาดของเกรเดียนท์ เป็นต้น เกรเดียนท์จะบอกถึงความเร็วในการเปลี่ยนแปลงของข้อมูล ในบริเวณรอยต่อของอวัยวะสองชนิดจะมีความเร็วในการเปลี่ยนแปลงของค่าความสว่างอย่างคงที่ซึ่งการใช้ตัวดำเนินการเกรเดียนท์ (Gradient Operator) กระทำกับข้อมูลในปริมาตรสามารถใช้หาขอบเขตของอวัยวะได้ Levoy (1988) ได้นำเสนอฟังก์ชันการถ่ายโอนซึ่งสามารถใช้ในการสร้างภาพเชิงปริมาตรได้ดีโดยใช้ค่าความสว่างและขนาดของเกรเดียนท์ ฟังก์ชันถ่ายโอนของ Levoy มีลักษณะดังสมการ 3.19

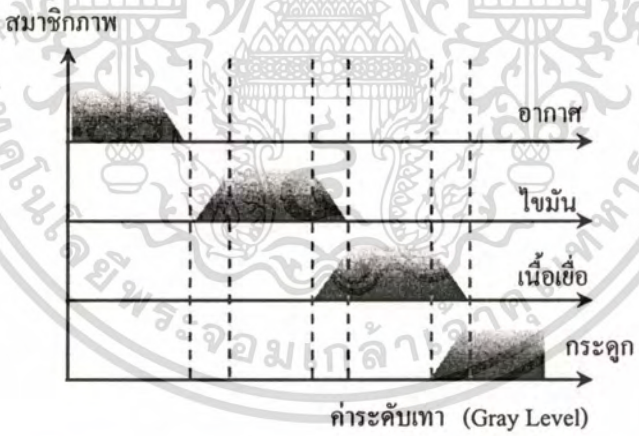
$$\begin{aligned}
&\alpha_i(r, f_v) = 1 - \frac{1}{r|\nabla_i|}|f_v - I_i| && \text{ถ้า } |\nabla_i| > 0 \text{ และ } I_i - r|\nabla_i| \le f_v \le I_i + r|\nabla_i| \\
&\alpha_i(r, f_v) = 1 && \text{ถ้า } |\nabla_i| = 0 \\
&\alpha_i(r, f_v) = 0 && \text{ค่าอื่นๆ}
\end{aligned} \tag{3.19}$$

ค่าความสว่างของจุดภาพที่ต้องการให้ปรากฏคือ f_v ซึ่งมีค่าความทึบแสงมากที่สุดส่วนค่าความสว่างที่ไม่ใช่ f_v และที่ขนาดของเกรเดียนท์ค่าต่างๆจะมีค่าความทึบแสงลดลงโดยรูปของฟังก์ชันถ่ายโอนจะมีลักษณะคล้ายเต็นท์ (Tent) ดังแสดงในรูปที่ 3.11



รูปที่ 3.11 ฟังก์ชันถ่ายโอนของ Levoy

ในกรณีที่มีหลายวัตถุอยู่ในข้อมูลและขอบเขตของวัตถุบางอย่างอาจมีการผสมกันเช่น ไขมันกับเนื้อเยื่อโดยไม่สามารถกำหนดได้แน่นอนว่าวัตถุเหล่านั้นเป็นอวัยวะประเภทใด การกำหนดค่าของความสว่างที่ใช้ในฟังก์ชันส่งผ่านจะกำหนดเป็นเซ็ทของข้อมูลที่เป็นลักษณะฟัซซี่เซ็ท (Fuzzy Set) ซึ่งแสดงดังรูปที่ 3.12

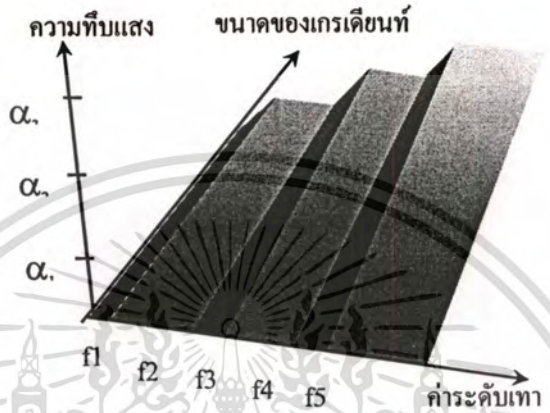


รูปที่ 3.12 การแยกแยะชนิดของอวัยวะซึ่งมีขอบเขตผสมกัน

การจะแยกวัตถุซึ่งขอบเขตบางส่วนผสมกันต้องใช้ฟังก์ชันส่งผ่านที่มีการกำหนดค่า f_v ตามค่าความสว่างของวัตถุซึ่งไม่ผสมกับวัตถุอื่น ในส่วนของวัตถุที่ผสมกันจะกำหนดให้ค่าของความทึบแสงมีค่าน้อยกว่าค่าความทึบแสงของวัตถุนั้นเพียงอย่างเดียว ซึ่งในกรณีของวัตถุที่มีขอบเขตบางส่วนผสมกันนี้ Levoy ได้นำเสนอฟังก์ชันถ่ายโอนดังสมการ 3.20

$$\alpha_i = \begin{cases} \frac{|\nabla_i|}{f_{n+1} - f_n}[\alpha_n(f_{n+1} - f_i) + \alpha_{n+1}(f_i - f_n)] & ; n = 0,2,4,... \\ \alpha_n|\nabla_i| & ; n = 1,3,5,... \end{cases} \quad (3.20)$$

ฟังก์ชันถ่ายโอนของสมการ 3.20 จะมีลักษณะดังรูปที่ 3.13



รูปที่ 3.13 ฟังก์ชันถ่ายโอนของข้อมูลที่มีมากกว่าหนึ่งวัตถุ

3.4.5 การให้สี

การให้สีแก่กลุ่มของข้อมูลทำให้สามารถแยกแยะกลุ่มข้อมูลได้ง่ายขึ้นโดยข้อมูลที่อยู่ในกลุ่มที่ต่างกันจะถูกให้สีต่างกัน ส่วนข้อมูลสองกลุ่มที่มีขอบเขตปนกันค่าสีของข้อมูลในส่วนนี้จะเป็นการอินเทอร์โพเลชั่นแบบเชิงเส้นของข้อมูลสองกลุ่มนั้น ฟังก์ชันถ่ายโอนของการให้สีจะถูกทำในแต่ละค่าสี แดง เขียว น้ำเงิน ดังชุดสมการเหล่านี้ตามลำดับ

$$R_i = T_r(I_i, |\nabla_i|, ..) \quad (3.21)$$

$$G_i = T_g(I_i, |\nabla_i|, ..) \quad (3.22)$$

$$B_i = T_b(I_i, |\nabla_i|, ..) \quad (3.23)$$

T_r, T_g, T_b เป็นฟังก์ชันถ่ายโอนของค่าสี แดง เขียว และ น้ำเงินตามลำดับ

ค่าสีของจุดภาพนี้จะถูกนำไปให้แสงเงาอีกครั้งในกระบวนการให้แสงเงาเพื่อให้ได้สีที่สมจริงมากขึ้น

3.5 การส่องสว่าง และ การให้แสงเงา

การส่องสว่างและการให้แสงเงาในคอมพิวเตอร์กราฟิกจะถูกทำเพื่อให้ภาพที่ได้มีความสมจริงมากยิ่งขึ้นโดยอาศัยสมการคณิตศาสตร์มาจำลองผลของแสงในธรรมชาติที่มีต่อวัตถุ ซึ่งประกอบด้วย การเกิดเงา การสะท้อน การกระเจิง และการถูกดูดกลืนของแสงเมื่อตกกระทบวัตถุผลของการให้แสงเงาในแต่ละวัตถุจะแตกต่างกันขึ้นอยู่กับทิศทางของแสง มุมระหว่างผู้สังเกต (Viewer) กับแสง และคุณสมบัติของพื้นผิวของวัตถุเป็นต้น

ในการสร้างภาพเชิงปริมาตร ความสมจริงของภาพที่ได้อาจไม่ใช่สิ่งที่สำคัญมากที่สุด การให้แสงเงาจะถูกทำให้สามารถมองเห็นโครงสร้างของวัตถุได้ดีขึ้นซึ่งจะทำให้ผู้สังเกตเข้าใจความหมายของภาพได้ง่าย นอกจากนั้นการใช้แบบจำลองของการส่องสว่างและให้แสงเงาที่มีความสมจริงมากๆจะต้องใช้เวลาในการประมวลผลมากตามไปด้วย องค์ประกอบที่ใช้ในการคำนวณการส่องสว่างและให้แสงเงามีดังนี้

3.5.1 แสงแวดล้อม (Ambient Light)

องค์ประกอบนี้เป็นส่วนที่ง่ายที่สุดของการให้แสง แสงแวดล้อมจะมีค่าเท่ากันในทุกทิศทางไม่ขึ้นอยู่กับทิศทางของแหล่งกำเนิดแสง แต่การใช้องค์ประกอบของแสงแวดล้อมอย่างเดียวในการให้แสงจะทำให้ได้ภาพไม่สมจริง เนื่องจากแสงที่เกิดขึ้นที่วัตถุจะมีความสว่างเท่ากันทั้งหมด ตัวอย่างขององค์ประกอบนี้ในธรรมชาติคือแสงจากดวงอาทิตย์เป็นต้น องค์ประกอบของแสงแบบนี้มีรูปแบบดังสมการ 3.24

$$L_a = L_i k_a \tag{3.24}$$

L_a คือ ความเข้มของแสงที่ผ่านการให้แสงแวดล้อมแล้ว

L_i คือ สีของแสงแวดล้อม

k_a คือ สัมประสิทธิ์การสะท้อนแสงแวดล้อมของวัตถุ (Ambient-Reflection Coefficient)

ผลของแสงแวดล้อมที่มีต่อวัตถุ แสดงดังรูปที่ 3.14

k_a=0.3 k_a=0.5 k_a=0.7 k_a=0.9

รูปที่ 3.14 ผลของแสงแวดล้อมที่มีต่อวัตถุ ซึ่งมีค่า k_a ต่าง ๆ กัน

3.5.2 การสะท้อนแบบกระจาย (Diffuse Reflection)

เนื่องจากแสงแวดล้อมเพียงอย่างเดียวจะทำให้ภาพที่ได้ไม่สมจริง การเพิ่มแหล่งกำเนิดแสงแบบจุด (Point Light Source) เข้าไปในระบบจะช่วยแก้ปัญหานี้ได้ แนวลำแสงของแหล่งกำเนิดแสงแบบนี้จะกระจายออกไปทุกทิศทุกทางอย่างสม่ำเสมอ พื้นผิวที่ตั้งฉากกับแหล่งกำเนิดแสงจะได้รับพลังงานแสงในปริมาณมากที่สุด ส่วนอื่นๆที่ไม่ตั้งฉากกับแหล่งกำเนิดแสงจะได้รับพลังงานที่น้อยกว่า

สมการของการสะท้อนแบบกระจายคือ

$$L_d = L_i k_d \cos\theta \tag{3.25}$$

L_d คือ ความเข้มของแสงสะท้อนแบบกระจาย

L_i คือ สีของแหล่งกำเนิดแสง

k_d คือ สัมประสิทธิ์การสะท้อนแสงแบบกระจายของวัตถุ (Diffuse-Reflection Coefficient)

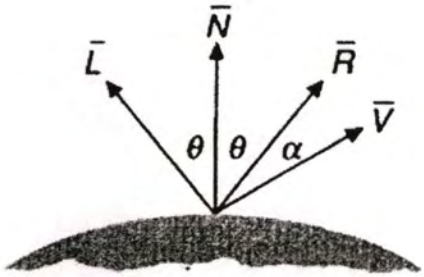
θ คือ มุมที่แสงตกกระทบกับผิวของวัตถุ

ผลของการสะท้อนแสงแบบกระจายร่วมกับผลของแสงแวดล้อมที่มีต่อวัตถุ แสดงในรูปที่ 3.15

รูปที่ 3.15 ผลของการสะท้อนแสงแบบกระจายร่วมกับผลของแสงแวดล้อมที่มีต่อวัตถุที่มีค่า k_d ต่าง ๆ กัน และแสงแวดล้อมมีค่า $k_a = 0.3$

3.5.3 การสะท้อนแบบกระจก (Specular Reflection)

การสะท้อนแบบนี้พบได้ในวัตถุที่เป็นมันวาวเช่น โลหะ กระจก หรือวัตถุที่มีผิวมัน สีของแสงที่สะท้อนออกมาจะเป็นจุดสีสว่าง ไม่ใช่สีของวัตถุเหมือนในกรณีของการสะท้อนแบบกระจาย สำหรับวัตถุที่สามารถสะท้อนแสงได้อย่างสมบูรณ์ เช่นกระจกเงา ค่าสัมประสิทธิ์การสะท้อนแสงของวัตถุ (s) ควรจะเข้าใกล้อนันต์ องค์ประกอบการสะท้อนแบบกระจกแสดงดังรูปที่ 3.16



รูปที่ 3.16 องค์ประกอบการสะท้อนแบบกระจก

สมการของการสะท้อนแบบกระจกคือ

$$L_s = L_i k_s O_s (\bar{R}.\bar{V})^s \tag{3.26}$$

L_s คือความสว่างของแสงที่เกิดจากการสะท้อนแบบกระจก

L_i คือสีของแหล่งกำเนิดแสง

k_s คือสัมประสิทธิ์การสะท้อนแสงแบบกระจก (Specular-Reflection Coefficient)

O_s คือสีของแสงสะท้อน

$\bar{V}$ คือเวกเตอร์ที่ชี้จากพื้นผิวไปที่ผู้สังเกต

$\bar{R}$ คือเวกเตอร์ปกติของการสะท้อน

s คือสัมประสิทธิ์การสะท้อนแสงของวัตถุ

ผลของการสะท้อนแสงแบบกระจกร่วมกับผลของการสะท้อนแบบกระจาย และแสงแวดล้อมที่มีต่อวัตถุ แสดงดังรูปที่ 3.17

k_s=0.3 k_s=0.5 k_s=0.7 k_s=0.9

รูปที่ 3.17 ผลของการสะท้อนแสงแบบกระจกร่วมกับผลของการสะท้อนแบบกระจาย และแสงแวดล้อมที่มีต่อวัตถุ

3.6 การประกอบภาพ

การประกอบภาพเป็นขั้นตอนสุดท้ายของการสร้างภาพเชิงปริมาตรโดยจะเป็นการรวมคุณสมบัติของจุดแสงที่ได้ถูกคำนวณไว้ก่อนหน้านี้มาสร้างเป็นจุดภาพบนระนาบการมอง การฉายแสงจะสร้างภาพโดยการใช้ข้อมูลทั้งปริมาตรข้อมูลมาทำการคำนวณโดยอาศัยสมมติฐานที่ว่าข้อมูลในปริมาตรมีคุณสมบัติ กระเจิง (Scatter) กีดขวาง (Occlude) สร้าง (Generate) และสะท้อน (Reflect) แสง ภาพในระนาบการมองจะเกิดจากผลของแสงที่ส่องลงบนข้อมูลในปริมาตรรวมกันตลอดหนึ่งแนวลำแสงโดยสามารถเขียนเป็นสมการทางคณิตศาสตร์ได้ตามที่นำเสนอไปแล้วในหัวข้อเรื่องสมการสร้างภาพเชิงปริมาตร สมการที่ 2.3

$$L(x) = \int_{x}^{x_B} e^{-\int_{x}^{x'} \phi_t(x'')dx''} \varepsilon(x')dx'$$

ซึ่งในหัวข้อนี้จะเป็นการกล่าวถึงรายละเอียดเพิ่มเติม และชี้ให้เห็นถึงคุณสมบัติต่าง ๆ เพิ่มมากขึ้น โดยจะกล่าวถึงวิธีการประกอบภาพแบบหน้าไปหลัง (Front-to-Back Composition) และการประกอบภาพจากหลังมาหน้า (Back-to-Front Composition) ดังมีรายละเอียดดังนี้

3.6.1 การประกอบภาพแบบหน้าไปหลัง

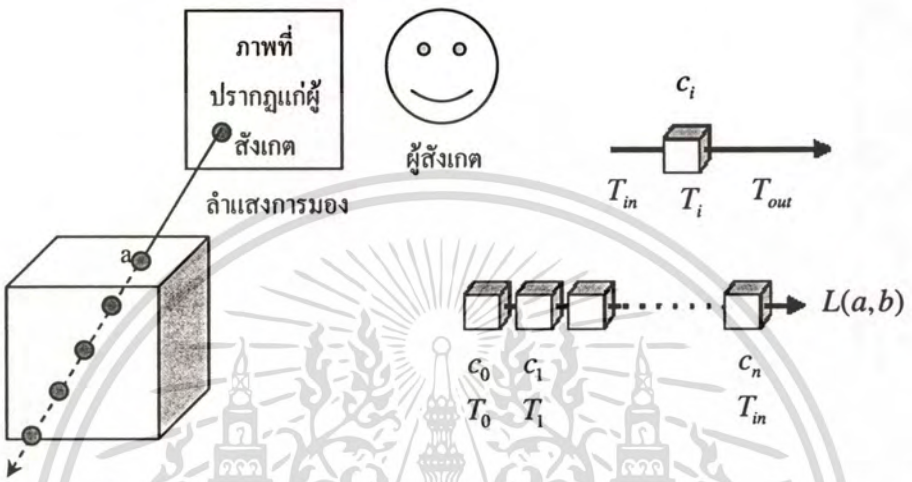
การประกอบภาพแบบหน้าไปหลังจะทำการสมมติสถานการณ์ว่าแสงได้เดินทางออกจากฉากหรือภาพผลลัพธ์ ลึกเข้าไปในวัตถุ โดยในระหว่างที่แสงเดินทางเข้าไปในเนื้อวัตถุนั้นจะมีการสูญเสียพลังงานด้วย ทำให้แสงสามารถเดินทางเข้าไปได้ลึกเพียงแค่ระดับที่มีความทึบแสงสะสมเท่ากับหนึ่งเท่านั้น จากคุณสมบัติข้อนี้ทำให้มีการนำเอาเทคนิคการสิ้นสุดลำแสงก่อน (Early ray termination) มาใช้ช่วยทำให้ลดการคำนวณลงไปได้ เพราะถ้าค่าความทึบแสงสะสมในแนวลำแสงนั้นมีค่ามากกว่าหนึ่งแล้ว ก็สามารถยกเลิกการคำนวณในแนวทางเดินแสงนั้นได้เลย เพราะถือว่าวัตถุหรือว็อกเซลที่อยู่ในแนวลำแสงนั้น ๆ ทึบแสงโดยสมบูรณ์แล้วนั่นเอง

สมการการประกอบแบบหน้าไปหลังนี้ได้อธิบายไว้แล้วในหัวข้อเรื่องสมการการสร้างภาพเชิงปริมาตรแบบไม่ต่อเนื่อง หรือสมการที่ 2.5 ดังนี้

$$\begin{aligned} L(x) &= \sum_{i=0}^{n-1} c_i \cdot \prod_{j=0}^{i-1} (1-\alpha_j) \\ &= c_0 + c_1(1-\alpha_0) + c_2(1-\alpha_0)(1-\alpha_1) + \ldots \\ &\quad + c_{n-1}(1-\alpha_0)\ldots(1-\alpha_{n-2}) \\ &= c_0 over \cdot c_1 over \cdot c_2 over \cdots over \cdot c_{n-1} \end{aligned}$$

จากสมการนี้ทำให้เราสามารถนิยามคุณสมบัติของวัตถุได้อีกอย่างหนึ่ง เพื่อทำให้เกิดความง่ายในการคำนวณ นั่นคือ ค่าความโปร่งแสง มีนิยามตามสมการ

$$T_j = 1 - \alpha_j \tag{3.28}$$



รูปที่ 3.18 ลักษณะและทิศทางของการประกอบภาพแบบหน้าไปหลัง

จากรูปที่ 3.18 สมการการประกอบภาพแบบหน้าไปหลังที่ตำแหน่งใดๆในแนวลำแสงสามารถเขียนได้เป็น

$$\begin{aligned} T_{out} &= T_{in} T_i \\ L_{out} &= L_{in} + T_{in} c_i \end{aligned} \tag{3.29}$$

L_{out} และ T_{out} เป็นความสว่างและความโปร่งแสงสะสมที่ทะลุผ่านจุดแสงในตำแหน่ง i

L_{in} และ T_{in} เป็นความสว่างและความโปร่งแสงสะสมที่เข้ามาสู่จุดแสงในตำแหน่ง i

c_i และ T_i เป็นความสว่างและความโปร่งแสงในจุดแสงที่กำลังพิจารณา

จากสมการ 3.29 สามารถเขียนวิธีการประกอบภาพแบบหน้าไปหลังได้ดังนี้

```
1:          Trans = 1.0;
            Inten = I[0];
            for(i = 1;  i <= n;  i++){
5:             Trans = Trans * T[i-1];
               Inten = Inten + Trans * c[i];
            }
```

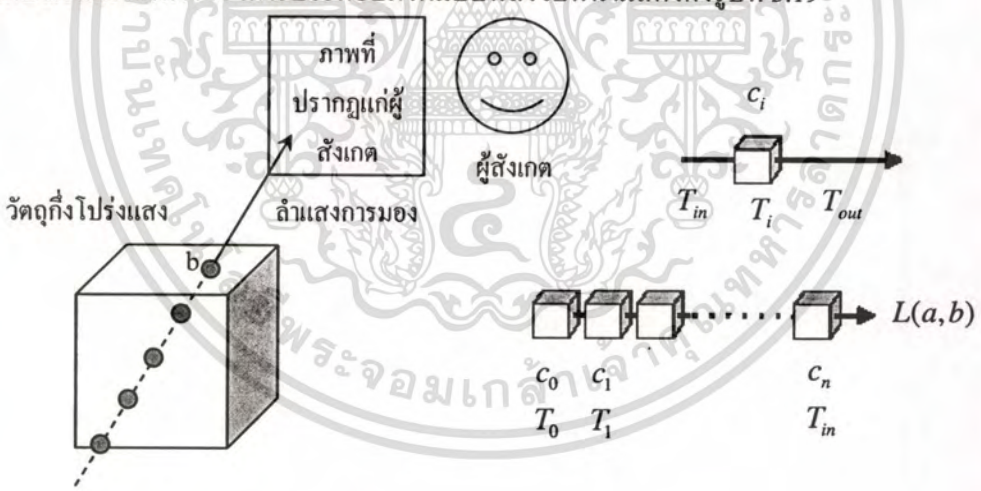
ในการประกอบภาพแบบหน้าไปหลังนี้จะต้องทำการคำนวณค่าของความโปร่งแสงสะสมก่อน จากนั้นนำค่าความโปร่งแสงสะสมไปคำนวณหาค่าความสว่างสะสมอีกครั้งหนึ่ง การที่ต้องคำนวณหาความโปร่งแสงสะสมก่อนเป็นข้อเสียของการประกอบภาพแบบหน้าไปหลังเพราะใช้การคำนวณมากกว่าการประกอบภาพแบบหลังไปหน้า แต่การประกอบภาพแบบหน้าไปหลังมีข้อดีคือใช้เทคนิคการสิ้นสุดลำแสงก่อนได้ โดยที่จุดแสงใดมีค่าของความโปร่งแสงสะสมเป็นศูนย์ (ตามสมการ 3.28) ก็สามารถหยุดการประกอบภาพในแนวลำแสงได้

3.6.2 การประกอบภาพแบบหลังไปหน้า

การประกอบภาพแบบหลังไปหน้านี้เป็นการสมมติสถานการณ์ว่าแสงได้เดินทางจากด้านหลังของวัตถุแล้วมากระทบที่ฉากหรือภาพผลลัพธ์ทุก ๆ ลำแสง โดยไม่มีการสูญเสียพลังงานระหว่างที่เดินทางผ่านวัตถุหรือปริมาตร สมการของการประกอบภาพแบบหลังไปหน้าคือ

$$L(a,b) = \sum_{i=0}^{n} c_i \prod_{j=i+1}^{n} (1-\alpha_j) \tag{3.30}$$

ลักษณะและทิศทางของการประกอบภาพแบบหลังไปหน้าแสดงดังรูปที่ 3.19



รูปที่ 3.19 ลักษณะและทิศทางของการประกอบภาพแบบหลังไปหน้า

สมการการประกอบภาพแบบหลังไปหน้าที่ตำแหน่งใดๆในแนวลำแสงสามารถเขียนได้เป็น

$$L_{out} = L_{in}T_i + c_i \tag{3.31}$$

L_{out} และ T_{out} เป็นความสว่างและความโปร่งแสงสะสมที่ทะลุผ่านจุดแสงในตำแหน่ง i

L_{in} และ T_{in} เป็นความสว่างและความโปร่งแสงสะสมที่เข้ามาสู่จุดแสงในตำแหน่ง i

c_i และ T_i เป็นความสว่างและความโปร่งแสงในจุดแสงที่กำลังพิจารณา

จากสมการ 3.31 สามารถเขียนวิธีการประกอบภาพแบบหลังไปหน้าได้ดังนี้

```
1:      Inten = c[0];
        for(i = 1; i <= n; i++){
                Inten = Inten * T[i] + c[i];
        }
```

3.7 สรุป

ในบทนี้เป็นการกล่าวถึงหลักการสร้างภาพเชิงปริมาตรแบบลำดับภาพทั้งหมด โดยได้แสดงให้เห็นวิธีการต่าง ๆ ที่สำคัญของการสร้างภาพโดยใช้วิธีการฉายแสง ซึ่งเป็นการสร้างภาพเชิงปริมาตรแบบลำดับภาพอีกแบบหนึ่ง โดยขั้นตอนต่าง ๆ นั้นประกอบด้วยการอินเทอร์โพเลชั่น การแบ่งกลุ่มข้อมูล การส่องสว่างและการให้แสงเงา โดยประกอบไปด้วยแสงแวดล้อม (Ambient Light) การสะท้อนแบบกระจาย (Diffuse Reflection) การสะท้อนแบบกระจก (Specular Reflection) และ สุดท้ายคือการประกอบภาพ ซึ่งอธิบายทั้งการประกอบภาพแบบหน้าไปหลัง และการประกอบภาพแบบหลังไปหน้า

สำหรับบทถัดไปนั้นจะเป็นการกล่าวถึงการสร้างภาพเชิงปริมาตรอีกแบบหนึ่ง นั่นคือการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุ โดยจะได้ยกตัวอย่างวิธีการสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิด ซึ่งเป็นวิธีที่ใช้ในวิทยานิพนธ์ฉบับนี้

บทที่ 4
การสร้างภาพเชิงปริมาตรแบบลำดับวัตถุ

4.1 บทนำ

หลังจากที่กล่าวถึงการสร้างภาพเชิงปริมาตรด้วยวิธีการแบบลำดับภาพโดยยกตัวอย่างวิธี Ray Casting ไปแล้วนั้น ในบทนี้เราจะกล่าวถึงการสร้างภาพเชิงปริมาตรในอีกรูปแบบหนึ่ง คือการสร้างภาพแบบลำดับวัตถุ (Object Order Algorithm) โดยจะกล่าวถึงการสร้างภาพเชิงปริมาตรแบบเร็วโดยใช้วิธีการแปลงเฉือนและบิด ที่ได้ถูกปรับปรุงโดย Lacroute (1995) เป็นสำคัญ โดยจะเริ่มจากการกล่าวถึงเทคนิคในการเพิ่มความเร็วในการสร้างภาพเชิงปริมาตรแบบต่าง ๆ ที่เป็นที่นิยมในงานวิจัยที่ผ่าน ๆ มาเป็นลำดับแรก แล้วค่อยลงรายละเอียดในหัวข้อถัด ๆ ไปในบทนี้

4.2 เทคนิคการเพิ่มความเร็วในการสร้างภาพเชิงปริมาตร

มีผู้พยายามคิดค้นหาวิธีการต่าง ๆ (Algorithm) เพื่อใช้ในการเพิ่มความเร็ว และลดเวลาในการคำนวณลงเป็นจำนวนมาก ในหัวข้อนี้เราจะกล่าวถึงวิธีที่ถูกนำมาใช้ในการสร้างภาพเชิงปริมาตรแบบต่าง ๆ ซึ่งเราได้เห็นจากบทที่ 3 แล้วว่าการสร้างภาพแบบเดิมนั้น เป็นวิธีการแบบง่าย ๆ ไม่มีการนำเอาเทคนิคการเพิ่มความเร็วเข้ามาช่วย ทำให้ต้องใช้เวลาในการคำนวณนาน ดังนั้นในหัวข้อนี้เทคนิคที่สำคัญสองอย่างได้ถูกนำมาใช้ คือ การใช้รูปแบบโครงสร้างข้อมูลแบบพิเศษ (Special Data Structure) และการสิ้นสุดลำแสงก่อน (Early Ray Termination)

4.2.1 การใช้รูปแบบโครงสร้างข้อมูลแบบพิเศษ (Special Data Structure)

วิธีการนี้อาศัยและใช้ประโยชน์จากความเกี่ยวเนื่องของข้อมูลภาพต้นฉบับ (Input Data Coherence) แล้วนำมาสร้างเป็นรูปแบบโครงสร้างข้อมูลแบบพิเศษ เพื่อทำให้สามารถนำข้อมูลไปใช้คำนวณได้อย่างรวดเร็ว งานวิจัยที่ถูกนำเสนอส่วนใหญ่จะใช้วิธีการสนใจข้อมูลส่วนที่มีค่าความทึบแสงสูง หรือเป็นข้อมูลภาพจริง ๆ เท่านั้น จะไม่สนใจข้อมูลในส่วนที่เป็นอากาศหรือค่าความทึบแสงต่ำแล้วนำค่าที่สนใจมาสร้างเป็นรูปแบบโครงสร้างข้อมูลแบบพิเศษขึ้น โดยจากงานวิจัยของ Levoy (1990) ได้ระบุว่าในปริมาตรข้อมูลภาพต้นฉบับมักจะมีข้อมูลส่วนที่โปร่งแสงหรือข้อมูลที่ไม่จำเป็นอยู่สูงถึง 70-95% ดังนั้นถ้าลดการคำนวณที่ไม่จำเป็นเหล่านี้ลงได้จะทำให้การสร้างภาพเชิงปริมาตร

โดยวิธีการฉายแสงธรรมดาเร็วขึ้นได้ 3-5 เท่าเลยทีเดียว ตัวอย่างงานวิจัยที่ใช้โครงสร้างข้อมูลแบบพิเศษนี้ เช่น การใช้โครงสร้างข้อมูลแบบออคทรีและพีรามิด (Octrees and Pyramids) โดย Meagher (1982) , เค-ดีทรี (K-D Trees) โดย Subramanian and Fussel (1990), การเข้ารหัสแบบรันเลงท์ (Run-length Encoding) โดย Reynolds *et. al.* (1987) และโครงสร้างข้อมูลโดยการแปลงระยะทาง (Distance Transform) โดย วรเทพ ไพบูลย์รัตนากร (2544) โครงสร้างข้อมูลเหล่านี้ทำให้เมื่อทำการฉายแสงเข้าไปในปริมาตรการคำนวณสามารถข้ามวอ็กเซล (Voxel) ที่โปร่งแสงไปได้ ทำให้การคำนวณที่ไม่จำเป็นลดน้อยลง

งานวิจัยที่มีการใช้โครงสร้างข้อมูลแบบพิเศษอีกรูปแบบหนึ่งคือ การแปลงข้อมูลของปริมาตรเริ่มต้น โดยใช้วิธีการแปลงฟูเรียร์ (Fourier Transform) แบบสามมิติเพื่อลดข้อมูลที่ไม่จำเป็นลง หลังจากนั้นจึงค่อยนำข้อมูลที่แปลงแล้วไปคำนวณ ทำให้ข้อมูลที่นำไปใช้คำนวณในขั้นตอนการสร้างภาพเชิงปริมาตรลดลง (Nikolaidis and Pitas, 2001:148) วิธีการนี้ถูกเรียกว่า การสร้างภาพเชิงปริมาตรแบบ
ฟูเรียร์ (Fourier Volume Rendering)

การใช้รูปแบบโครงสร้างข้อมูลแบบพิเศษนี้จะต้องทำการคำนวณล่วงหน้า (Preprocessing) ก่อนเสมอ ดังนั้นจึงต้องมีเวลาที่สูญเสียไปในการคำนวณล่วงหน้านี้ แต่ทั้งนี้ก็สามารถยอมรับได้ หากเวลาโดยรวมในการสร้างภาพลดลง แต่ยังไงก็ตามงานวิจัยที่กล่าวมาก็ยังไม่สามารถคำนวณและตอบสนองต่อผู้ใช้ในระดับของเวลาจริง (Real Time and Interactive) ได้ ดังนั้นจึงมีผู้คิดค้นวิธีในการสร้างภาพแบบใหม่ ๆ ออกมาอยู่เสมอ

จากงานวิจัยที่กล่าวมาจะเห็นได้ว่าถึงแม้มีการใช้โครงสร้างข้อมูลแบบพิเศษกับการสร้างภาพเชิงปริมาตรแบบลำดับภาพแล้วก็ตาม ไม่ว่าจะเป็นโครงสร้างข้อมูลแบบออคทรีหรือการแปลงปริมาตรโดยใช้คุณสมบัติของฟูเรียร์ แต่ความยุ่งยากในการคำนวณก็ยังมีอยู่เพราะต้องทำการพิจารณาข้อมูลปริมาตรเป็นสามมิติ อีกทั้งยังเสียเวลาในการสลับการทำงานระหว่างสมาชิกภายในทรีหรือเกิดการสูญเสียเวลาจากโอเวอร์เฮด (Overhead) ขึ้น เมื่อลำแสงผ่านจากทรี (Tree) หนึ่งไปยังอีกทรีหนึ่งในลำดับการฉายแสง ทำให้การใช้เทคนิคโครงสร้างข้อมูลแบบพิเศษนี้ยังมีข้อจำกัดอยู่บ้างสำหรับการสร้างภาพเชิงปริมาตรแบบลำดับภาพ แต่ถ้าหากใช้โครงสร้างข้อมูลแบบพิเศษกับการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุ เราสามารถพิจารณาทีละแผ่นสไลด์ (Image Slice) ซึ่งมองเป็นข้อมูลสองมิติได้เลย ทำให้ลดความยุ่งยากในการคำนวณ และลดค่าโอเวอร์เฮด (Overhead) ในการค้นหาข้อมูลของสมาชิกในทรี (Trees) ลงได้ เพราะเป็นการเข้าถึงข้อมูลตามลำดับของข้อมูลในหน่วยความจำหรือตามลำดับที่ถูกเก็บอยู่ในดิสก์ ซึ่งการใช้โครงสร้างข้อมูลแบบพิเศษนี้เป็นข้อดีที่เหมาะกับการนำมาใช้ในการสร้างภาพเชิง

ปริมาตรแบบลำดับวัตถุเป็นอย่างมาก ดังจะเห็นได้จากงานวิจัยของ Meagher (1982) ซึ่งใช้ควอดทรี (Quad trees) ในการพิจารณาข้อมูลเพื่อละเลยจุดภาพที่ไม่สนใจ

4.2.2 การสิ้นสุดลำแสงก่อน (Early Ray Termination)

เป็นอีกวิธีการที่สามารถทำได้ง่ายมากในการสร้างภาพเชิงปริมาตรแบบลำดับภาพ โดยที่การสร้างภาพเชิงปริมาตรแบบนี้จะทำการตรวจสอบความเข้มแสงสะสมตลอดแนวลำแสงว่ามีค่ามากกว่าค่าเทรชโชล (Threshold) ที่ตั้งไว้หรือไม่ในขั้นตอนการประกอบภาพแบบหน้าไปหลัง (Front to Back) ถ้าหากมีค่ามากกว่าก็จะจบการทำงานในแนวลำแสงนั้นทันที ทำให้ไม่ต้องคำนวณตลอดแนว ทุก ๆ แนวลำแสง

การสิ้นสุดลำแสงก่อนนี้ช่วยลดเวลาที่ใช้ในการคำนวณลงไปได้มากพอสมควร โดยจากงานวิจัยของ Levoy (1990) เมื่อตั้งค่าเทรชโชลความทึบแสงสะสมไว้ที่ 95% พบว่าการสิ้นสุดลำแสงก่อนนี้คำนวณได้รวดเร็วกว่าวิธีฉายแสงธรรมดา 1.6-2.2 เท่า และเมื่อใช้ร่วมกับโครงสร้างข้อมูลแบบออคทรีแล้วจะทำให้ความเร็วเพิ่มขึ้น 5-11 เท่าเลยทีเดียว

ส่วนการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุนั้น ได้มีผู้เสนองานวิจัยเกี่ยวกับการสิ้นสุดลำแสงก่อนออกมาเช่นกันคือ งานวิจัยของ Raynolds (1987) โดยได้ใช้ประโยชน์จากการเข้ารหัสแบบ Run-length โดยที่เจ้าของงานวิจัยชิ้นนี้ได้ตั้งข้อสังเกตไว้ว่า ในการสร้างภาพเชิงปริมาตรแบบ ลำแสงขนาน (Parallel Projection) จะต้องนำข้อมูลของปริมาตรต้นฉบับแบบลำดับวัตถุมาพิจารณาทีละเส้น เพื่อจะรวมค่าความทึบแสงไปยังเส้นของข้อมูลภาพผลลัพธ์ขนานกันไป ทำให้จำกัดขอบเขตการพิจารณาข้อมูลเหลือเพียงมิติเดียว และสะดวกในการเข้ารหัสแบบ Run-length มากยิ่งขึ้น ซึ่งจะทำให้เราสามารถข้ามหรือละเลยจุดภาพ (Voxel) ที่มีความทึบแสงสะสมมากกว่าค่าเทรชโชลที่ตั้งไว้ได้ง่ายขึ้น

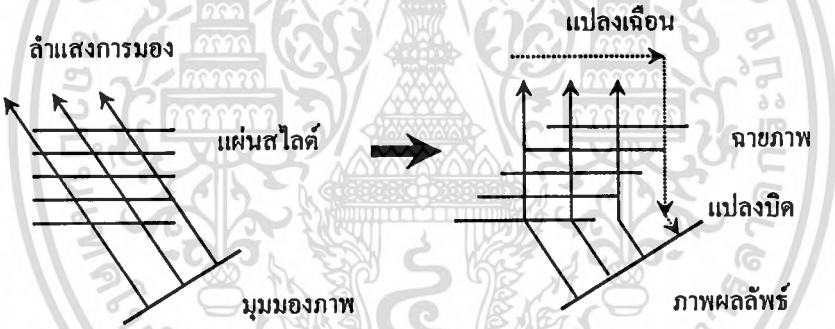
จากที่กล่าวมาทั้งหมดจะเห็นได้ว่าวิธีการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุ สามารถที่จะพิจารณาข้อมูลตลอดทั้งปริมาตร และใช้เทคนิคการเข้ารหัสและโครงสร้างข้อมูลแบบพิเศษ เพื่อละเลยหรือกระโดดข้ามการทำงานในส่วนที่ไม่จำเป็นได้ง่ายกว่าการสร้างภาพแบบลำดับภาพ ส่วนเทคนิคการสิ้นสุดลำแสงก่อนนั้นก็สามารถเพิ่มเข้าไปในวิธีการฉายแสงซึ่งเป็นการสร้างภาพแบบลำดับภาพได้อย่างง่ายดาย ดังนั้นเทคนิคหรือวิธีการสร้างภาพสมัยใหม่จะต้องสามารถใช้ประโยชน์จากทั้งสองเทคนิคที่กล่าวมาทั้งหมดให้ได้ เพื่อให้ความรวดเร็วในการสร้างภาพจะเหลือเวลาน้อยที่สุดเข้าใกล้การประมวลผลแบบเวลาจริง และตอบสนองการทำงานของผู้ใช้ได้อย่างเหมาะสม

การสร้างภาพเชิงปริมาตรที่จะกล่าวถึงในหัวข้อถัดไป เป็นการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุที่ถูกปรับปรุงโดย Lacroute (1995) ซึ่งมีการใช้ทั้งสองเทคนิคที่กล่าวมาและมีการใช้เทคนิคการแบ่งกลุ่มข้อมูลแบบเร็วเข้าร่วมด้วย ทำให้การสร้างภาพด้วยวิธีนี้ สามารถทำได้รวดเร็วกว่าวิธีก่อน ๆ มาก ดังจะได้กล่าวในลำดับต่อไป

4.3 การสร้างภาพเชิงปริมาตรแบบเร็วโดยใช้วิธีการแปลงเฉือนและบิด

จากหัวข้อที่แล้ว การสร้างภาพเชิงปริมาตรแบบลำดับภาพเช่น วิธีการฉายแสง หรือ Ray Casting เป็นวิธีการที่ง่ายในการพัฒนาโปรแกรม และสามารถประยุกต์ใช้เทคนิคการสิ้นสุดลำแสงก่อนได้ง่าย ส่วนการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุนั้น มีประสิทธิภาพสูงในการใช้โครงสร้างข้อมูลแบบพิเศษ เพื่อกระโดดข้ามการทำงานที่ไม่จำเป็นได้ง่าย ซึ่งจะเห็นได้ว่าแต่ละวิธีมีข้อดีแตกต่างกัน แต่ถ้าหากเราสามารถรวมข้อดี หรือความสามารถเหล่านี้เข้าไว้ด้วยกันได้ จะทำให้การสร้างภาพวิธีใหม่ มีความรวดเร็ว ลดเวลาในการทำงานลงได้มากยิ่งขึ้น

วิธีการสร้างภาพเชิงปริมาตรที่จะนำเสนอในหัวข้อต่อไปนี้เป็นการรวมเอาข้อดีของการใช้เทคนิคทั้งสองแบบเข้าด้วยกัน คือ มีความง่าย และยังคงประสิทธิภาพสูงทั้งในการลดงานที่ไม่จำเป็น และการสิ้นสุดลำแสงก่อน โดยเรียกวิธีการนี้ว่า การสร้างภาพเชิงปริมาตร โดยใช้วิธีการแปลงเฉือนและบิด (Shear-warp factorization)



รูปที่ 4.1 หลักการสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิด

จากรูปขั้นตอนในการสร้างภาพเชิงปริมาตรโดยใช้การแปลงเฉือนและบิดนั้นประกอบด้วยขั้นตอนหลัก ๆ ดังนี้

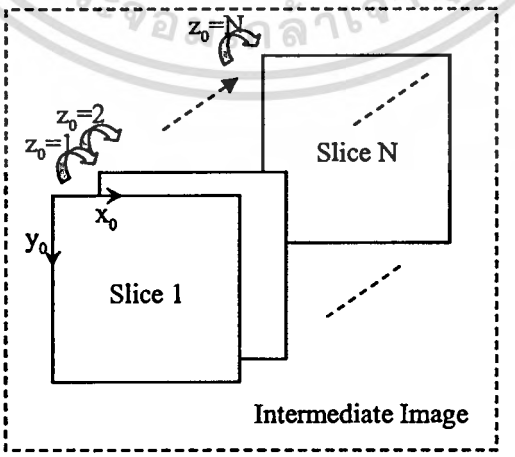
1. ทำการแปลงข้อมูลปริมาตรไปยังพิกัดวัตถุเฉือน โดยการเลื่อนแต่ละสไลซ์ออกไป มีทิศทางในการเลื่อนที่เป็นไปได้อยู่ 3 ทาง ให้เลือกเลื่อนสไลซ์ในแนวที่ตั้งฉากกับทิศทางมุมมองมากที่สุด ซึ่งในการเลื่อนนั้นอาจเลื่อนไปเป็นระยะทางที่ไม่ใช่จำนวนเต็ม ทำให้บางวอกเซลต้องมีการคำนวณค่าเพื่อชดเชยข้อมูลที่หายไปหรือรีแซมปลิง

2. ทำการประกอบแต่ละสไลซ์ที่ถูกรีแซมปลิงเข้าด้วยกัน ทำการประกอบแบบหน้าไปหลังด้วยตัวกระทำ "โอเวอร์" (Over Operator) ขั้นตอนนี้เป็นการฉายปริมาตรในพิกัดวัตถุเฉือนไปเป็นภาพสองมิติที่บิดเบี้ยวหรือภาพระหว่างกลางนั่นเอง (Intermediate Image Plane)
3. แปลงภาพที่บิดเบี้ยวไปยังพิกัดภาพ โดยใช้การแปลงบิด ขั้นตอนนี้เป็นการรีแซมปลิงครั้งที่สองเพื่อสร้างภาพที่ถูกต้อง

เราสามารถเขียนอัลกอริธึม (Algorithm) ของวิธีการแปลงแบบเฉือนและบิดแบบคร่าว ๆ ได้ดังนี้

```
1:  for z_o = 1 to VolumeDept
        for y_i = 1 to ImageHeight      } เป็นการพิจารณาทีละ
            for x_i = 1 to ImageWidth   } แผ่นสไลซ์ (สองมิติ)
                for_each y_o in ResamplingFilter(x_i , y_i )   }
5:                  for_each x_o in ResamplingFilter(x_i , y_i) } เป็นการใช้ Filter แบบสองมิติ หรือ Bi-linear Interpolation
                        add contribution of Voxel[x_o , y_o , z_o] to ImagePixel[x_i , y_i]
```

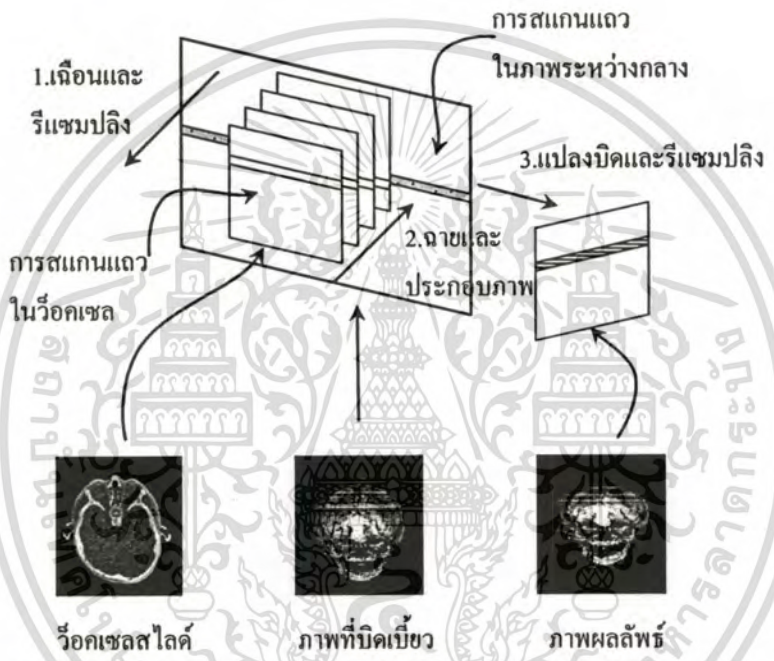
จากอัลกอริธึมด้านบนจะเห็นได้ว่าการแปลงแบบเฉือนและบิดนั้นเป็นการผสมกันระหว่างวิธีลำดับภาพ (วงรอบการทำซ้ำด้านใน บรรทัดที่ 4-5) และวิธีลำดับวัตถุ (วงรอบการทำซ้ำด้านนอก บรรทัดที่ 1-3) โดยบรรทัดที่ 2-3 จะเป็นการนำข้อมูลมาพิจารณาทีละแผ่นสไลซ์ แล้วค่อย ๆ ขยับลึกเข้าไปในก้อนของปริมาตร ตามในบรรทัดที่ 1 ส่วนบรรทัดที่ 4-5 จะเป็นการอินเทอร์โพเลชั่นแบบไบลีเนียร์ และสุดท้ายบรรทัดที่ 6 จะเป็นการฉายจุดภาพที่คำนวณได้ลงบนภาพระหว่างกลางหรือ Intermediate Image ซึ่งขั้นตอนที่กล่าวมาสามารถอธิบายขยายความได้ตามรูปที่ 4.2 ดังนี้



รูปที่ 4.2 การฉายภาพแบบขนานโดยใช้การแปลงเฉือนและบิด

ในขั้นตอนนี้ การนำปริมาตรมาทำการคำนวณทีละสไลซ์เป็นการเข้าถึงข้อมูลปริมาตรตามลำดับข้อมูลที่ถูกเก็บอยู่ในหน่วยความจำหลักหรือฮาร์ดดิสค์ ช่วยลดการเข้าถึงดิสค์ทำให้การอ่านข้อมูลขึ้นมาคำนวณทำได้รวดเร็วขึ้น และที่สำคัญยังช่วยลดการอินเทอร์โพเลชั่นลงได้ เพราะเป็นการคำนวณแบบสองมิติ และในหนึ่งแผ่นสไลซ์จะใช้ค่าน้ำหนักในการอินเทอร์โพเลชั่นเหมือนกันทั้งสไลซ์ ซึ่งแตกต่างกับวิธีการลำดับภาพ ซึ่งจะต้องคำนวณค่าน้ำหนักและคำนวณค่าอินเทอร์โพเลชั่นในทุก ๆ จุดภาพ

จากขั้นตอนที่กล่าวมาทั้งหมดสามารถสรุปขั้นตอนและวิธีการได้ดังรูปที่ 4.3



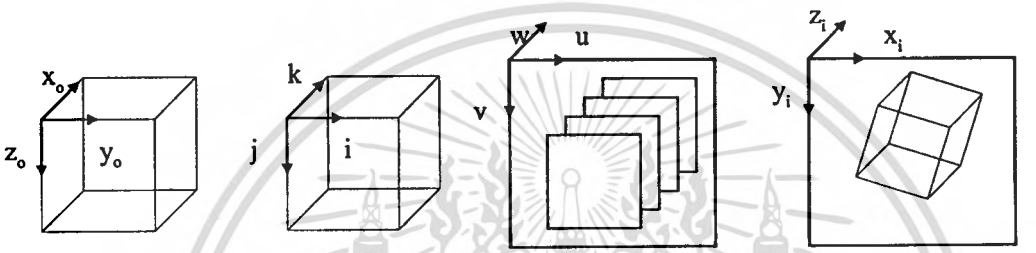
รูปที่ 4.3 แสดงวิธีการสร้างภาพเชิงปริมาตรโดยใช้การแปลงแบบเฉือนและบิด

4.4 หลักการแปลงเฉือนและบิด

ในหัวข้อนี้จะเป็นการกล่าวถึงรายละเอียด และสมการทางคณิตศาสตร์ที่จำเป็นในการสร้างภาพเชิงปริมาตรแบบเฉือนและบิด โดยจะเริ่มกล่าวถึงระบบพิกัดต่าง ๆ ที่เกี่ยวข้อง การหาแกนมุมมองหลัก การแปลงปริมาตรจากพิกัดวัตถุไปยังพิกัดวัตถุมาตรฐาน เมตริกซ์การเฉือนและบิด การฉายภาพไปยังระนาบภาพระหว่างกลาง และสรุปสุดท้ายด้วยเมตริกซ์การแปลงที่สมบูรณ์

4.4.1 ระบบพิกัดที่เกี่ยวข้อง

ระบบพิกัดที่เกี่ยวข้องกับวิธีการสร้างภาพเชิงปริมาตรโดยวิธีการแปลงเฉือนและบิดจะประกอบไปด้วยระบบพิกัด 4 ระบบ ดังแสดงในรูปที่ 4.4 ซึ่งประกอบด้วย พิกัดวัตถุ (Object Space), พิกัดวัตถุมาตรฐาน (Standard Object Coordinate), พิกัดวัตถุเฉือน (Sheared Object Coordinate) หรือบางครั้งเรียกว่าพิกัดภาพระหว่างกลาง (Intermediate Image Coordinate) และพิกัดภาพ (Image Coordinate) ซึ่งทั้งหมดเป็นระบบมือขวา



รูปที่ 4.4 ระบบพิกัดต่าง ๆ ที่ใช้ในการแปลงแบบเฉือนและบิด

ระบบพิกัดวัตถุหมายถึง ระบบพิกัดของปริมาตรโดยปกตินั่นเอง ซึ่งจะมีจุดกำเนิดที่มุมหนึ่งของปริมาตร และระยะทางหนึ่งหน่วยในแต่ละแกนมีค่าเท่ากับความยาวของวอกเซล โดยชื่อของแต่ละแกนถูกกำหนดเป็น x_0 , y_0 และ z_0

เรากำหนดพิกัดวัตถุมาตรฐานโดยการหมุนแกนของระบบพิกัดวัตถุให้แกนมุมมองหลัก (Principal Viewing Axis) เป็นแกนที่สาม ซึ่งแกนหลักนี้ก็คือแกนในระบบพิกัดวัตถุที่ทำมุมขนานกับทิศทางของมุมมองมากที่สุด โดยแกนของระบบพิกัดวัตถุมาตรฐานจะประกอบไปด้วยแกน i , j และ k โดยที่ แกน k คือแกนมุมมองหลัก

พิกัดวัตถุเฉือนกำหนดได้โดยการเฉือนพิกัดวัตถุมาตรฐานด้วยเมตริกซ์การแปลงเฉือน ระบบพิกัดนี้ยังเป็นระบบพิกัดภาพระหว่างกลาง (Intermediate Image) ซึ่งภาพในระบบนี้จะเป็นภาพที่บิดเบี้ยวยังไม่ใช่ภาพผลลัพธ์ที่ถูกต้อง จุดกำเนิดของระบบพิกัดนี้อยู่ที่มุมขอบซ้ายบนของภาพระหว่างกลาง มีชื่อประจำแกนคือ u, v และ w

ระบบพิกัดสุดท้ายเป็นระบบพิกัดของภาพผลลัพธ์ โดยการแปลงระบบพิกัดวัตถุเฉือนด้วยเมตริกซ์การแปลงบิด จุดกำเนิดของพิกัดนี้อยู่ที่มุมซ้ายบนของภาพ มีชื่อแกนทั้งสาม เป็น x_i , y_i และ z_i

เมื่อเราต้องทำการแปลงจากระบบพิกัดหนึ่งไปสู่อีกระบบพิกัดหนึ่ง จะต้องเกี่ยวข้องกับเมตริกซ์การแปลง ซึ่งขั้นตอนทั้งหมดเริ่มจากการแปลงพิกัดวัตถุ ไปเป็นพิกัดมาตรฐาน จากพิกัดมาตรฐานไปเป็นพิกัดวัตถุเฉือน และขั้นตอนสุดท้ายคือแปลงจากพิกัดวัตถุเฉือนไปเป็นพิกัดภาพ เราสามารถแทนขั้นตอนการแปลงพิกัดทั้งหมดเหล่านี้ด้วยสมการที่ 4.1 ดังนี้

$$\begin{bmatrix} x_i \\ y_i \\ z_i \\ w_i \end{bmatrix} = M_{view} \begin{bmatrix} x_o \\ y_o \\ z_o \\ w_o \end{bmatrix} \tag{4.1}$$

โดยที่ M_{view} แทนเมตริกซ์การแปลงทั้งหมด มีขนาดเป็น 4x4 ซึ่งจะอธิบายสรุปในหัวข้อเรื่องสุดท้ายคือเมตริกซ์การแปลงที่สมบูรณ์ ต่อไปจะเป็นการกล่าวถึงวิธีการในการหาแกนมุมมองหลัก

4.4.2 การหาแกนมุมมองหลัก

เรานิยามแกนมุมมองหลัก คือ แกนในพิกัดวัตถุที่วางตัวทำมุมน้อยที่สุดกับเวคเตอร์ของทิศทางการมอง ในพิกัดภาพเวคเตอร์ของมุมมองมีทิศพุ่งเข้า นั่นคือ

$$\vec{v}_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.2}$$

กำหนดให้ $\vec{v}_o$ เป็นเวคเตอร์ทิศทางของมุมมองที่อยู่ในพิกัดวัตถุ จะได้ว่า

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} v_{o,x} \\ v_{o,y} \\ v_{o,z} \end{bmatrix} \tag{4.3}$$

เมื่อ m_{ij} เป็นสมาชิกของเมตริกซ์การแปลงมุมมอง M_{view} ซึ่งจะมีเฉพาะตำแหน่ง 3x3 ของมุมซ้ายบนเพราะ $\vec{v}_i$ และ $\vec{v}_o$ เป็นเวคเตอร์ เพื่อป้องกันการสับสนเราจะเขียนเป็น $M_{view,3x3}$ ซึ่งเป็นเมตริกซ์ย่อยขนาด 3x3

และจากกฎของคาร์เมอร์ (Cramer's Rule) เราสามารถแก้สมการเชิงเส้นได้คือ

$$v_{o,x} = \frac{\begin{vmatrix} 0 & m_{12} & m_{13} \\ 0 & m_{22} & m_{23} \\ 1 & m_{32} & m_{33} \end{vmatrix}}{\left|M_{view,3x3}\right|} = \frac{m_{12}m_{23} - m_{22}m_{13}}{\left|M_{view,3x3}\right|} \tag{4.4}$$

$$v_{o,y} = \frac{\begin{vmatrix} m_{11} & 0 & m_{13} \\ m_{21} & 0 & m_{23} \\ m_{31} & 1 & m_{33} \end{vmatrix}}{\left|M_{view,3x3}\right|} = \frac{m_{21}m_{13} - m_{11}m_{23}}{\left|M_{view,3x3}\right|} \tag{4.5}$$

$$v_{o,z} = \frac{\begin{vmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & 0 \\ m_{31} & m_{32} & 1 \end{vmatrix}}{\left|M_{view,3x3}\right|} = \frac{m_{11}m_{22} - m_{21}m_{12}}{\left|M_{view,3x3}\right|} \tag{4.6}$$

เนื่องจากที่ตัวหารของทั้ง 3 สมการมีค่าเท่ากันเราสามารถตัดตัวหารทิ้งได้ ทำให้ได้

$$\vec{v}_o = \begin{bmatrix} m_{12}m_{23} - m_{22}m_{13} \\ m_{21}m_{13} - m_{11}m_{23} \\ m_{11}m_{22} - m_{21}m_{12} \end{bmatrix} \tag{4.7}$$

พิจารณาค่าโคซายน์ (Cosine) ของมุมระหว่างเวคเตอร์ของมุมมองกับแต่ละแกนในพิกัดวัตถุนั้นขึ้นอยู่กับค่า Dot product ของ $\vec{v}_o$ กับเวคเตอร์หนึ่งหน่วยของแต่ละแกน ค่า Dot product ที่มีค่าสูงสุดเกิดจากมุมนั้น ๆ มีค่าน้อยที่สุด ฉะนั้นเราสามารถหาทิศทางมุมมองหลักได้จาก

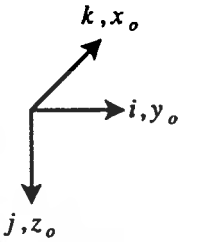
$$c = \max(|v_{o,x}|, |v_{o,y}|, |v_{o,z}|) \tag{4.8}$$

ถ้า $c = |v_{o,x}|$ แล้วจะได้ว่าแกนมุมมองหลักคือ x_o ถ้า $c = |v_{o,y}|$ แล้วจะได้ว่าแกนมุมมองหลักคือ y_o ส่วนกรณีอื่น ๆ แกนมุมมองหลักคือแกน z_o

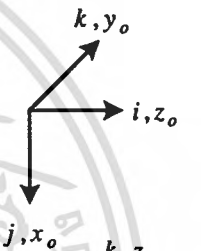
4.4.3 การแปลงไปยังพิกัดวัตถุมาตรฐาน

หลังจากหาแกนมุมมองหลักได้จากวิธีการในหัวข้อที่แล้ว ในขั้นตอนต่อมาจะต้องทำการแปลงปริมาตรในพิกัดวัตถุให้ไปอยู่ในพิกัดมาตรฐานโดยการคูณด้วยเมตริกซ์การแปลง P (Permutation Matrix) ซึ่งค่าของ P นี้จะขึ้นอยู่กับว่าแกนใดของพิกัดวัตถุที่เป็นแกนมุมมองหลัก ดังนี้

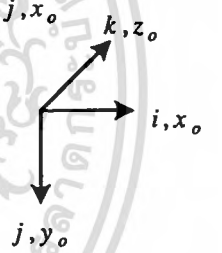
กรณีแกน x_o เป็นแกนมุมมองหลัก $P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$



กรณีแกน y_o เป็นแกนมุมมองหลัก $P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$



กรณีแกน z_o เป็นแกนมุมมองหลัก $P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$



การแปลงจากพิกัดวัตถุไปยังพิกัดวัตถุมาตราฐาน คือ

$$\begin{bmatrix} i \\ j \\ k \\ 0 \end{bmatrix} = P \begin{bmatrix} x_o \\ y_o \\ z_o \\ 0 \end{bmatrix} \tag{4.9}$$

กำหนดให้ M'_{view} เป็นเมตริกซ์ที่ถูกแปลงมุมมอง

$$M'_{view} = M_{view} P^{-1} \tag{4.10}$$

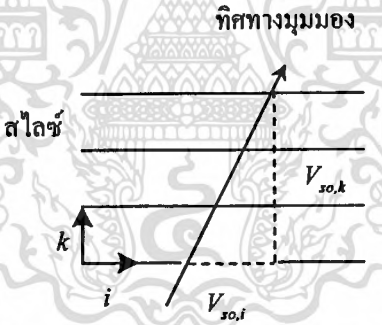
เมตริกซ์นี้ทำการแปลงตำแหน่งจากพิกัดวัตถุไปยังพิกัดวัตถุมาตรฐาน ซึ่งในพิกัดวัตถุมาตรฐานนี้แกน k จะเป็นแกนมุมมองหลักเสมอ

4.4.4 เมตริกซ์การเฉือนและบิด

ในขั้นต่อไปหลังจากที่ทำการแปลงมุมมองด้วยเมตริกซ์ M'_{view} แล้วคือการเฉือนในทิศทางแนวแกน i และ j ด้วย M'_{shear} แล้วกระทำต่อด้วยเมตริกซ์ผูกพันในการบิด M'_{warp} ในการแปลงนั้นต้องคำนึงถึงเงื่อนไขที่ว่า หลังจากแปลงเฉือนแล้วนั้นทิศทางของมุมมองต้องตั้งฉากกับระนาบ (i, j) ในระบบพิกัดวัตถุมาตรฐาน เวคเตอร์ทิศทางมุมมองคือ

$$\vec{v}_{so} = P \cdot \vec{v}_o = \begin{bmatrix} m'_{12}m'_{23} - m'_{22}m'_{13} \\ m'_{21}m'_{13} - m'_{11}m'_{23} \\ m'_{11}m'_{22} - m'_{21}m'_{12} \end{bmatrix} \tag{4.11}$$

เมื่อ m'_{ij} เป็นสมาชิกของเมตริกซ์ที่ผ่านการแปลงมุมมอง M'_{view} เราทำการฉายเวคเตอร์ลงบนระนาบ (i, j) จะมีค่าความชัน (Slope) เท่ากับ $v_{so,i}/v_{so,k}$ ดังรูปที่ 4.5 ซึ่งเราจำเป็นที่จะต้องเลื่อนสไลซ์ในแนวแกน i เพื่อที่จะทำให้ทิศทางของมุมมองตั้งฉากกับระนาบที่ k ใด ๆ โดยเพิ่มค่าความชันเป็นลบเข้าไป



รูปที่ 4.5 การพิจารณาหาค่าสัมประสิทธิ์สำหรับการแปลงเฉือน

เราจะได้ว่า

$$s_i = \frac{-v_{so,i}}{v_{so,k}} = \frac{m'_{22}m'_{13} - m'_{12}m'_{23}}{m'_{11}m'_{22} - m'_{21}m'_{12}} \tag{4.12}$$

และทำนองเดียวกัน

$$s_j = \frac{-v_{so,j}}{v_{so,k}} = \frac{m'_{11}m'_{23} - m'_{21}m'_{13}}{m'_{11}m'_{22} - m'_{21}m'_{12}} \tag{4.13}$$

เราสามารถเขียนสัมประสิทธิ์ของการแปลงแบบเฉือนและบิดในรูปของเมตริกซ์ที่ถูกแปลงมุมมองได้เป็น

$$M'_{view} = M'_{view}\begin{bmatrix}1 & 0 & -s_i & 0\\ 0 & 1 & -s_j & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & s_i & 0\\ 0 & 1 & s_j & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{4.14}$$

$$M'_{view} = \begin{bmatrix}m'_{11} & m'_{12} & (m'_{13} - s_i m'_{11} - s_j m'_{12}) & m'_{14}\\ m'_{21} & m'_{22} & (m'_{23} - s_i m'_{21} - s_j m'_{22}) & m'_{24}\\ m'_{31} & m'_{32} & (m_{33} - s_i m'_{31} - s_j m'_{32}) & m'_{34}\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & s_i & 0\\ 0 & 1 & s_j & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{4.15}$$

จากสมการที่ 4.15 เมตริกซ์ด้านซ้ายคือเมตริกซ์การแปลงบิดและด้านขวาคือเมตริกซ์การแปลงเฉือน ถึงจุดนี้เราสามารถกระจายเมตริกซ์ในการแปลงมุมมองออกมาเป็น 2 เมตริกซ์ โดยในการทำงานเราจะนำเมตริกซ์การเฉือนเข้าไปคูณปริมาตรในพิกัดมาตรฐาน หลังจากนั้นทำการคำนวณการฉายภาพลงบนภาพระหว่างกลางทีละแผ่นสไลซ์ ขั้นตอนสุดท้ายจึงค่อยทำการแปลงบิดภาพระหว่างกลางไปยังพิกัดภาพในสองมิติ เพื่อเป็นภาพผลลัพธ์

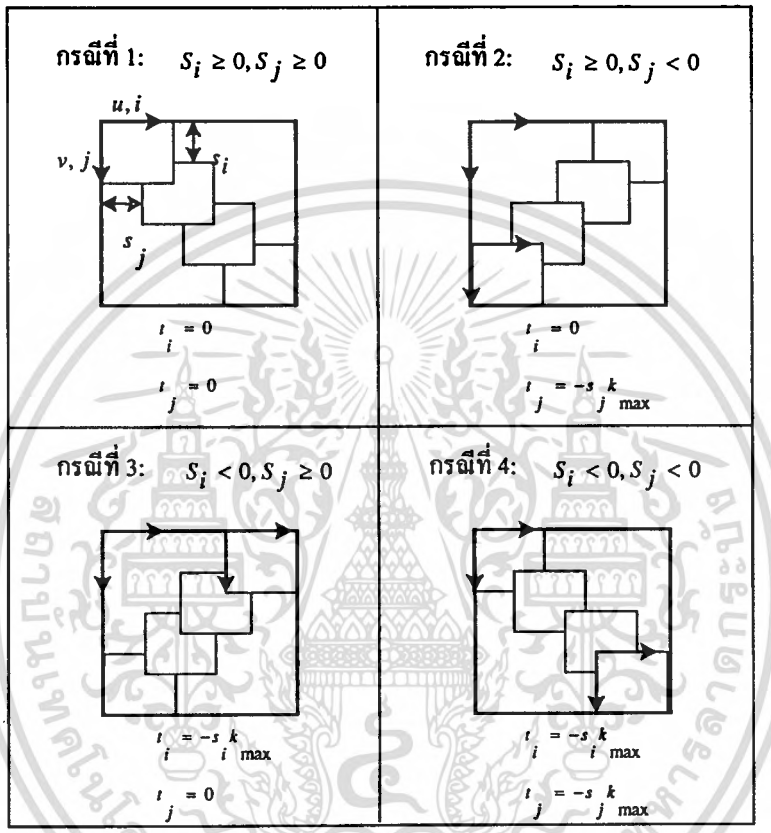
4.4.5 การฉายภาพไปยังระนาบภาพระหว่างกลาง

หลังจากที่เราทำการแปลงเฉือนไปยังพิกัดวัตถุเฉือนแล้ว พิกัดนี้ยังไม่เหมาะสมกับข้อมูลภาพระหว่างกลาง (ภาพที่บิดเบี้ยว) เพราะจุดกำเนิดของพิกัดวัตถุเฉือนไม่ได้อยู่ที่มุมซ้ายบนของภาพระหว่างกลาง เราจึงต้องทำการแปลงย้ายพิกัดวัตถุเฉือนเพื่อให้จุดกำเนิดไปอยู่ที่มุมซ้ายบนของภาพระหว่างกลาง ซึ่งจะเกิดพิกัดของภาพระหว่างกลางขึ้นมา

ดูจากรูปที่ 4.6 แสดงถึงกรณีที่เป็นไปได้ในการแปลงย้ายทั้งหมด 4 วิธีที่อาจเกิดขึ้นในการแปลงย้าย โดยแต่ละกรณีขึ้นอยู่กับเครื่องหมายของสัมประสิทธิ์ในการเฉือนทั้งสองตัว (s_i, s_j) อัลกอริธึมในการสร้างภาพนี้จำเป็นต้องทราบข้อมูลเพิ่มเติมอีกหนึ่งอย่างนั่นก็คือลำดับของสไลซ์ในพิกัดภาพระหว่างกลาง ซึ่งเป็นลำดับการฉายภาพในแนวแกน k ของพิกัดวัตถุมาตรฐาน เพราะเราใช้การประกอบภาพแบบหน้าไปหลัง เมื่อพิจารณาเวคเตอร์ทิศทางของมุมมองหาก $v_{so,k} > 0$ เราจะได้ว่าที่ระนาบ $k = 0$ นั้นเป็นสไลซ์ด้านหน้า ส่วนในกรณีอื่นเราจะได้ว่า $k = k_{max}$ เป็นสไลซ์ด้านหน้า

เมื่อทำการเลือกเมตริกซ์การแปลงย้ายที่เหมาะสมได้แล้ว เราสามารถเขียนเมตริกซ์ในการแปลงเฉือนใหม่ได้เป็น

$$M_{shear} = \begin{bmatrix} 1 & 0 & 0 & t_i \\ 0 & 1 & 0 & t_j \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & s_i & 0 \\ 0 & 1 & s_j & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & s_i & t_i \\ 0 & 1 & s_j & t_j \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.16)$$



รูปที่ 4.6 การแปลงพิกัดภาพระหว่างกลางในกรณีต่าง ๆ

จากสมการที่ 4.15 และสมการที่ 4.16 จะได้ว่าเมตริกซ์การแปลงบิดคือ

$$M_{warp} = \begin{bmatrix} m'_{11} & m'_{12} & (m'_{13} - s_i m'_{11} - s_j m'_{12}) & m'_{14} \\ m'_{21} & m'_{22} & (m'_{23} - s_i m'_{21} - s_j m'_{22}) & m'_{24} \\ m'_{31} & m'_{32} & (m_{33} - s_i m'_{31} - s_j m'_{32}) & m'_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -t_i \\ 0 & 1 & 0 & -t_j \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.17)$$

แต่เนื่องจากการแปลงบิดจะเป็นการแปลงในระบบสองมิติ ดังนั้นจึงสามารถลบในแถวที่ 3 และ หลักที่ 3 ของสมการที่ 4.17 ออกได้เป็น

$$M_{warp2D} = \begin{bmatrix} m'_{11} & m'_{12} & (m'_{14} - t_i m'_{11} - t_j m'_{12}) \\ m'_{21} & m'_{22} & (m'_{23} - t_i m'_{21} - t_j m'_{22}) \\ 0 & 0 & 1 \end{bmatrix} \quad (4.18)$$

ดังนั้นจึงสามารถสรุปได้ว่าในการแปลงภาพที่บิดเบี้ยวจากพิกัดภาพระหว่างกลางไปเป็นภาพผลลัพธ์เราใช้ เมตริกซ์การแปลงบิดกระทำในสองมิติซึ่งมีสมการดังนี้

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = M_{warp2D} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad (4.19)$$

4.4.6 เมตริกซ์การแปลงที่สมบูรณ์

จากสมการที่ 4.1 ที่เคยได้กล่าวไปแล้วนั้น จะเห็นได้ว่าการสร้างภาพเชิงปริมาตร โดยการแปลงเฉือนและบิดมีกระบวนการในการแปลงประกอบด้วย การเฉือนในสามมิติ และการบิดในสองมิติ โดยที่ M_{view} แทนเมตริกซ์การแปลงที่สมบูรณ์ทั้งหมด มีขนาดเป็น 4x4 ประกอบด้วยเมตริกซ์การแปลงแบบต่าง ๆ ดังนี้

$$M_{view} = M_{warp} M_{shear} P \quad (4.20)$$

กระบวนการทำงานโดยสรุปตามลำดับมีการทำงานตามลำดับดังนี้

1. หาแกนมุมมองหลัก (สมการที่ 4.7 และ 4.8) จากนั้นทำการเลือกเมตริกซ์ P
2. คำนวณหาเมตริกซ์ที่ถูกแปลงมุมมอง จาก M_{view} และ P (สมการที่ 4.10)
3. คำนวณหาสัมประสิทธิ์การเฉือน (สมการที่ 4.12 และ 4.13) จากเมตริกซ์ที่ถูกแปลงมุมมอง M'_{view}
4. ทำการคำนวณการย้ายตำแหน่งจากพิกัดวัตถุมาตรฐานไปยังพิกัดภาพระหว่างกลาง (รูปที่ 4.6)
5. คำนวณเมตริกซ์การเฉือน (สมการที่ 4.16) และเมตริกซ์การแปลงบิด (สมการที่ 4.18)

ที่กล่าวมาทั้งหมดในหัวข้อนี้ทำให้เราทราบถึงเมตริกซ์การแปลงที่จำเป็นต้องใช้ในวิธีการสร้างภาพเชิงปริมาตร โดยการแปลงเฉือนและบิด ส่วนในหัวข้อต่อไปจะได้กล่าวถึงการฉายภาพลงบนภาพระหว่างกลาง (Intermediate Image) ในพิกัดภาพวัตถุเฉือนโดยละเอียด

4.5 การประกอบภาพไปยังภาพระหว่างกลาง

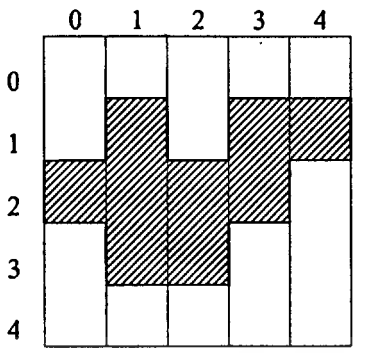
ในหัวข้อที่ผ่านมาเป็นการกล่าวถึงขั้นตอนโดยรวมทั้งหมด แต่ยังไม่ได้กล่าวละเอียดนักในหัวเรื่องเกี่ยวกับการฉายภาพลงบนภาพระหว่างกลาง ก่อนที่จะแปลงบิดกลับมาเป็นภาพที่สมบูรณ์ ซึ่งในขั้นตอนการฉายภาพนี้ มีอยู่ด้วยกันหลายขั้นตอน จึงแยกออกมากล่าวโดยละเอียดในหัวข้อนี้

ระหว่างทำการฉายภาพลงไปบนภาพระหว่างกลางนั้นเพื่อให้สามารถลดการทำงานที่ไม่จำเป็นลง จึงได้มีการนำเอาการเข้ารหัสแบบรันเลงท์ (Run-length) มาใช้ร่วมด้วย มีผลให้ขั้นตอนอื่น ๆ ที่ตามมาในขั้นตอนการประกอบภาพคำนวณลดน้อยลง ทำให้เวลาที่ถูกใช้ไปในขั้นตอนทั้งหมดลดลงเช่นกัน การฉายภาพจะประกอบด้วยขั้นตอนเหล่านี้คือ การจัดหมวดหมู่ของชุดข้อมูลหรือแบ่งกลุ่มข้อมูล (Data Classification) เพื่อกำหนดค่าความทึบแสงให้แก่กลุ่มของข้อมูลได้อย่างถูกต้อง, การอินเทอร์โพเลชั่นหาค่าของข้อมูลที่จุดแซมปลิง (Sampling Point) หลังจากมีการแปลงแบบเฉือนและเลื่อนสไลซ์แล้ว ต่อจากนั้นจะต้องมีการกำหนดค่าสีให้แก่จุดข้อมูลที่ทำการอินเทอร์โพเลชั่น ด้วยวิธีค้นหาค่าสีจากการเปิดตาราง เมื่อได้ค่าสีของจุดแซมปลิงแล้วจึงทำการฉายภาพลงไปบนภาพระหว่างกลางต่อไป

4.5.1 การประกอบภาพแบบเร็วโดยใช้การเข้ารหัสแบบรันเลงท์ (Run-length)

การเข้ารหัสแบบรันเลงท์นั้นจะเป็นการบอกถึงความเกี่ยวข้องของชุดข้อมูลที่อยู่ติดกัน โดยจะมองข้อมูลเป็นลักษณะของไบนารี คือมีเพียงค่า “0” หรือ “1” เท่านั้น เมื่อทำการเข้ารหัสจะสนใจเฉพาะข้อมูลที่ต้องการจริง ๆ เท่านั้น การเข้ารหัสรันเลงท์มีวัตถุประสงค์เพื่อต้องการที่จะลดขนาดของข้อมูลให้เหลือน้อยที่สุดโดยไม่เข้าไปยุ่งกับค่าของข้อมูลทำให้คุณภาพของข้อมูลไม่มีการเปลี่ยนแปลง ข้อดีของการเข้ารหัสแบบนี้คือจะไม่พิจารณาถึงความซับซ้อนของบริเวณ แต่จะแสดงในรูปแบบข้อมูลแถวเดียว ส่วนข้อเสียคือการเข้ารหัสแบบนี้ไม่สามารถระบุถึงเส้นขอบของบริเวณได้ ทำให้ไม่สามารถแบ่งแยกบริเวณหลาย ๆ บริเวณในรูปเดียวกันได้

ในการนำเอารหัสรันเลงท์มาใช้ในงานวิจัยนี้ จะพิจารณาข้อมูลคล้ายกับหลักการของไบนารีคือจะพิจารณาที่ค่าความทึบแสงของว็อคเซล ถ้าหากว็อกเซลใดมีค่าความทึบแสงจะถูกเข้ารหัสไว้ ส่วนว็อคเซลที่โปร่งแสง หรือค่าความทึบแสงต่ำกว่าค่าโทรชโชลที่ตั้งไว้ก็จะถูกละเลยไป

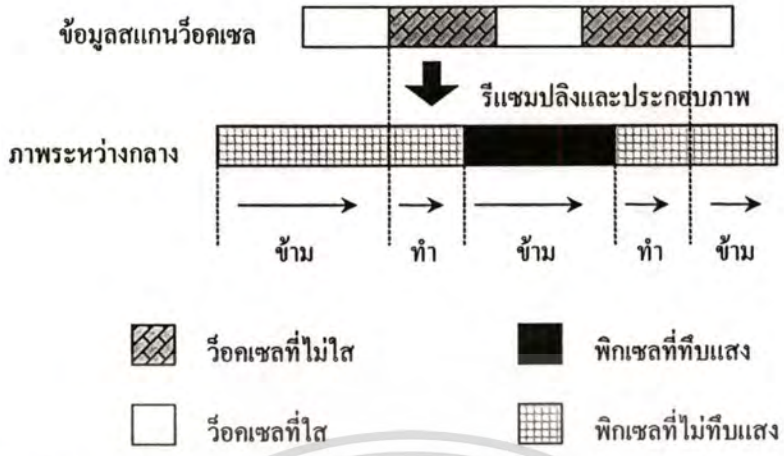


รหัสรันเลงท์	Index
(1, 1) 1, (1, 3) 2	1, 2
(2, 0)	3
(3, 1)	4
.	.
.	.

รูปที่ 4.7 รหัสรันเลงท์ของข้อมูลแบบไบนารี

จากรูปที่ 4.7 ข้อมูลขนาด 5x5 จะถูกแทนด้วยรหัสรันเลงท์ ทั้งหมด 4 ชุด โดยทำการเก็บตำแหน่งเริ่มต้นของข้อมูลที่สนใจแล้วตามด้วยจำนวนของจุดข้อมูลที่สนใจที่อยู่ติดกันทั้งหมด ตัวอย่างเช่น (2, 0) 4 คือจากตำแหน่งที่ (2, 0) มีข้อมูลติดกันในแนวนอน 4 จุดข้อมูล เป็นต้น

ในหัวข้อที่ 4.2 ที่เราได้กล่าวไปแล้วเกี่ยวกับเทคนิคการเพิ่มความเร็วในการสร้างภาพเชิงปริมาตรนั้น และจากรูปที่ 4.3 ที่ได้แสดงขั้นตอนการสร้างภาพเชิงปริมาตรโดยวิธีการแปลงเฉือนและบิดทั้งหมด จะเห็นได้ว่าเมื่อจะทำการประกอบภาพของแผ่นสไลซ์ลงไปบนภาพระหว่างกลาง จะทำการสแกน (scan) วีอคเซลของสไลซ์ขนานไปกับแถวของจุดข้อมูล (pixel) ของภาพระหว่างกลาง เป็นการนำเอาข้อมูลมาพิจารณาทีละหนึ่งมิติ จากการกระทำดังกล่าวนี้ทำให้เราสามารถนำเอาเทคนิคของโครงสร้างข้อมูลแบบพิเศษด้วยการเข้ารหัสรันเลงท์ มาช่วยในการประกอบภาพได้ เพื่อให้สามารถละเลยหรือข้ามงานที่ไม่จำเป็น และในขั้นตอนนี้ยังสามารถใช้เทคนิคการสิ้นสุดลำแสงก่อน(Early Ray Termination)ได้อีกด้วย กล่าวคือจะหยุดประกอบภาพในตำแหน่งที่พิกเซลของภาพระหว่างกลางมีค่าความทึบแสงเข้าใกล้หนึ่ง ซึ่งสามารถอธิบายได้ดังรูป



รูปที่ 4.8 ขั้นตอนการประกอบภาพโดยใช้การเข้ารหัสรันเลงท์และการสิ้นสุดลำแสงก่อน

การทำงานในรูปที่ 4.8 นั้นจะเริ่มทำงานไปตามรหัสรันเลงท์โดยจะทำเฉพาะว็อคเซลที่ไม่ใส (ส่วนที่ใสคือพื้นหลังและอากาศ) จากนั้นจะพิจารณาว่าภาพที่จะทำการประกอบนั้นทึบแสงหรือไม่หากทึบแสงก็จะทำการข้ามส่วนนั้นอีก ทำให้อัลกอริธึมมีประสิทธิภาพสูงเพราะจะใช้เวลาการคำนวณเฉพาะในส่วนที่มีผลต่อภาพผลลัพธ์เท่านั้น

4.5.2 การแบ่งกลุ่มข้อมูล

การสร้างภาพในแบบลำดับวัตถุในวิทยานิพนธ์นี้ได้ใช้การแบ่งกลุ่มข้อมูลเหมือนกับที่กล่าวไว้ในขั้นตอนของการสร้างภาพเชิงปริมาตรแบบลำดับภาพ คือ ตามสมการของ Levoy เนื่องจากเป็นวิธีการที่ง่ายและมีประสิทธิภาพเพียงพอที่จะนำมาแบ่งกลุ่มของข้อมูลต้นฉบับ Lacroute ได้เสนอวิธีการแบ่งกลุ่มข้อมูลแบบเร็วโดยการใช้โครงสร้างข้อมูลแบบออคทรีไว้ด้วย แต่เนื่องจากการใช้โครงสร้างข้อมูลแบบนี้ต้องใช้หน่วยความจำเป็นจำนวนมาก และข้อมูลปริมาตรต้นฉบับที่ใช้ในงานวิจัยนี้ก็มีขนาดใหญ่มากด้วย ซึ่งถ้าหากคำนวณด้วยวิธีการของโครงสร้างข้อมูลแบบออคทรีจะต้องสิ้นเปลืองเวลามากกว่า เพราะคอมพิวเตอร์ต้องใช้หน่วยความจำเสมือนบนฮาร์ดดิสก์ (Hard disk) ซึ่งสามารถเข้าถึงข้อมูลได้ช้ากว่าหน่วยความจำชนิดชั่วคราว (RAM) มาก การแบ่งกลุ่มข้อมูลด้วยโครงสร้างข้อมูลแบบนี้จึงไม่ถูกนำมาใช้

ในการแบ่งกลุ่มข้อมูลเพื่อคำนวณหาค่าความทึบแสง ได้ใช้วิธีการประมวลผลแบบคำนวณล่วงหน้า (Pre-Processing) โดยได้คำนวณหาค่าเกรเดียนท์ และค่านอร์มอลเว็คเตอร์ไว้แล้ว โดยนำปริมาตรต้นฉบับมาคำนวณและจัดเก็บใหม่ตามโครงสร้างนี้คือ จัดเก็บค่าระดับเทาของว็อกเซล (1 Byte) ซึ่งจะถูกนำไปใช้ในการถ่วงน้ำหนักในการให้สีโดยวิธีค้นหาค่าจากตาราง, ขนาดของเกรเดียนท์

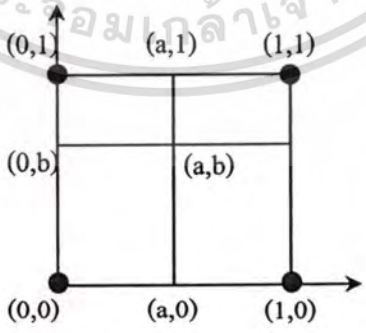
(1 Byte) จะถูกนำไปใช้เมื่อต้องการหาขอบของภาพในปริมาตร และค่านอร์มอลเว็คเตอร์ (2 Byte) จะถูกนำไปใช้เป็นดัชนีในการค้นหาค่าสีจากตาราง ซึ่งจะเห็นได้ว่าข้อมูลวอ็อคเซลของปริมาตรเริ่มต้นหนึ่งไบต์ เมื่อจะทำการสร้างภาพเชิงปริมาตรต้องใช้หน่วยความจำ 4 ไบต์เพื่อเก็บค่าต่าง ๆ ตามที่กล่าวมา ดังนั้นหากปริมาตรเริ่มต้นมีขนาด 512^3 จะต้องใช้หน่วยความจำสูงถึง 512 Mbytes ทีเดียวในการจัดเก็บข้อมูลในระหว่างที่ทำการคำนวณ ซึ่งถ้าหากนำเอาโครงสร้างข้อมูลแบบออคทรีมาใช้ร่วมด้วย จะต้องใช้หน่วยความจำในระหว่างที่ทำการคำนวณอีกเป็นจำนวนมาก ซึ่งจะเห็นได้ว่าไม่เหมาะกับการใช้งานจริงกับปริมาตรขนาดใหญ่

4.5.3 การอินเทอร์โพเลชั่น

หลังจากทำการแปลงเฉือนและเลื่อนสไลซ์แล้วนั้น จะต้องทำการฉายภาพลงไปบนภาพระหว่างกลาง แต่เนื่องจากการคูณด้วยเมตริกซ์การเฉือนทำให้มีพิกัดที่ไม่ใช่จำนวนเต็มอยู่ เราจึงต้องทำการอินเทอร์โพเลชั่นข้อมูล ก่อนที่จะฉายลงไปบนภาพระหว่างกลาง การอินเทอร์โพเลชั่นสามารถทำได้ง่าย ๆ ด้วยวิธีการแบบไบลีเนียร์ (Bi-Linear) ซึ่งอาศัยข้อมูลเพียงสี่จุดรอบข้างเท่านั้น สมการของการอินเทอร์โพเลชั่นแบบเชิงเส้น (Linear) มีดังนี้

$$f(x_d) = \left(\frac{f(x_1) - f(x_0)}{x_1 - x_0} \right) (x_d - x_0) + f(x_0) \tag{4.21}$$

โดยจะต้องทำการคำนวณหาค่าทั้งหมดสามครั้ง คือในแกน x สองครั้ง และในแกน y อีกหนึ่งครั้ง สามารถเขียนเป็นวิธีการได้ดังนี้



สมมติค่าความยาวระหว่างจุดต่อจุดเท่ากับหนึ่งหน่วย และ *f(x)* คือค่าของจุดข้อมูลใดๆ ทำการอินเทอร์โพเลชั่นในแนวแกน *x* สองครั้งจะได้ว่า

$$f_{a,0} = (f_{1,0} - f_{0,0})a + f_{0,0}$$
$$= a f_{1,0} - a f_{0,0} + f_{0,0}$$
$$= (1 - a) f_{0,0} + a f_{1,0}$$

ในทำนองเดียวกัน

$$f_{a,1} = (1 - a) f_{0,1} + a f_{1,1}$$

และในแกน y

$$f_{a,b} = (1 - b) f_{a,0} + b f_{a,1}$$
$$= (1 - a)(1 - b) f_{0,0} + a(1 - b) f_{1,0} + b(1 - a) f_{0,1} + ab f_{1,0}$$
$$= A_1 f_{0,0} + A_2 f_{1,0} + A_3 f_{0,1} + A_4 f_{1,1}$$

โดยที่ $A_1 = (1 - a)(1 - b)$

$A_2 = a(1 - b)$

$A_3 = b(1 - a)$

$A_4 = ab$

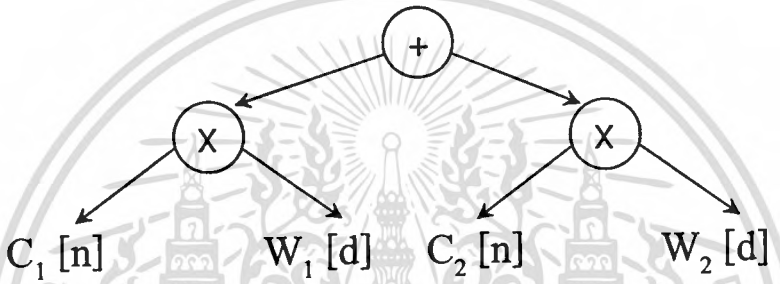
จากชุดสมการคณิตศาสตร์ที่ผ่านมา A ใด ๆ คือ ค่าสัมประสิทธิ์ที่นำไปคูณกับค่าจุดข้อมูลเพื่อใช้ในการอินเทอร์โพเลชั่นแบบไบลีเนียร์ ซึ่งจะเห็นได้ว่าการแปลงเฉือนทำให้การอินเทอร์โพเลชั่นสามารถทำได้สะดวกมากขึ้นเพราะค่าสัมประสิทธิ์ A จะเหมือนกันทั้งแผ่นสไลซ์ทำให้การคำนวณสามารถทำได้รวดเร็วขึ้น อธิบายได้ดังรูป

รูปที่ 4.9 แสดงการอินเทอร์โพเลชั่นแบบไบลีเนียร์ในการฉายภาพ

ทุกจุดที่ทำการแซมปลิงจะอยู่ห่างจากสี่จุดรอบข้างด้วยระยะทางที่เท่ากัน เพราะการแปลงแบบเฉือนจะทำการเลื่อนสไลต์มาอยู่ในตำแหน่งที่เหมาะสมแล้วนั่นเอง จากรูปจะเห็นได้ว่าการแปลงแบบเฉือนนี้มีประโยชน์ในการอินเทอร์โพเลชั่นเป็นอย่างมาก

4.5.4 การค้นหาค่าสี และแสงเงาจากตาราง

เมื่อได้ทำการอินเทอร์โพเลชั่นหาค่าของจุดที่สนใจหรือจุดแซมปลิงแล้วนั้น จะต้องทำการกำหนดค่าสีให้แก่จุดข้อมูลนั้น ๆ ก่อนที่จะทำการฉายลงไปบนภาพระหว่างกลาง โดยในขั้นตอนนี้สามารถกำหนดทรานเฟอร์ฟังก์ชั่นของการให้สี (Transfer Function) ได้หลายแบบ หลายวิธี แต่วิธีที่จะนำเสนอต่อไปนี้เป็นวิธีที่ง่ายต่อการนำไปใช้งาน อีกทั้งมีความรวดเร็วในการคำนวณอีกด้วย นั่นก็คือการค้นหาค่าสีจากตารางหรือ Look up Table โดยในขั้นตอนนี้จะต้องทำการคำนวณค่าสีใส่ไว้ในตารางล่วงหน้า หลังจากนั้นจึงค่อยทำการกำหนดค่าสีให้แก่ข้อมูลจุดนั้นตามวิธีการดังรูป



รูปที่ 4.10 วิธีการกำหนดค่าสีโดยค้นหาจากตาราง

จากรูปจะเป็นการกำหนดค่าสีให้แก่จุดข้อมูลโดยยกตัวอย่างชนิดของข้อมูล (Material type) สองชนิด ซึ่งชนิดของกลุ่มข้อมูลนี้เราสามารถกำหนดได้หลายชนิด เช่น กระดูก เนื้อเยื่อ ไขมัน อากาศ เป็นต้น การกำหนดค่าสีจะเริ่มจากการนำเอาค่าสีในตารางที่ถูกชี้โดยค่านอร์มอลเว็คเตอร์ที่จุดนั้น ๆ มาทำการถ่วงน้ำหนักด้วยค่าการถ่วงน้ำหนักที่ถูกชี้โดยค่าระดับเทาของจุดข้อมูล ก็จะได้ค่าสีที่จุดนั้นออกมา เมื่อทำการประกอบภาพไปบนภาพระหว่างกลางก็เพียงแต่นำค่าสีไปรวมกับค่าสีของจุดข้อมูลในตำแหน่งที่ตรงกันของสไลต์ถัดไปเท่านั้น

ที่กล่าวมาเป็นการให้ค่าสีเพียงหนึ่งสีเท่านั้น ถ้าหากต้องการให้ภาพผลลัพธ์เป็นแบบ RGB แล้วก็ต้องทำการคำนวณค่าสีแยกกันอิสระทั้งหมดสามครั้ง โดยสร้างตารางค่าสีของแต่ละสีขึ้นมาและทำตามขั้นตอนการค้นหาจากตาราง สุดท้ายก็จะได้ภาพผลลัพธ์เป็นสีตามที่ต้องการ

ค่าในตารางการถ่วงน้ำหนักนั้น สามารถกำหนดได้โดยตรงจากผู้ใช้ ส่วนตารางค่าสีจะต้องมีการคำนวณล่วงหน้าโดยใช้สมการการให้แสงและเงาของ Phong illumination model (Lacroute, 1995:156) ดังนี้

$$L = L_a + \sum_i L_i k_d (\bar{n} \cdot \bar{l}_i) + L_i k_s (\bar{n} \cdot \bar{h}_i)^s \qquad (4.22)$$

โดยที่

L คือค่าสีหรือค่าการส่องสว่างของวัตถุแต่ละชนิด มีค่าเท่ากับ C_i ในสมการที่ 2.5 ในหัวข้อเรื่องการประกอบภาพ

$\bar{n}$ คือ นอร์มอลเว็คเตอร์เชิงพื้นผิวของจุดที่สนใจ

$\bar{l}_i$ คือ เวคเตอร์หนึ่งหน่วยที่มีทิศทางไปยังแหล่งกำเนิดแสง

$\bar{h}_i$ คือ เวคเตอร์หนึ่งหน่วยที่หาค่าได้จากสมการ

$\bar{h}_i = \frac{\bar{l}_i + \bar{v}}{|\bar{l}_i + \bar{v}|}$ โดยที่ $\bar{v}$ เป็นเว็คเตอร์หนึ่งหน่วยที่มีทิศทางไปยังผู้สังเกตุ

L_i คือ ค่าสีของแหล่งกำเนิดแสง i ซึ่งมีได้หลายแหล่งกำเนิดแสง

L_a คือ ค่าการส่องสว่างจากแสงแวดล้อม

k_d คือ สัมประสิทธิ์การสะท้อนแสงแบบกระจายของวัตถุชนิด d

k_s คือ สัมประสิทธิ์การสะท้อนแสงแบบกระจก

s คือ ส่วนประกอบการยกกำลังของการสะท้อนแบบกระจก

จากสมการที่ 4.22 เราทำการตั้งสมมติฐานว่าระยะห่างของจุดกำเนิดแสงกับวัตถุอยู่ที่ระยะอนันต์ เพราะฉะนั้นทำให้เวคเตอร์ $\bar{l}_i$ มีค่าคงที่กับทุก ๆ ว็อคเซล สำหรับการฉายภาพมุมมองขนาน ผู้สังเกตก็อยู่ห่างจากภาพเป็นระยะอนันต์เช่นกัน ทำให้ค่า $\bar{v}$ และ $\bar{h}_i$ มีค่าคงที่เช่นกัน จากสมมติฐานนี้ทำให้ค่าสีหรือค่าการส่องสว่างของวัตถุแต่ละชนิด (L) ขึ้นอยู่กับเพียงปัจจัยเดียวเท่านั้นคือ ค่านอร์มอลเว็คเตอร์เชิงพื้นผิวของจุดที่สนใจ ดังนั้นตารางค่าสีหรือ Look up Table จึงสามารถสร้างขึ้นมาได้โดยมีเพียงค่านอร์มอลเว็คเตอร์เป็นตัวชี้ระบุค่าข้อมูล (Index)

นอร์มอลเว็คเตอร์ในภาพเชิงปริมาตรมีนิยาม คือ เว็คเตอร์หนึ่งหน่วยที่ขนานไปกับเกรเดียนท์ที่คำนวณโดยใช้ข้อมูลระดับเทา ถ้ากำหนดให้ d (x, y, z) แทนเกรเดียนท์ และ n (x, y, z) แทนนอร์มอลเว็คเตอร์ จะได้ว่า

$$n(x, y, z) = \frac{\nabla d(x, y, z)}{|\nabla d(x, y, z)|} \tag{4.23}$$

เกรเดียนท์สามารถคำนวณโดยการใช้ Central difference gradient operator ได้ดังนี้

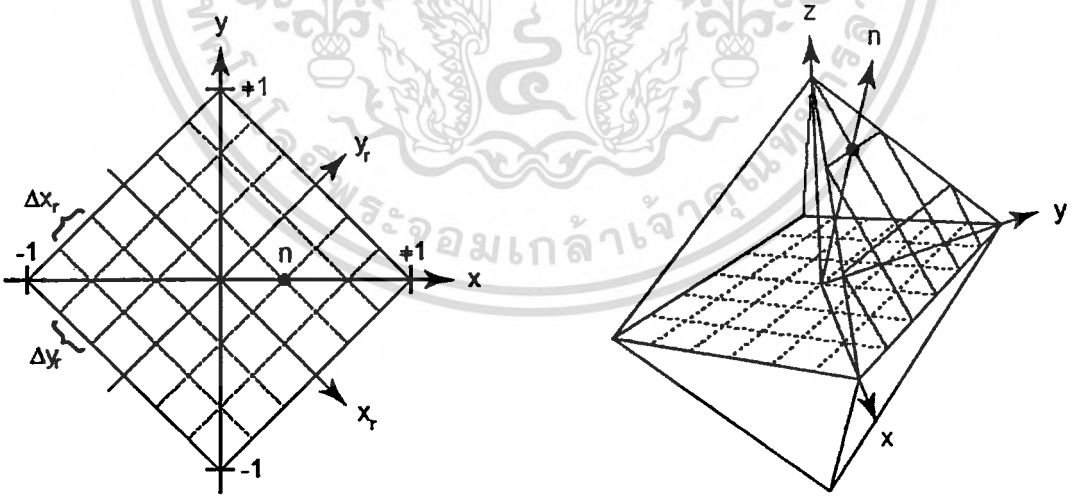
$$\nabla d(x,y,z) = \frac{1}{2}[d(x+1,y,z) - d(x-1,y,z)]\bar{i} + \frac{1}{2}[d(x,y+1,z) - d(x,y-1,z)]\bar{j} + \frac{1}{2}[d(x,y,z+1) - d(x,y,z-1)]\bar{k} \quad (4.24)$$

ซึ่งค่านอร์มอลเว็คเตอร์นี้จะถูกคำนวณล่วงหน้าและเก็บไว้ตั้งแต่เมื่อทำการโหลดเอาข้อมูลเชิงปริมาตรเข้ามาในหน่วยความจำ แต่แทนที่จะเก็บข้อมูลที่เป็นเลขทศนิยมจำนวนสามตัว เพื่อเป็นการประหยัดหน่วยความจำ ดังนั้นจึงใช้การเข้ารหัสซึ่งนำเสนอโดย Fletcher and Robertson (1993) เพื่อแทนค่าตัวเลขเหล่านี้ด้วยจำนวนเต็มค่าหนึ่งทำให้ง่ายต่อการนำไปใช้ค้นหาค่าสีจากตารางมากยิ่งขึ้น

ก่อนอื่นทำการนิยามว่าการเข้ารหัสเว็คเตอร์แบบหนึ่งหน่วยต้องมีคุณสมบัติดังนี้คือ

$$|n_x| + |n_y| + |n_z| = 1 \quad (4.25)$$

เมื่อ n_x, n_y, n_z คือค่าของเว็คเตอร์ในแนวแกน x, y, z และจำเป็นต้องมีจุดเริ่มต้นอยู่ที่จุดกำเนิดของกริดที่มีการหมุนไป 45° ดังแสดงในรูปที่ 4.11



รูปที่ 4.11 ตารางกริดและพีระมิดแปดหน้าที่ถูกใช้เพื่อการเข้ารหัสนอร์มอลเว็คเตอร์

จากรูป ตารางกริดด้านซ้ายมือเป็นฐานของพีระมิดแปดหน้าด้านขวามือโดยที่นอร์มอลเว็คเตอร์ n มีจุดเริ่มต้นที่จุดกำเนิดของพีระมิด และมีองค์ประกอบในแนวแกนต่าง ๆ เป็น n_x, n_y, n_z ซึ่งมีคุณสมบัติตามสมการ 4.25 ที่ได้นิยามไปก่อนหน้านี้ โดยสามารถหาค่าต่าง ๆ ได้ตามสมการ

$$n_x = \frac{y_r + x_r}{\sqrt{2}}$$

$$n_x = \frac{y_r - x_r}{\sqrt{2}} \qquad (4.26)$$

$$n_x = \pm(1 - \frac{|y_r + x_r| + |y_r - x_r|}{\sqrt{2}})$$

เมื่อ $x_r \equiv x_i \nabla x_r$

$y_r \equiv y_i \nabla y_r$

จากสมการจะได้ว่าการเข้ารหัสนั้นเพียงแค่เก็บค่าของ n_x , n_y และค่าบอกทิศทางว่าเป็นบวกหรือลบของแกน z เท่านี้ก็เพียงพอในการที่จะถอดรหัสกลับมาเว็คเตอร์ค่าเดิมได้ Lacroute ได้เสนอไว้ว่าข้อมูล 13 บิต เพียงพอที่จะเก็บค่าการเข้ารหัสของเว็คเตอร์หนึ่งเว็คเตอร์ได้ โดยแบ่งเป็นค่าของ n_x จำนวนหกบิต ค่าของ n_y หกบิต และค่าทิศทางของแกน z อีกหนึ่งบิต เพราะฉะนั้นตารางค่าสีที่มีนอร์มอลเว็คเตอร์เป็น Index นี้จะมีจำนวนสมาชิกได้ทั้งหมด 2^{13} หรือ 8,192 สมาชิก และต้องคำนวณค่าสีเก็บไว้ทั้งหมด แต่อย่างไรก็ตามเวลาที่สูญเสียไปในการคำนวณค่าในตารางถือว่าน้อยมากเมื่อเปรียบเทียบกับการคำนวณการให้สีแบบเดิมแก่ทุก ๆ จุดในปริมาตร

4.6 คุณสมบัติที่สำคัญของวิธีการสร้างภาพแบบลำดับวัตถุ

ในหัวข้อนี้จะเป็นการกล่าวถึงข้อดีของการสร้างภาพเชิงปริมาตรด้วยวิธีการแปลงเฉือนและบิด ที่ทำให้การสร้างภาพด้วยวิธีนี้สามารถลดความยุ่งยากในการคำนวณ และทำให้เวลาที่ใช้ลดลงด้วยคุณสมบัติที่สำคัญ ๆ มีดังนี้

1. การแปลงเฉือนทำให้น้ำหนักในการอินเทอร์โพเลชั่นมีค่าเท่ากันทุกจุดในสไลด์เดียวกัน ทำให้ไม่ต้องคำนวณค่าน้ำหนักใหม่ทุกจุด (ดังแสดงในรูปที่ 4.9)
2. มีการเข้ารหัสด้วยรหัสรันเลงท์ในขั้นตอนของการประกอบภาพ ทำให้ลดการทำงานที่ไม่จำเป็นลงได้มาก
3. ในขั้นตอนการประกอบภาพสามารถทำได้ง่าย เพราะเส้นสแกนของปริมาตรกับภาพระหว่างกลางอยู่ในแนวขนานกัน

4. มีการใช้เทคนิคการสิ้นสุดลำแสงก่อนโดยทำการตรวจสอบความเข้มแสงสะสมที่เกิดบนภาพระหว่างกลางในขณะทำการประกอบภาพ ทำให้ไม่ต้องทำงานในจุดที่มีความทึบแสงอย่างสมบูรณ์แล้ว (ค่าความทึบแสงที่จุดนั้นเท่ากับหนึ่ง)
5. การกำหนดค่าสีสามารถทำได้จากการเปิดตาราง ไม่จำเป็นต้องคำนวณให้ค่าสีในทุก ๆ จุด ทำให้ลดเวลาในการคำนวณลง

4.7 สรุป

ในบทนี้เป็นการกล่าวถึงการสร้างภาพเชิงปริมาตรด้วยวิธีการแปลงเฉือนและบิดทั้งหมด โดยชี้ให้เห็นเทคนิคต่าง ๆ ที่ถูกนำมาใช้เพิ่มความเร็วในการคำนวณของการสร้างภาพเชิงปริมาตรแบบใหม่ ๆ หลังจากนั้นได้กล่าวถึงวิธีการแปลงเฉือนและบิด และเมตริกซ์การแปลงที่เกี่ยวข้อง และกล่าวถึงในรายละเอียดของขั้นตอนและวิธีการของการประกอบภาพด้วยเทคนิคต่าง ๆ คือ การลดงานที่ไม่จำเป็นโดยการเข้ารหัสแบบรันเลงท์ การอินเทอร์โพเลชั่นแบบไบลีเนียร์ การให้สีโดยการค้นหาค่าจากตาราง และบอกถึงวิธีการในการเข้ารหัสเว็คเตอร์เพื่อใช้เป็นดัชนีในการค้นหาค่าสีในตารางอีกด้วย

จากที่กล่าวมาทั้งหมดในบทนี้จะเห็นได้ว่าขั้นตอนการสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิดนี้ สามารถทำให้การคำนวณลดความยุ่งยากลง อีกทั้งมีขั้นตอนที่ช่วยให้ลดเวลาในการคำนวณลงได้เป็นอย่างดี

บทที่ 5
โครงสร้างของคอมพิวเตอร์ขนานแบบคลัสเตอร์

5.1 บทนำ

ในบทนี้จะเป็นการกล่าวถึงรายละเอียดกว้าง ๆ ของระบบการประมวลผลแบบขนาน ที่เรียกว่า คอมพิวเตอร์ขนานแบบคลัสเตอร์ โดยอาจจะไม่กล่าวถึงรายละเอียดเชิงวิชาการมากนัก เพราะต้องการให้เห็นภาพองค์ประกอบโดยรวมเกี่ยวกับการทำงานของระบบนี้มากกว่าการอ้างอิงเชิงทฤษฎี ซึ่งเนื้อหาภายในบทจะเริ่มจากการชี้ให้เห็นความสำคัญของเครื่องคอมพิวเตอร์ประสิทธิภาพสูงในการคำนวณทางวิทยาศาสตร์ ชนิดและโครงสร้างของคอมพิวเตอร์แบบขนาน การจำแนกชนิดของคอมพิวเตอร์แบบคลัสเตอร์ และชี้ให้เห็นความแตกต่างระหว่าง คอมพิวเตอร์แบบคลัสเตอร์ กริด และเครือข่ายแบบ P2P ส่วนประกอบโดยรวมของระบบ โครงสร้างของระบบคลัสเตอร์ เครื่องมือและโปรแกรมอัตถะประโยชน์ หัวข้อสุดท้ายจะเป็นการอธิบายสรุปเกี่ยวกับการจัดการระบบคลัสเตอร์

5.2 ความสำคัญของคอมพิวเตอร์ความเร็วสูง

ในการคำนวณทางวิทยาศาสตร์ วิศวกรรมศาสตร์ หรือทางด้านอื่น ๆ ในปัจจุบันและอนาคตจำเป็นต้องใช้คอมพิวเตอร์ที่มีประสิทธิภาพที่สูงมากขึ้นในการประมวลผล ไม่ว่าจะเป็นการสร้างภาพเชิงปริมาตรทางการแพทย์ขนาดใหญ่มาก ดังเช่นในงานวิจัยนี้ หรืองานทางด้านอื่น ๆ เช่น การจำลองสภาวะอากาศของโลก การวิจัยตัวยาใหม่ ๆ การวิเคราะห์รูปแบบรหัสพันธุกรรมหรือ DNA ของเชื้อโรคที่ไม่เคยรู้จัก อย่างเช่น การวิจัยรหัสพันธุกรรมของไวรัสต้นเหตุของโรค Severe Acute Respiratory Syndrome หรือ SARS ที่เพิ่งเกิดขึ้นในเดือนมีนาคม พ.ศ. 2546 ที่ผ่านมา การประมวลผลคลื่นสัญญาณต่าง ๆ ที่ได้รับจากอวกาศในชื่อโครงการ SETI@home (The Search for Extraterrestrial Intelligence :SETI: http://setiathome.berkeley.edu) หรือโครงการวิจัย TeraGrid (http://www.teragrid.org) ซึ่งเป็นโครงการคอมพิวเตอร์แบบกริด เพื่อใช้ประโยชน์ร่วมกันในการประมวลผลงานทางด้านวิทยาศาสตร์และเทคโนโลยี และสุดท้ายโครงการงานวิจัยเกี่ยวกับสภาวะแผ่นดินไหวของโลก (http://www.neesgrid.org) เป็นต้น

ดังนั้นจะเห็นได้ว่าถ้าหากเรามีเครื่องคอมพิวเตอร์ที่มีประสิทธิภาพในการคำนวณสูง ผลลัพธ์ที่ต้องการของปัญหาจะถูกคำนวณออกมาได้รวดเร็ว ทำให้เราสามารถเข้าใจปัญหา และสามารถนำผลไปวิเคราะห์เพื่อใช้งานต่อไปได้ถูกต้องและง่ายยิ่งขึ้น

และหนึ่งในบรรดาคอมพิวเตอร์ความเร็วสูงที่นิยมนำมาใช้ในงานวิจัยต่าง ๆ คือ ระบบคอมพิวเตอร์แบบคลัสเตอร์ ซึ่งเป็นเทคโนโลยีที่ได้รับการพัฒนาให้ดีขึ้นโดยลำดับ และเริ่มถูกนำมาใช้ในงานวิจัยอย่างกว้างขวาง แต่อาจจะมีคำถามว่าทำไมถึงต้องมีการสร้างระบบคลัสเตอร์ขึ้นมาในโลกปัจจุบันซึ่งมีซุปเปอร์คอมพิวเตอร์ หรือคอมพิวเตอร์ความเร็วสูงที่ผลิตในเชิงการค้าได้อย่างมีประสิทธิภาพอย่างมากมายแล้ว คำตอบสั้น ๆ ก็คือ เรื่องของงบประมาณ ซุปเปอร์คอมพิวเตอร์นั้นมีราคาที่ค่อนข้างแพงมากเมื่อเปรียบเทียบกับคอมพิวเตอร์แบบอื่น ๆ หรือเปรียบเทียบราคากันไม่ได้เลยกับคอมพิวเตอร์ส่วนบุคคลทั่วไป ดังนั้นการสร้างระบบคลัสเตอร์จึงน่าจะเป็นอีกทางเลือกหนึ่งที่เหมาะสมสำหรับห้องวิจัยและมหาวิทยาลัยซึ่งมีงบประมาณไม่มากนัก ในการที่จะมีคอมพิวเตอร์ความเร็วสูงแต่ไม่ต้องลงทุนในด้านงบประมาณมากนักเพื่อใช้ในการคำนวณ เพราะระบบคลัสเตอร์จะเป็นการนำเอาเทคโนโลยีที่ราคาถูกกว่ามาทำงานร่วมกันให้เกิดประสิทธิภาพสูงสุด และเพียงพอสำหรับงานวิจัยต่าง ๆ

อุปกรณ์ที่นำมาใช้ต่อร่วมกันของคอมพิวเตอร์แบบคลัสเตอร์สามารถจะเป็นอุปกรณ์ที่เก่าหรือใหม่ก็ได้ ขอเพียงแค่เป็นอุปกรณ์ที่ยังมีสภาพดีอยู่และสามารถทำงานร่วมกับอุปกรณ์อื่น ๆ ได้ ตัวอย่าง ในสมัยแรกที่มีการพัฒนาระบบแบบคลัสเตอร์ขึ้นมานั้น เครื่องคอมพิวเตอร์ในระบบก็ใช้หน่วยประมวลผลของบริษัทอินเทล รุ่น Intel DX4 ในการทำงาน ฟังดูอาจจะไม่หรูหรามากนัก แต่ระบบนี้ก็สามารถทำงานได้จริง และทำให้งานวิจัยต่าง ๆ สำเร็จลงได้ และเป็นต้นแบบให้กับคอมพิวเตอร์แบบคลัสเตอร์ในปัจจุบันที่มีเทคโนโลยีความเร็วของหน่วยประมวลผลเพิ่มมากขึ้น ดังนั้นจึงเห็นได้ว่าในการสร้างระบบคลัสเตอร์ไม่จำเป็นต้องทำการลงทุนกับฮาร์ดแวร์ใหม่ สามารถใช้อุปกรณ์ที่มีอยู่แล้วได้เลย หรือถ้าหากมีงบประมาณเพียงพอก็สามารถที่จะเพิ่มหน่วยประมวลผลที่มีความเร็วสูงมากขึ้นเข้ามาในระบบเพิ่มเติมได้ด้วย

5.3 ชนิดและโครงสร้างของคอมพิวเตอร์แบบขนาน

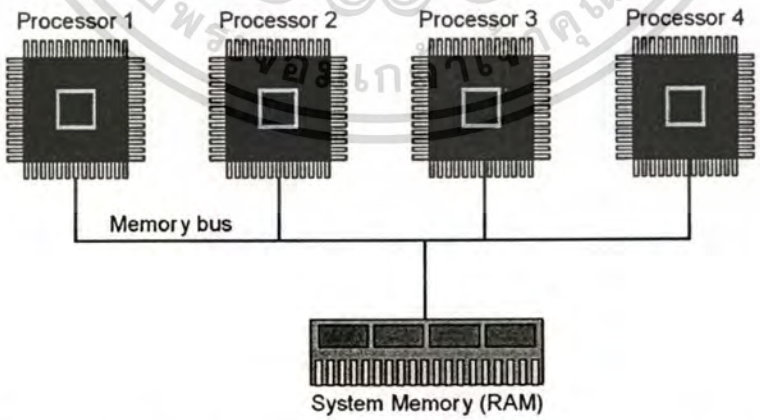
เครื่องคอมพิวเตอร์ประสิทธิภาพสูงที่กล่าวในหัวข้อที่ผ่านมานั้น ทั้งซุปเปอร์คอมพิวเตอร์และคอมพิวเตอร์แบบคลัสเตอร์ ล้วนแล้วแต่เป็นเครื่องคอมพิวเตอร์คำนวณแบบขนานทั้งสิ้น เพราะเทคโนโลยีนี้เป็นการใช้ประโยชน์จากการนำหลาย ๆ หน่วยประมวลผลมาทำงานร่วมกันให้เกิดประสิทธิภาพในการคำนวณมากกว่าการใช้เพียงแค่หน่วยประมวลผลเดียว ซึ่งหลักการการประมวลผลแบบขนานเป็นวิธีการในการแบ่งปัญหาขนาดใหญ่ออกเป็นปัญหาย่อยๆ แล้วทำการแก้ปัญหาเหล่านั้น

ไปพร้อม ๆ กัน ในปัจจุบันระบบประมวลผลแบบขนานถูกใช้อย่างแพร่หลายทั้งในการคำนวณทางวิทยาศาสตร์และการประยุกต์ใช้กับงานทั่วๆไป จากความต้องการความสามารถในการประมวลผลที่มากขึ้นนี้ทำให้มีความต้องการระบบคอมพิวเตอร์ที่มีความเร็วสูงขึ้น ราคาถูกลง และมีการพัฒนาอย่างต่อเนื่อง

สำหรับหลักการทำงานของเครื่องคอมพิวเตอร์แบบขนานนั้นกล่าวคือ การที่เราสามารถนำเอาหลายๆ หน่วยประมวลผลมาทำงานร่วมกันหรือคำนวณปัญหาเดียวกัน โดยที่หน่วยประมวลผลเหล่านี้ต้องสามารถทำการสื่อสารระหว่างกันได้ ไม่ว่าจะโดยวิธีการใช้หน่วยความจำร่วมกัน (Share Memory) หรือโดยวิธีการส่งข้อความระหว่างหน่วยประมวลผลต่อหน่วยประมวลผล (Message Passing) [Lester,1993] ซึ่งเครื่องคอมพิวเตอร์ที่ใช้หลักการหน่วยความจำร่วมนี้จะเรียกว่า ระบบแบบหลายหน่วยประมวลผล (Multiprocessor) และระบบคอมพิวเตอร์ที่ใช้หลักการส่งผ่านข้อความจะถูกเรียกว่า ระบบแบบหลายเครื่องประมวลผล (Multi computer) ซึ่งแต่ละหน่วยประมวลผลจะมีหน่วยความจำเป็นของตัวเองทำให้สามารถเรียกโครงสร้างแบบนี้ได้อีกอย่างหนึ่งว่า ระบบคอมพิวเตอร์แบบกระจาย (Distributed Computing) ซึ่งรายละเอียดของคอมพิวเตอร์ขนานแต่ละแบบสามารถกล่าวได้โดยละเอียดดังนี้

5.3.1 ระบบแบบหลายหน่วยประมวลผล (Multiprocessor)

หลักการทำงานของเครื่องคอมพิวเตอร์แบบหลายหน่วยประมวลผลสามารถแสดงรูปแบบการเชื่อมต่อได้ดังรูปต่อไปนี้



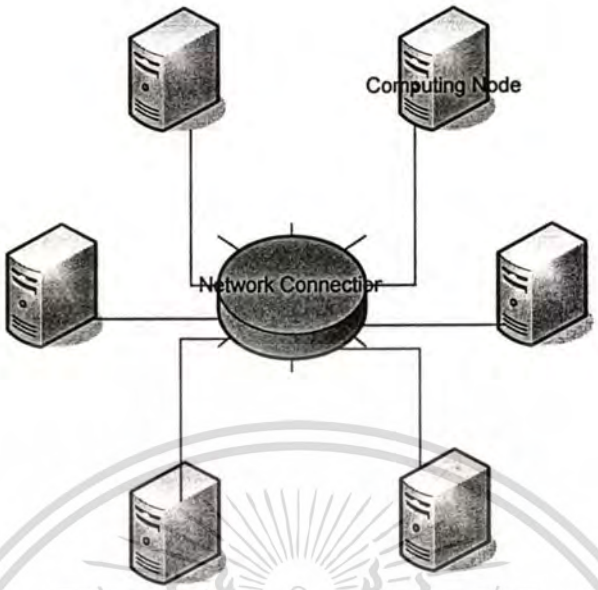
รูปที่ 5.1 โครงสร้างการทำงานของสถาปัตยกรรมแบบหลายหน่วยประมวลผล

จากรูปที่ 5.1 จะเห็นได้ว่าแต่ละหน่วยประมวลผลสามารถทำงานไปพร้อม ๆ กันได้ เพราะทุกหน่วยประมวลผลสามารถที่จะเข้าถึงหน่วยความจำส่วนกลางผ่านทางช่องสัญญาณบัส (Bus) ได้อย่างเท่าเทียมกัน การทำงานของหน่วยประมวลผลในระบบแบบนี้จะทำงานคล้าย ๆ กับระบบที่มีหน่วยประมวลผลเดียว คือ จะทำการอ่านค่าข้อมูลมาจากหน่วยความจำส่วนกลาง ทำการคำนวณค่าใหม่ แล้วใส่กลับเข้าไปที่หน่วยความจำส่วนกลางเช่นเดิม แต่ว่าทุกหน่วยประมวลผลจะสามารถทำงานต่าง ๆ เหล่านี้ได้อย่างพร้อม ๆ กัน ฉะนั้นจะได้ว่า หากในระบบมีหน่วยประมวลผล N หน่วย ระบบนี้จะมีความเร็วสูงที่สุดที่เป็นไปได้ เท่ากับ N เท่าของเครื่องที่มีหน่วยประมวลผลเพียงตัวเดียว แต่ในทางปฏิบัติความเร็วที่เพิ่มขึ้นต่อหน่วยประมวลผลมักจะไม่ได้เป็นไปตามทฤษฎีมากนัก เพราะปัญหาหลาย ๆ อย่างในทางปฏิบัติ

ปัญหาของระบบนี้ที่พบได้บ่อยคือ ความคับคั่งในการเข้าถึงหน่วยความจำ (Memory Contention) ซึ่งปัญหาเกิดขึ้นเนื่องมาจากในระบบมีหลายหน่วยประมวลผลที่ต้องการเข้าถึงหน่วยความจำเพื่ออ่านหรือเขียน ทำให้หน่วยความจำไม่สามารถตอบสนองต่อความต้องการในการอ่านหรือเขียนทั้งหมดพร้อมกันได้ จึงมีผลทำให้เกิดปัญหานี้ขึ้น ดังนั้นการออกแบบโครงสร้างของระบบที่ดีจึงมีส่วนสำคัญที่จะช่วยแก้ปัญหาเรื่องความคับคั่งในการเข้าถึงหน่วยความจำได้ ทางออกหนึ่งก็คือการกำหนดให้แต่ละหน่วยประมวลผลมีหน่วยความจำชั่วคราว (Local cache memory) เป็นของตัวเอง เพื่อช่วยให้หน่วยประมวลผลสามารถเก็บข้อมูลในการคำนวณหรือผลลัพธ์หลังจากการคำนวณได้มากขึ้น ก่อนที่จะติดต่อกับหน่วยความจำส่วนกลางต่อไป

5.3.2 ระบบแบบหลายเครื่องประมวลผล (Multi computer)

การหลีกเลี่ยงปัญหาความคับคั่งในการเข้าถึงหน่วยความจำในหัวข้อที่ผ่านมา สามารถแก้ไขได้อีกวิธีหนึ่งนั่นคือ การกระจายให้ทุก ๆ เครื่องมีหน่วยความจำขนาดใหญ่เป็นของตัวเอง แล้วทำการติดต่อสื่อสารกันผ่านทางเครือข่ายโดยใช้วิธีการส่งข้อความ ซึ่งระบบนี้มักจะถูกเรียกว่าระบบหลายเครื่องประมวลผลหรือ Multi computer โดยสามารถแสดงรูปแบบการเชื่อมต่อของระบบได้ดังรูปต่อไปนี้



รูปที่ 5.2 โครงสร้างการทำงานของสถาปัตยกรรมแบบหลายเครื่องประมวลผล

จากรูปที่ 5.2 แต่ละเครื่องจะสามารถทำงานได้โดยอิสระต่อกัน และเป็นเครื่องคนละรุ่นคนละรูปแบบได้โดยอิสระ (Heterogeneous) หรือจะเป็นเครื่องรุ่นเดียวกันแบบเดียวกันทั้งหมดก็ได้ (Homogeneous) โดยเมื่อทำการคำนวณจะเก็บข้อมูลเกี่ยวกับการคำนวณไว้ที่หน่วยความจำของแต่ละเครื่องเอง แล้วทำการสื่อสารข้อมูลและรับส่งค่าผลลัพธ์กับเครื่องอื่น ๆ ผ่านทางเครือข่ายโดยวิธีการส่งผ่านข้อความ ซึ่งหลักการนี้จะแตกต่างกับหัวข้อที่ผ่านมาทั้งในด้านโครงสร้างของฮาร์ดแวร์และวิธีการเขียนโปรแกรม เพราะในหัวข้อที่ผ่านมาทุกหน่วยประมวลผลจะใช้หน่วยความจำที่เดียวกัน แต่ในโครงสร้างแบบนี้หน่วยประมวลผลแต่ละหน่วยจะมีพื้นที่เก็บข้อมูลเป็นของตัวเองแยกจากกัน หรืออาจจะกล่าวได้ว่ามีหน่วยความจำสองรูปแบบ คือ หน่วยความจำแบบส่วนตัวของหน่วยประมวลผลหรือหน่วยความจำแบบโลคอล (Local memory) และหน่วยความจำของหน่วยประมวลผลอื่นหรือหน่วยความจำแบบรีโมต (Remote memory) ซึ่งการเข้าถึงหน่วยความจำแบบโลคอลนั้นแต่ละหน่วยประมวลผลสามารถเข้าถึงได้โดยตรง แต่ถ้าหากเป็นการเข้าถึงเพื่ออ่านหรือส่งค่าให้หน่วยความจำอื่น ๆ ต้องกระทำผ่านเครือข่ายตามวิธีการส่งผ่านข้อความ

โครงสร้างการเชื่อมต่อของระบบแบบหลายเครื่องประมวลผลนี้มีอยู่ด้วยกันหลายรูปแบบ ซึ่งรูปแบบการเชื่อมต่อเครือข่ายแต่ละแบบนั้นมีวัตถุประสงค์เพื่อให้ได้ประสิทธิภาพการคำนวณให้สูงมากที่สุดเท่าที่จะเป็นไปได้ โดยสามารถแยกตามรูปแบบและโครงสร้างการเชื่อมต่อออกได้เป็นสามประเภทดังนี้

- **คอมพิวเตอร์ระบบคลัสเตอร์ (Clustering Computer)**

แนวความคิดของระบบคลัสเตอร์เกิดขึ้นในปี ค.ศ. 1960 โดยบริษัท IBM ต้องการเชื่อมต่อระบบเมนเฟรม (Mainframe) ขนาดใหญ่เข้าด้วยกันเพื่อใช้ในเชิงการค้า แต่ในตอนนั้นระบบคลัสเตอร์ยังไม่เป็นที่รู้จักอย่างแพร่หลายเนื่องจากข้อจำกัดทางด้านราคาและความแพร่หลายของฮาร์ดแวร์ (Hardware) นอกจากนั้นยังขาดเครื่องมือ (Tools) มาตรฐานสำหรับการเชื่อมต่อเครื่องในระบบเข้าด้วยกัน จนในช่วงกลางปี 1980 เป็นช่วงที่เทคโนโลยีระบบคลัสเตอร์สามารถเกิดขึ้นได้เนื่องจากสามารถสร้างไมโครโพรเซสเซอร์ที่มีประสิทธิภาพได้ และเกิดระบบเครือข่ายความเร็วสูงขึ้น ซึ่งเครื่องคอมพิวเตอร์ในระบบเครือข่ายสามารถสื่อสารกันได้ภายในเวลาไม่เกินหนึ่งมิลลิวินาที นอกจากนั้นความต้องการความเร็วในการประมวลผลงานทางวิทยาศาสตร์และวิศวกรรมศาสตร์ซึ่งมีมากขึ้นอย่างต่อเนื่องก็มีส่วนผลักดันให้เกิดเทคโนโลยีคลัสเตอร์ขึ้น

ระบบคลัสเตอร์ได้เกิดขึ้นอย่างเป็นรูปธรรมในปี 1994 เมื่อ Thomas Sterling และ Don Becker จาก CESDIS (The Center of Excellence in Space Data and Information Sciences) ต้องการที่จะวิเคราะห์ข้อมูลจากอวกาศที่มีจำนวนมากและซับซ้อนซึ่งต้องใช้เครื่องคอมพิวเตอร์ที่สามารถประมวลผลได้เร็วมากๆและในขณะนั้นมีเพียงเครื่องซุปเปอร์คอมพิวเตอร์ซึ่งมีราคาแพงแต่เนื่องจากงบประมาณที่มีอยู่จำกัดทำให้ Sterling และ Becker มีแนวความคิดที่จะใช้ระบบคลัสเตอร์มาทำการประมวล โดยได้สร้างระบบคลัสเตอร์จากเครื่องคอมพิวเตอร์ซึ่งใช้ชิพ Intel DX4 เป็นหน่วยประมวลผลกลาง (CPU) และส่วนประกอบอื่นๆซึ่งสามารถหาได้ง่ายจากท้องตลาดและได้ตั้งชื่อระบบคลัสเตอร์นี้ว่า Beowulf (Sterling and Becker, 1994)

ระบบคลัสเตอร์มักจะเป็นระบบที่แต่ละหน่วยประมวลผลหรือโหนดการคำนวณ (Computing node) อยู่ใกล้กัน หรืออยู่ในพื้นที่ใกล้เคียงกัน แล้วทำการเชื่อมต่อระหว่างโหนดต่อโหนดหรือแต่ละหน่วยประมวลผลผ่านทางเครือข่าย เช่น Gigabit Ethernet, Myrinet, InfiniBand, Quadrics หรือ เครือข่ายรูปแบบอื่น ๆ ซึ่งต้องเป็นเทคโนโลยีเครือข่ายที่มีความเร็วสูง และค่า Latency Time ต่ำ เพื่อให้การติดต่อสื่อสารระหว่างกันสามารถเป็นไปได้อย่างรวดเร็ว และสูญเสียเวลาในเครือข่ายน้อยที่สุดเสมือนว่าทุกหน่วยประมวลผลอยู่ใกล้กันมากจนเกือบจะเหมือนอยู่บนแผงวงจรรวม Mother Board เดียวกัน ทำให้การส่งผ่านข้อความเพื่อแลกเปลี่ยนข้อมูลต่าง ๆ ระหว่างคำนวณเป็นไปอย่างรวดเร็ว

สำหรับรูปแบบการเชื่อมต่อและโครงสร้างการทำงานต่าง ๆ ทั้งในด้านซอฟแวร์และฮาร์ดแวร์จะกล่าวโดยละเอียดในหัวข้อต่อ ๆ ไปของบทนี้

- **กริดคอมพิวเตอร์**

เนื่องจากมีความต้องการคอมพิวเตอร์ความเร็วสูงเพื่อใช้ในการคำนวณทางคณิตศาสตร์ที่มีความยุ่งยากซับซ้อนมาก ๆ ตัวอย่างเช่น การคำนวณทางไฟไนต์อีลิเมนต์ (Finite Element) การวิเคราะห์โครงสร้างรหัสพันธุกรรมของไวรัส การสังเคราะห์โปรตีน ซึ่งเป็นงานที่จำเป็นต้องใช้ประสิทธิภาพของคอมพิวเตอร์ในการคำนวณสูงมาก แต่ลักษณะงานเหล่านี้ไม่ต้องการความเร็วในการสื่อสารระหว่างหน่วยประมวลผลมากนัก หน่วยประมวลผลสามารถสื่อสารระหว่างกันด้วยความเร็วไม่สูงมาก การแลกเปลี่ยนข้อมูลระหว่างหน่วยประมวลผลมีเพียงเล็กน้อยเท่านั้น ซึ่งจากความต้องการและเงื่อนไขเหล่านี้ทำให้มีความพยายามในการพัฒนาระบบคอมพิวเตอร์ขนาดใหญ่ขึ้น โดยเป็นการรวมเอาระบบคลัสเตอร์ หลาย ๆ คลัสเตอร์เข้าไว้ด้วยกันทำให้สามารถทำการคำนวณร่วมกันได้

กริดคอมพิวเตอร์เป็นระบบคอมพิวเตอร์แบบกระจายที่มีขนาดใหญ่มาก และมีความเร็วในการคำนวณสูงมากเช่นกัน ทำให้การคำนวณปัญหาขนาดใหญ่สามารถเสร็จได้ในเวลาอันรวดเร็ว ตัวอย่างของกริดคอมพิวเตอร์ขนาดใหญ่เช่น โครงการ TeraGrid (http://www.teragrid.org) ซึ่งเป็นการรวมเอาระบบคลัสเตอร์จากหลาย ๆ ที่ให้มาทำงานร่วมกัน ทำให้ความสามารถในการประมวลผลของระบบนี้มีความเร็วสูงขึ้น

การเชื่อมต่อในระบบกริดคอมพิวเตอร์นั้นจะทำการเชื่อมคลัสเตอร์แต่ละที่เข้าด้วยกันโดยใช้เครือข่ายที่มีอยู่แล้วในปัจจุบัน นั่นคือเครือข่ายอินเทอร์เน็ตนั่นเอง ทำให้แต่ละไซต์ (Site) สามารถอยู่กระจายในที่ต่าง ๆ ได้ทั่วโลก ขอเพียงแค่เชื่อมต่อกับระบบอินเทอร์เน็ต ก็จะสามารถทำงานร่วมกันในระบบของกริดได้แล้ว

การจัดการทรัพยากรต่าง ๆ ในระบบกริดคอมพิวเตอร์จะต้องมีซอฟแวร์ (Software) ที่เรียกว่า Grid Middleware เป็นตัวกลางในการจัดการและเชื่อมการทำงานเข้าด้วยกัน ซอฟท์แวร์ที่ได้รับความนิยมในปัจจุบันตัวอย่างเช่น Globus Project (http://www.globus.org)

- **เครือข่ายแบบ P2P (Peer-to-Peer)**

โครงข่ายคอมพิวเตอร์แบบ Peer-to-Peer เป็นคอมพิวเตอร์ความเร็วสูงอีกรูปแบบหนึ่งที่ได้รับความนิยมมากขึ้นเรื่อย ๆ เพราะเครื่องที่ทำงานในระบบคือเครื่องของผู้ใช้งานทั่ว ๆ ไป ที่เชื่อมต่ออยู่กับระบบอินเทอร์เน็ต เพราะฉะนั้นขนาดและจำนวนของหน่วยประมวลผลในระบบจึงสามารถเพิ่มขยายได้เรื่อย ๆ ตามจำนวนของผู้ที่เข้าร่วมในโครงการ หรืออาจกล่าวได้ว่าคอมพิวเตอร์ทุกเครื่องที่ต่ออยู่กับระบบอินเตอร์เน็ตสามารถที่จะร่วมประมวลผลในรูปแบบของเครือข่ายประเภทนี้ได้ งานวิจัยในปัจจุบันที่ใช้การประมวลผลโดยใช้เครือข่ายแบบ P2P มีตัวอย่างเช่น โครงการการค้นหาสิ่งมีชีวิตจากอวกาศหรือ SETI@Home ซึ่งเคยได้กล่าวไปแล้ว โดยโครงการนี้จะทำการรับเอาคลื่นวิทยุจากจานรับ

สัญญาณจากอวกาศมาทำการประมวลผลเพื่อหาสัญญาณที่ไม่ใช่รูปแบบของสัญญาณในธรรมชาติทั่วไป อีกโครงการหนึ่งเช่น การวิจัยตัวยาชนิดใหม่โดยใช้โครงสร้างแบบกริดและเครือข่ายแบบ P2P ในชื่อโครงการ DesignDrug@Home (http://www.gridbus.org/vlab/)

คอมพิวเตอร์ความเร็วสูงแบบหลายหน่วยประมวลผลทั้งสามชนิด สามารถที่จะแสดงให้เห็นถึงความแตกต่างของแต่ละรูปแบบได้ดังตารางต่อไปนี้

ตารางที่ 5.1 เปรียบเทียบความแตกต่างของคอมพิวเตอร์แบบกระจาย (Buyya, 1999)

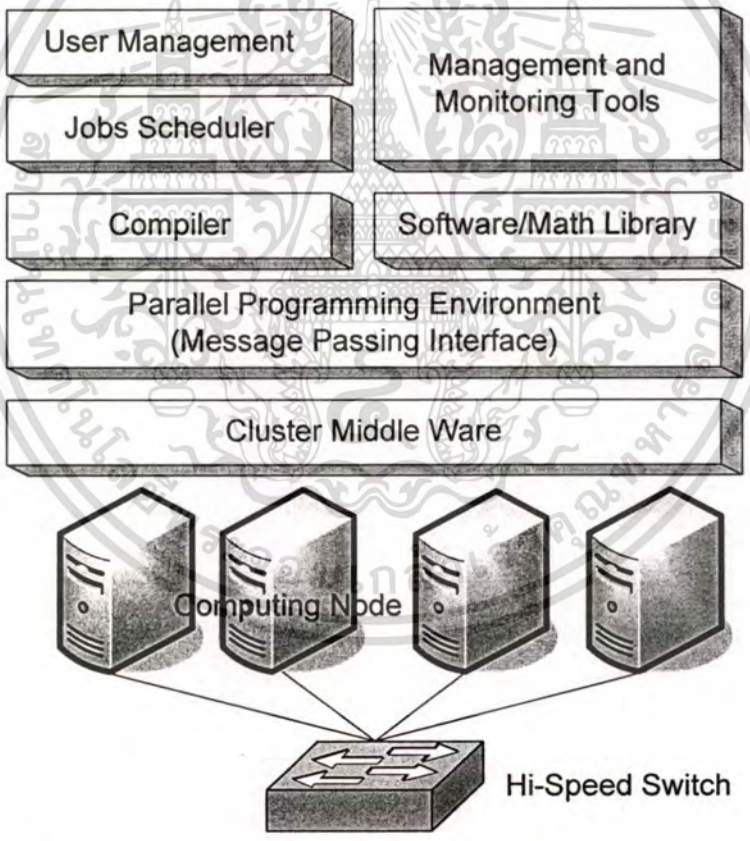
Characteristic	Cluster	Grid	P2P
Population	Commodity Computers	High-end computers	Edge of network (desktop PC)
Ownership	Single	Multiple	Multiple
Discovery	Membership Services	Centralised Index & Decentralised Info	Decentralized
User Management	Centralised	Decentralised	Decentralised
Resource management	Centralized	Distributed	Distributed
Allocation/Scheduling	Centralised	Decentralised	Decentralised
Inter-Operability	VIA based?	No standards yet	No standards
Single System Image	Yes	No	No
Scalability	100s	1000?	Millions? [@Home]
Capacity	Guaranteed	Varies, but high	Varies
Throughput	Medium	High	Very High
Speed(Latency Time. Bandwidth)	Low, high	High, Low	High, Low

ซึ่งจากที่กล่าวมาทั้งหมดจะเห็นได้ว่าคอมพิวเตอร์ความเร็วสูงแบบหลายหน่วยประมวลผลเหล่านี้เริ่มเข้ามามีบทบาทในงานวิจัยต่าง ๆ ในยุคปัจจุบันมากยิ่งขึ้น จากเดิมที่งานวิจัยหลาย ๆ อย่าง ต้องทำการทดลองในห้องปฏิบัติการ หรือแสดงได้เพียงแค่รูปแบบสมการเพียงอย่างเดียว แต่ในปัจจุบันสามารถที่จะจำลองการทำงานหรือหาผลลัพธ์ได้บนคอมพิวเตอร์ความเร็วสูงเหล่านี้ ทำให้งานทางด้านวิทยาศาสตร์และเทคโนโลยีหลาย ๆ ด้านมีการพัฒนาอย่างรวดเร็วมากขึ้น

5.4 ส่วนประกอบโดยรวมของคอมพิวเตอร์แบบคลัสเตอร์

เราอาจกล่าวได้โดยง่ายว่าระบบคลัสเตอร์คือ กลุ่มของคอมพิวเตอร์ที่มีการเชื่อมต่อผ่านเครือข่ายความเร็วสูงและสามารถคำนวณงานที่ถูกแบ่งออกเป็นงานย่อย ๆ แล้วร่วมกันได้ ซึ่งรูปแบบของเครือข่ายและการเชื่อมต่อนี้มีด้วยกันหลายวิธีและหลายชนิดเครือข่าย ส่วนการที่จะทำให้กลุ่มของคอมพิวเตอร์สามารถทำงานร่วมกันได้นั้นจะต้องมีซอฟแวร์เป็นตัวกลางในการเชื่อมการทำงานของแต่ละหน่วยประมวลผลเข้าด้วยกัน (Spector, 2000) ซึ่งมาตรฐานที่นิยมใช้คือ ระบบการส่งผ่านข้อความหรือ Message Passing Interface และคลัสเตอร์มิดเดิลแวร์ (Cluster Middle Ware)

ส่วนประกอบโดยรวมทั้งหมดของระบบคอมพิวเตอร์แบบคลัสเตอร์นั้นจะประกอบไปด้วยส่วนต่าง ๆ ดังบล็อกไดอะแกรมด้านล่างดังนี้



รูปที่ 5.3 บล็อกไดอะแกรมแสดงส่วนประกอบของระบบคลัสเตอร์

จากรูปสามารถที่จะแบ่งการทำงานออกได้ทั้งหมดสามส่วนด้วยกันคือ

1. โครงสร้างพื้นฐานของระบบคลัสเตอร์ ซึ่งจะประกอบไปด้วยส่วนประกอบทางฮาร์ดแวร์ การเชื่อมต่อเครือข่าย ระบบปฏิบัติการ และคลัสเตอร์มิดเดิลแวร์
2. เครื่องมือและโปรแกรมอัตถะประโยชน์ ประกอบด้วย โปรแกรมสื่อสารระหว่างโหนดโดยการส่งข้อความ คอมไพเลอร์ และชุดคำสั่งทางคณิตศาสตร์
3. การจัดการระบบคลัสเตอร์ ประกอบด้วยตัวจัดการคิวงาน การจัดการสิทธิของผู้ใช้งานในระบบ และระบบตรวจสอบและจัดการคลัสเตอร์

ซึ่งจะได้กล่าวโดยละเอียดในหัวข้อที่ 5.6, 5.7 และ 5.8 ตามลำดับต่อไป

5.5 การจำแนกคอมพิวเตอร์แบบคลัสเตอร์

เราสามารถที่จะจำแนกประเภทของระบบคลัสเตอร์ออกตามเกณฑ์ต่าง ๆ ได้ดังนี

5.5.1 จำแนกตามงานที่ประยุกต์ใช้

ระบบคลัสเตอร์สามารถนำเอาประยุกต์ใช้ได้กับงานที่หลากหลาย ไม่ว่าจะเป็นงานทางด้านการคำนวณหรืองานทางด้านการเป็นเครื่องแม่ข่ายให้บริการงานต่าง ๆ โดยสามารถแบ่งย่อยได้เป็นสองประเภทดังนี้

- ระบบคลัสเตอร์แบบประสิทธิภาพสูง (High Performance (HP) Clusters) มักจะถูกนำไปประยุกต์ใช้ในการคำนวณทางด้านวิทยาศาสตร์และคณิตศาสตร์ ระบบคลัสเตอร์แบบนี้จะถูกสร้างขึ้นมาให้มีความรวดเร็วในการคำนวณมากที่สุด ประสิทธิภาพของหน่วยประมวลผลจะต้องสูงเพียงพอ อีกทั้งเครือข่ายที่ใช้ในการเชื่อมต่อต้องมีคุณภาพดีมาก ประสิทธิภาพในการคำนวณจึงจะสูงตามไปด้วย ระบบคลัสเตอร์ที่ถูกสร้างขึ้นมาเพื่อใช้ในงานวิจัยนี้ก็เป็นคลัสเตอร์แบบความเร็วสูงเพื่อทำให้เวลาที่ใช้ในการสร้างภาพเชิงปริมาตรใช้เวลาในการสร้างภาพน้อยที่สุด
- ระบบคลัสเตอร์แบบเสถียรภาพสูง (High Availability (HA) Clusters) ระบบคลัสเตอร์แบบนี้จะเน้นไปทางด้านเครื่องแม่ข่ายที่ให้บริการงานต่าง ๆ เช่น ให้บริการเป็นเว็บเซอร์ฟเวอร์ (Web Server) หรือให้บริการพื้นที่เก็บข้อมูลบนเครือข่าย (Storage Server) เพื่อจะทำให้มั่นใจได้ว่าผู้ใช้งานทั่ว ๆ ไปจะสามารถเข้าถึงทรัพยากรต่าง ๆ ได้ตลอดเวลา การให้บริการไม่ได้ขึ้นอยู่เครื่องเซอร์ฟเวอร์เครื่องใดเครื่องหนึ่งเพียงเครื่องเดียว

การสร้างระบบคลัสเตอร์แต่ละแบบนั้นจะมีข้อแตกต่างกัน วิธีการต่างกัน และซอฟต์แวร์ที่ใช้มีลักษณะแตกต่างกัน เพราะระบบหนึ่งต้องการความรวดเร็วในการคำนวณเพียงอย่างเดียว ส่วนอีกระบบหนึ่งต้องการความเชื่อถือในการเข้าถึงบริการได้ตลอดเวลา

5.5.2 จำแนกตามลักษณะของเครื่องในระบบ

ระบบคลัสเตอร์เกิดจากการเชื่อมต่อเครื่องคอมพิวเตอร์หลาย ๆ เครื่องเข้าด้วยกัน โดยเครื่องคอมพิวเตอร์เหล่านี้อาจจะมีลักษณะเหมือนกันทั้งหมดหรือไม่เหมือนกันเลยก็ได้ ซึ่งถ้าหากจำแนกระบบคลัสเตอร์ตามลักษณะของฮาร์ดแวร์ในระบบแล้ว สามารถที่จะจำแนกออกได้เป็นสองประเภทเช่นกันคือ

- ระบบคลัสเตอร์แบบเนื้อเดียว (Homogeneous Cluster) ระบบคลัสเตอร์แบบนี้เป็นระบบที่มีองค์ประกอบทางด้านฮาร์ดแวร์และซอร์ฟแวร์ที่ใช้ในแต่ละเครื่องเหมือนกันทั้งหมดได้แก่ หน่วยประมวลผลกลาง ชนิดและขนาดของหน่วยความจำ ชนิดและขนาดของฮาร์ดดิสก์ และชนิดของระบบปฏิบัติการ เป็นต้น โดยระบบคลัสเตอร์นี้เป็นแบบที่นิยมสร้าง เนื่องจากการบริหารจัดการระบบสามารถทำได้อย่างสะดวก นอกจากนั้นการเขียนโปรแกรมเพื่อทำการประมวลผลบนระบบคลัสเตอร์สามารถเขียนในครั้งเดียวแล้วทำงานได้กับทุก ๆ เครื่องในระบบ
- ระบบคลัสเตอร์แบบเนื้อผสม (Heterogeneous Cluster) ระบบคลัสเตอร์แบบนี้เป็นระบบที่มีความยืดหยุ่นสูงโดยสามารถสร้างจากเครื่องแบบใดก็ได้ที่สนับสนุนการประมวลผลแบบขนาน แต่ปัญหาของระบบนี้คือการสร้างโปรแกรมสำหรับประมวลผลแบบขนานจะมีความยุ่งยากและซับซ้อนมากขึ้นเนื่องจากต้องทำการสร้างโปรแกรมที่สามารถประมวลผลเฉพาะของแต่ละเครื่อง แต่ละระบบปฏิบัติการ เช่น เครื่องที่ใช้ระบบปฏิบัติการลีนุกส์ (Linux) และระบบที่ใช้ไมโครซอฟส์วินโดว์ (Microsoft Windows) เป็นระบบปฏิบัติการ โปรแกรมที่สร้างขึ้นมาอาจไม่สามารถใช้ด้วยกันได้ วิธีแก้ปัญหาอย่างหนึ่งคือสร้างโปรแกรมให้สนับสนุนมาตรฐานเช่น ANSI C เป็นต้น หรืออาจสร้างโปรแกรมที่สามารถประมวลผลได้บนทุกระบบโดยที่ใช้รหัสต้นฉบับ (Source Code) ตัวเดียวกัน เช่น ภาษาจาวา (Java)

สิ่งหนึ่งที่ต้องให้ความสำคัญมากเช่นกันในระบบคลัสเตอร์แบบเนื้อผสมนี้ คือ การสมดุลงานในระบบ หรือ Load Balancing เพราะเนื่องจากว่าคอมพิวเตอร์ที่เชื่อมต่ออยู่กับระบบแบบนี้สามารถที่จะมีองค์ประกอบภายในแตกต่างกันได้ ทำให้คอมพิวเตอร์แต่ละโหนดมีความสามารถและประสิทธิภาพในการคำนวณผลได้รวดเร็วแตกต่างกัน เมื่อทำงานร่วมกันนั้นจะต้องการการกระจายงานที่เหมาะสม เพื่อให้เครื่องที่มีประสิทธิภาพมากกว่ามีโอกาสได้รับ

งานไปคำนวณมากกว่า ระบบทั้งหมดไม่ต้องรอค่าผลลัพธ์จากเครื่องที่มีความเร็วต่ำ ถ้าหากทำการสมดุลงานได้ดีแล้ว ประสิทธิภาพของระบบก็จะดีตามไปด้วย แต่ถ้าหากไม่คำนึงถึงเรื่องนี้แล้วประสิทธิภาพโดยรวมจะต่ำตามไปด้วย

5.6 โครงสร้างพื้นฐานของระบบคลัสเตอร์

ในการสร้างระบบคอมพิวเตอร์แบบคลัสเตอร์ จะต้องมีส่วนประกอบพื้นฐานของระบบซึ่งจะประกอบไปด้วยส่วนประกอบทางฮาร์ดแวร์ ระบบปฏิบัติการ และซอฟต์แวร์ที่ทำงานในระดับของเคอร์เนล (Kernel) ของระบบปฏิบัติการซึ่งเรียกซอฟต์แวร์นี้ว่า คลัสเตอร์มิดเดิลแวร์ และสุดท้ายต้องเชื่อมแต่ละโหนดเข้าด้วยกันผ่านทางเครือข่าย

5.6.1 ส่วนประกอบทางฮาร์ดแวร์

องค์ประกอบทางด้านฮาร์ดแวร์ที่จำเป็นในการพิจารณาเพื่อสร้างเป็นระบบคลัสเตอร์นั้นมีส่วนประกอบต่าง ๆ ดังนี้

- แผงวงจรรวม Mother Board หรือ System Board
- หน่วยประมวลผลหรือ CPUs
- หน่วยความจำชั่วคราว (RAM)
- หน่วยความจำหลักหรือ Disk Storage
- การ์ดเน็ตเวิร์ค (Network adapter)
- ตัวถังเครื่อง (Cases)

ซึ่งแต่ละโหนดควรจะมีส่วนประกอบต่าง ๆ ดังนี้เพื่อให้การทำงานมีประสิทธิภาพมากที่สุด ส่วนคุณสมบัติต่าง ๆ ทางด้านฮาร์ดแวร์ สามารถพิจารณาให้เหมาะสมตามความต้องการและงบประมาณที่มีอยู่

5.6.2 ระบบปฏิบัติการ

ระบบปฏิบัติการจะเป็นส่วนสำคัญอย่างมากต่อการทำงานของคอมพิวเตอร์และระบบคลัสเตอร์ เพราะคอมพิวเตอร์ในระบบคลัสเตอร์ต้องสามารถทำงานเองได้โดยอิสระ ไม่ขึ้นกับเงื่อนไขของเครื่องอื่น ถึงแม้มีเครื่องใดเครื่องหนึ่งในระบบหยุดไป คลัสเตอร์ก็ยังสามารถทำงานได้ ดังนั้นระบบปฏิบัติการจึงเป็นความจำเป็นในส่วนนี้ที่จะทำให้คอมพิวเตอร์แต่ละเครื่องสามารถทำงานโดยอิสระต่อกันได้ ระบบปฏิบัติการที่สามารถใช้กับระบบคลัสเตอร์ได้มีหลากหลาย เช่น Linux, Solaris, FreeBSD, Tru64 UNIX, HP-UX ซึ่งการเลือกใช้ระบบปฏิบัติการจะเป็นระบบปฏิบัติการตระกูลยูนิกส์

หรือ Microsoft Windows ทั้งนี้ขึ้นอยู่กับความสะดวก ความเชี่ยวชาญและอุปกรณ์ที่เลือกใช้ อีกอย่างหนึ่งคือจะขึ้นอยู่กับซอร์ฟแวร์และชุดคำสั่งที่เลือกใช้ด้วย เช่น ถ้าหากเลือกใช้ OpenMosix (http://openmosix.sf.net) ที่ทำงานได้บนลีนุกซ์เท่านั้นก็จำเป็นต้องเลือกใช้ลีนุกซ์เป็นระบบปฏิบัติการ อีกทั้งฮาร์ดแวร์ที่สนับสนุนด้วย

แต่โดยส่วนใหญ่ซอฟต์แวร์และชุดคำสั่งที่ถูกพัฒนาขึ้นมาเพื่อใช้ในระบบคลัสเตอร์นี้จะทำงานเข้ากันได้กับระบบปฏิบัติการตระกูลยูนิกส์เกือบทุกตัวอยู่แล้ว

5.6.3 คลัสเตอร์มิดเดิลแวร์

คลัสเตอร์มิดเดิลแวร์คือซอฟต์แวร์ที่ทำงานในระดับเดียวกันกับแกนหรือเคอร์เนล (Kernel) ของระบบปฏิบัติการ มีหน้าที่ในการกระจายโพรเซส (Process) จากโหนดหนึ่งไปยังโหนดอื่น ๆ ที่อยู่ในระบบคลัสเตอร์เดียวกัน โดยส่วนใหญ่คลัสเตอร์มิดเดิลแวร์นี้จะเป็นซอฟต์แวร์ที่เขียนเพิ่มเติมเข้าไปในแกนของระบบปฏิบัติการ เพื่อให้ทุกเครื่องที่อยู่ในระบบคลัสเตอร์เสมือนเป็นเครื่องคอมพิวเตอร์หลายหน่วยประมวลผลขนาดใหญ่ ที่เสมือนมีหน่วยความจำ หน่วยประมวลผลอยู่ที่เดียวกันเหมือนกับระบบคอมพิวเตอร์แบบ SMP (Symmetric Multi Processor) หรือ MMP (Massive Multi Processor) โดยมีเครือข่ายเป็นช่องทางสำหรับการติดต่อคล้ายกับบัส (Bus) ภายใน โพรเซสที่เกิดในโหนดหนึ่งสามารถข้ามหรือกระจายไปยังโหนดอื่น ๆ ได้โดยสะดวก การเขียนโปรแกรมเพื่อใช้งานบนระบบที่ใช้คลัสเตอร์มิดเดิลแวร์นี้ก็สามารถทำได้ง่าย เพราะผู้ใช้ไม่ต้องศึกษาเพิ่มเติมเกี่ยวกับการใช้งานชุดคำสั่งเหมือนกับการเขียนโปรแกรมบนสภาวะการโปรแกรมแบบขนาน (Parallel Programming Environment) ที่จะได้กล่าวในหัวข้อถัดไป การสมดุลงานหรือ Load Balancing เอง ผู้ใช้งานก็ไม่จำเป็นต้องคำนึงถึง กล่าวคือทุกอย่างทั้งการกระจายงาน การสมดุลงาน คลัสเตอร์มิดเดิลแวร์จะเป็นตัวจัดการให้ทั้งหมด แต่ทั้งนี้ในเรื่องของประสิทธิภาพและ ความรวดเร็วแล้ว อาจจะด้อยกว่าการเขียนโปรแกรมบนสภาวะการโปรแกรมแบบขนานเพราะในสภาวะแวดล้อมแบบขนาน ผู้ใช้จะต้องทำการเขียนลำดับขั้นตอนทุกอย่างด้วยตัวเอง ทำให้ความเข้าใจในเนื้องานมีมากกว่า และทำให้โปรแกรมทำงานได้รวดเร็วกว่าด้วย ตัวอย่างของคลัสเตอร์มิดเดิลแวร์มีดังนี้

- BPROC (Beowulf distribute PROCess space) เป็นซอฟต์แวร์ที่ถูกพัฒนาโดยห้องวิจัย CESDIS ของ NASA ซึ่งเป็นต้นกำเนิดของระบบคลัสเตอร์แบบ Beowulf ระดับการทำงานของ BPROC นี้จะเป็นเพียงแค่โปรแกรมจำลองสภาวะของคลัสเตอร์เท่านั้น ไม่ได้ทำงานลึกลงไปในแกนของระบบปฏิบัติการอย่างแท้จริง แต่ก็ถือได้ว่าเป็นจุดเริ่มของคลัสเตอร์มิดเดิลแวร์รุ่นหลัง ๆ โดยหลักการทำงานก็ตรงไปตรงมาตามวิธีการของคลัสเตอร์มิดเดิลแวร์ คืออนุญาตให้มีการโอนย้าย Process จากเครื่องหนึ่งไปยังอีกเครื่องหนึ่ง และช่วยกันคำนวณจนได้ผลลัพธ์ที่

ต้องการออกมาได้ เมื่อทำการคำนวณจะมีโหนดหนึ่งที่ถูกกำหนดให้เป็นโหนดหลักหรือ Master Node และส่วนที่เหลือทั้งหมดจะเรียกว่า สลาฟโหนด (Slave Node) โดยโหนดหลักหรือ Master Node จะทำการเก็บค่าหมายเลขโพรเซส (Process ID) ที่ทำงานอยู่ และรายละเอียดเกี่ยวกับโหนดอื่น ๆ ในระบบไว้ แล้วทำการกระจายโพรเซสไปยังโหนดอื่น ๆ ส่วน Slave Node เมื่อได้รับคำสั่งให้สร้างโพรเซสหรือได้รับงานมาจากมาสเตอร์ จะทำการคำนวณและส่งค่าผลลัพธ์กลับไปให้ยังมาสเตอร์โหนดต่อไป

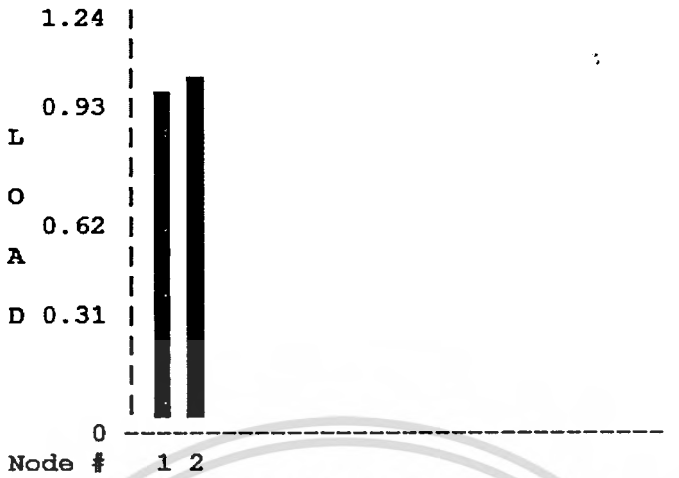
- MOSIX เป็นซอฟต์แวร์ที่ถูกพัฒนาขึ้นมาโดย Hebrew University of Jerusalem เพื่อทำหน้าที่ทำให้เกิดการกระจายงานในระบบตามมาตรฐานของคลัสเตอร์มิดเดิลแวร์ ซึ่งแตกต่างจาก BPROC ตรงที่ MOSIX ไม่ได้เป็นเพียงแค่โพรเซสที่จำลองสภาวะเท่านั้น แต่ MOSIX ทำงานในระดับของแกนระบบปฏิบัติการ จึงทำให้มีประสิทธิภาพที่ดีกว่า สามารถควบคุมการกระจายงานต่าง ๆ ได้ดีกว่า คุณสมบัติที่สำคัญ ๆ คือ การกระจายโพรเซสในระบบคลัสเตอร์ การทำหน้าที่สมดุลงานในระบบให้อัตโนมัติ และจัดการหน่วยความจำของระบบทั้งหมดอย่างมีประสิทธิภาพ อันมีผลทำให้ระบบของ MOSIX สามารถขยายได้และมีประสิทธิภาพสูง การทำงานของ MOSIX จะเป็นไปโดยอัตโนมัติทุกอย่างโดยที่ผู้ใช้งานหรือผู้เขียนโปรแกรมไม่ต้องศึกษาเพิ่มเติมมากนัก ก็สามารถเขียนโปรแกรมให้ทำงานบนระบบ คลัสเตอร์ได้อย่างมีประสิทธิภาพ

  การทำงานของระบบจะไม่ได้เป็นการควบคุมจากศูนย์กลาง (Decentralized) เหมือนกับ BPROC โดยจะไม่มีการแบ่งว่าโหนดไหนเป็น Master หรือ Slave Node แต่ทุกโหนดสามารถกระจายงานข้ามไปมาระหว่างกันได้อย่างสะดวกและง่ายดาย

  ตัวอย่างโปรแกรม Shell Script ของ UNIX ที่ทำการทดสอบบนระบบคลัสเตอร์โดยใช้ OpenMOSIX (http://openmosix.sf.net) เป็นคลัสเตอร์มิดเดิลแวร์

```
1:  #!/bin/bash
    for x in 1 2 3 4
    do
      awk 'BEGIN {for(i=0;i<10000;i++)for(j=0;j<10000;j++);}' &
5:  done
```

และรูปที่ 5.4 คือ กราฟแสดงการกระจายงานระหว่างที่ทำการประมวลผล

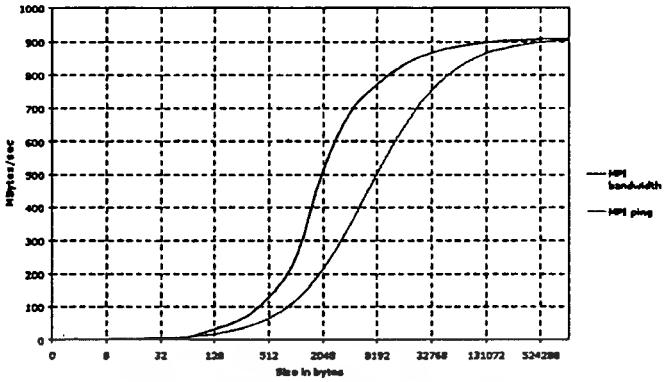


รูปที่ 5.4 กราฟแสดงการกระจายงานระหว่างที่ประมวลผลในระบบคลัสเตอร์มิดเดิลแวร์

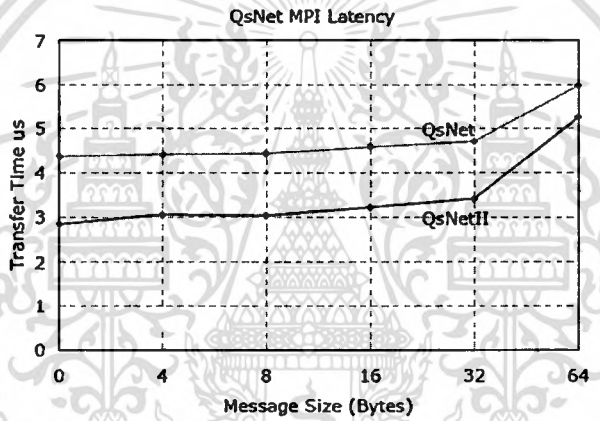
5.6.4 การเชื่อมต่อเครือข่าย

การสื่อสารระหว่างแต่ละเครื่องในระบบคลัสเตอร์จะทำผ่านระบบเครือข่ายความเร็วสูงซึ่งอุปกรณ์เครือข่ายแต่ละชนิดจะมีความเร็วและราคาแตกต่างกันไป ตัวอย่างอุปกรณ์เครือข่ายที่นิยมนำมาใช้ในการสร้างระบบคลัสเตอร์ มีดังนี้

- Ethernet (http://www.10gea.org) ในปัจจุบันอุปกรณ์ Ethernet นั้นได้ถูกพัฒนาให้มีความเร็วในการรับส่งข้อมูลสูงมากขึ้นจนถึงระดับ Gigabit Ethernet หรือ 10 Gigabit Ethernet คือ มีความเร็วในการส่งผ่านข้อมูลประมาณ 1-10 พันล้านบิตต่อวินาที
- Myrinet (http://www.myri.com) มีความเร็วในการส่งผ่านข้อมูลประมาณ 2 พันล้านบิตต่อวินาที (Gigabit per second) และมีค่า Latency Time ต่ำกว่าเครือข่ายแบบ Ethernet มาก แต่มีราคาแพงกว่า Fast Ethernet
- Quadrics (http://www.quadrics.com) มีความเร็วในการส่งข้อมูลอยู่ที่ 340-900 MB/second หรือประมาณ 2.65-7 Gbit/sec และค่า Latency Time มีค่าต่ำมาก



รูปที่ 5.5 กราฟแสดงความเร็วในการส่งข้อมูลของเครือข่ายของ Quadrics



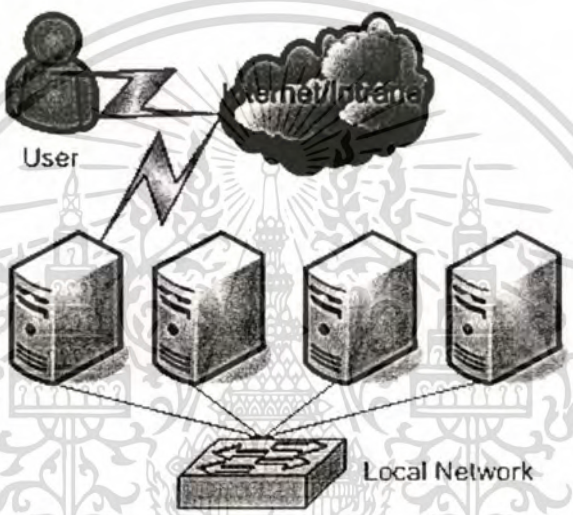
รูปที่ 5.6 กราฟแสดงค่า Latency Time ของเครือข่ายของ Quadrics

- InfiniBand (http://www.infinibandta.org) เป็นเทคโนโลยีที่มีความเร็วในการสื่อสารข้อมูลสูงมากถึง 5 Gbit/sec สำหรับการ์ดรุ่น 1x และ 20 Gbit/sec สำหรับการ์ดรุ่นที่มีความเร็ว 4x อีกทั้งค่า Latency Time มีค่าต่ำมาก น้อยกว่า 10 microsecond

สำหรับรูปแบบวิธีการเชื่อมต่อเครือข่าย (Network Topology) ของระบบคลัสเตอร์นั้นมีหลายรูปแบบที่ถูกนำเสนอไว้ ไม่ว่าจะเป็นการเชื่อมต่อแบบวงแหวน การเชื่อมต่อแบบ cube และ HyperCube (Lester, 1993) ซึ่งรูปแบบการเชื่อมต่อเหล่านี้ต้องมีสายสัญญาณเชื่อมต่อระหว่างโหนดค่อนข้างมาก ทำให้ต้องมีการลงทุนทางด้านเครือข่ายค่อนข้างสูง ดังนั้นรูปแบบของเครือข่ายแบบนี้จะไม่เหมาะกับการนำไปใช้งานในระบบจริง ดังนั้นในงานวิจัยนี้จึงนำเสนอเพียงแค่สองรูปแบบที่เป็นไปได้ในทางปฏิบัติและไม่ต้องลงทุนเกี่ยวกับเครือข่ายมากนัก ดังนี้

- การเชื่อมต่อโดยผ่านเกตเวย์

การเชื่อมต่อคลัสเตอร์ด้วยวิธีนี้จะเป็นดังภาพที่ 5.7 โดยจะมีคอมพิวเตอร์เครื่องหนึ่งที่ทำหน้าที่เป็นทางเข้าหรือทางออกหรือเกตเวย์ (Gate way) ให้แก่ระบบทั้งหมด เมื่อผู้ใช้ติดต่อเข้ามาในระบบจะต้องทำการติดต่อกับเครื่องที่เป็นเกตเวย์ ส่วนเครื่องอื่น ๆ จะทำงานอยู่เบื้องหลังเท่านั้น ทำให้วิธีนี้มีประโยชน์ในเรื่องของการรักษาความปลอดภัยของระบบ เพราะมีเพียงเครื่องเดียวเท่านั้นที่เชื่อมต่อกับภายนอก เมื่อต้องการรักษาความปลอดภัยก็สนใจเพียงเครื่องที่เป็นทางเข้าของระบบ แต่การเชื่อมต่อวิธีนี้ก็รับประกันไม่ได้เสมอไปนักว่าจะไม่ถูกบุกรุกจากภายนอก การเชื่อมต่อสามารถแสดงได้ดังนี้

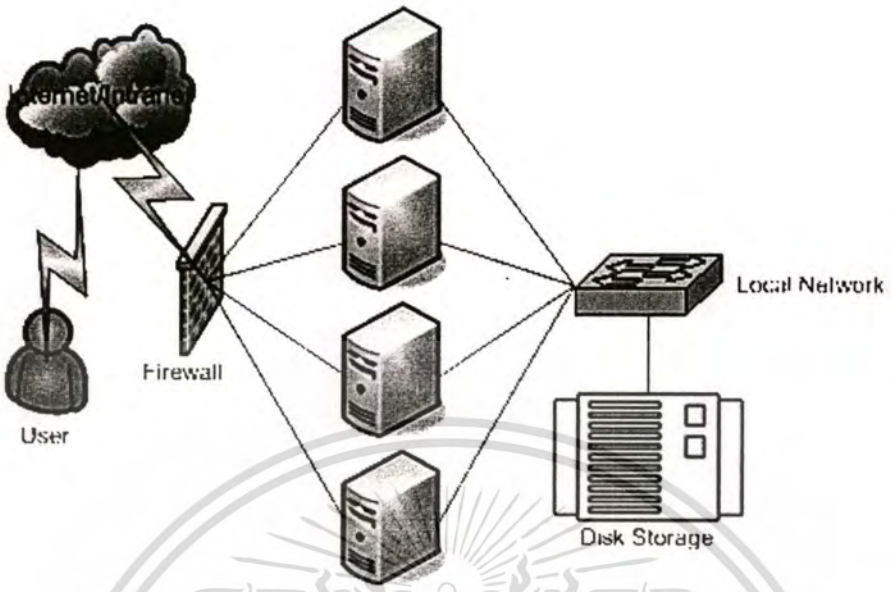


รูปที่ 5.7 การเชื่อมต่อกับเครือข่ายภายนอกผ่านเกตเวย์

การเชื่อมต่อแบบนี้มีข้อดีในส่วนของการดูแลเรื่องความปลอดภัย แต่จะไม่ค่อยเหมาะนักกับงานที่เป็นด้านการให้บริการหรือคลัสเตอร์ที่ต้องการความคงทนสูง (High Availability Cluster) อย่างเช่นเครื่องแม่ข่ายให้บริการเว็บ หรืออีเมล์ ดังนั้นจึงต้องใช้รูปแบบการเชื่อมต่อตามวิธีการในหัวข้อถัดไป

- การเชื่อมต่อแบบทุกเครื่องเชื่อมกับภายนอก

การเชื่อมต่อเครือข่ายแบบนี้ ทุกเครื่องจะทำการเชื่อมต่อกับเครือข่ายภายนอก ทำให้เครื่องจากภายนอกสามารถเข้าถึงทรัพยากรของแต่ละเครื่องได้โดยตรง โดยจะมีประโยชน์ในกรณีที่เครื่อง คลัสเตอร์เหล่านี้ทำหน้าที่ให้บริการงานต่าง ๆ เช่นเครื่องแม่ข่ายของงานเว็บไซต์ การเชื่อมต่อแบบนี้มักจะนำเอาไฟร์วอล (Firewall) มาวางไว้ด้านหน้าของระบบอีกชั้น เพื่อเป็นการป้องกันการบุกรุกจากภายนอกได้ในระดับหนึ่ง ดังรูป



รูปที่ 5.8 การเชื่อมต่อแบบทุกเครื่องเชื่อมต่อกับเครือข่ายภายนอก

และเนื่องจากระบบนี้มักจะถูกใช้เป็นเครื่องแม่ข่ายที่ให้บริการอันเดียวกัน ดังนั้นจึงต้องมีที่เก็บข้อมูลรวมศูนย์ไว้ที่เดียว โดยอาจจะเป็น Storage ตามมาตรฐานทั่วไป เช่น NAS (Network Attach Server) เป็นต้น

สำหรับโครงสร้างพื้นฐานของระบบคลัสเตอร์มีทั้งหมดสี่ส่วนคือ โครงสร้างทางด้านฮาร์ดแวร์ ระบบปฏิบัติการ คลัสเตอร์มิดเดิลแวร์ และการเชื่อมต่อเครือข่าย ซึ่งได้กล่าวถึงโดยละเอียดแล้วในหัวข้อนี้ แต่ว่าการที่ระบบคลัสเตอร์จะทำงานได้อย่างมีประสิทธิภาพมากขึ้น หรือการที่จะทำให้ผู้ใช้งานมีความสะดวกในการเขียนโปรแกรมเพื่อทำงานบนระบบเหล่านี้ได้นั้น ต้องมีการกล่าวถึงรายละเอียดเพิ่มเติมของส่วนที่เป็นชุดคำสั่งทางซอฟต์แวร์ (Software Library) และเครื่องมือต่าง ๆ ดังในหัวข้อถัดไป

5.7 เครื่องมือและโปรแกรมอัตถะประโยชน์

ในหัวข้อนี้จะเป็นการกล่าวถึงเครื่องมือต่าง ๆ ที่ถูกใช้ในการสร้างและการเขียนโปรแกรมบนคอมพิวเตอร์ระบบคลัสเตอร์ โดยจะกล่าวถึงโปรแกรมจัดการสภาวะการโปรแกรมแบบขนาน หรือ Parallel Programming Environment ซึ่งในระบบคลัสเตอร์ส่วนใหญ่มักจะเลือกใช้โปรแกรมการสื่อสารระหว่างโหนดโดยการส่งข้อความหรือ Message Passing Interface นั่นเอง หลังจากนั้นจะเป็นการกล่าวถึงตัวแปรภาษาที่สนับสนุนการทำงานบนระบบคลัสเตอร์ และสุดท้ายจะได้กล่าวถึงชุดคำสั่งทางคณิตศาสตร์ที่ถูกพัฒนาขึ้นมาเพื่ออำนวยความสะดวกในการเขียนโปรแกรม ให้สามารถทำได้ง่ายและสะดวกสบายมากยิ่งขึ้น ดังมีรายละเอียดต่าง ๆ ต่อไปนี้

5.7.1 โปรแกรมสื่อสารระหว่างโหนดโดยการส่งข้อความ

ในรูปที่ 5.3 ซึ่งแสดงส่วนประกอบทั้งหมดของระบบคลัสเตอร์ ในชั้นของสภาวะแวดล้อมการโปรแกรมแบบขนาน (Parallel Programming Environment) นั้น จะเป็นชั้นของซอฟร์แวร์ที่จะคอยอำนวยความสะดวกพื้นฐานให้แก่ผู้ใช้งานระบบ ทั้งในส่วนของชุดคำสั่ง (Library) พื้นฐานที่จำเป็นในการสื่อสารระหว่างโหนด การส่งค่าตัวแปร การส่งค่าข้อมูลแบบเจาะจง การส่งข้อมูลแบบกระจายหรือบรอดแคสต์ (Broadcasting) และอื่น ๆ อีกทั้งมีการเตรียมซอฟต์แวร์สำหรับการติดต่อ (Software Interface) กับตัวแปรภาษาหรือคอมไพเลอร์ และคำสั่งใช้งานทั่วไป

สำหรับมาตรฐานของสภาวะแวดล้อมการโปรแกรมแบบขนานที่ได้รับความนิยม และมีการพัฒนาชุดคำสั่งตามมาตรฐานขึ้นมาใช้งานจริงมากที่สุดคือ มาตรฐานการส่งผ่านข้อความ หรือ Message Passing Interface ซึ่งมีชุดคำสั่งที่สร้างขึ้นมาใช้งานและได้รับความนิยมมากคือ MPICH (http://www-unix.mcs.anl.gov/mpi/mpich/) และ LAM (http://www.lam-mpi.org)

โดยทั้ง MPICH และ LAM จะเป็นซอฟแวร์ไลบรารี (Software Library) ที่ถูกพัฒนาขึ้นมาตามมาตรฐาน MPI (http://www.mpi-forum.org/) เวอร์ชั่น 1.1 หรือสูงกว่า และจะมีการเตรียมฟังก์ชั่นพื้นฐานต่าง ๆ มากมาย รวมทั้งมีคำสั่งใช้งานพื้นฐานให้แก่ผู้ใช้เพื่อทำการคอมไพล์ (Compile) โปรแกรมที่เขียนขึ้นมาและคำสั่งใช้งานอื่น ๆ เพื่อติดต่อและสั่งงานกับระบบคลัสเตอร์

ซึ่งรายละเอียดเกี่ยวกับรายละเอียดของการเขียนโปรแกรมแบบขนาน และการใช้งานคำสั่งของ MPICH นั้น จะกล่าวโดยละเอียดในบทถัดไป

5.7.2 คอมไพเลอร์

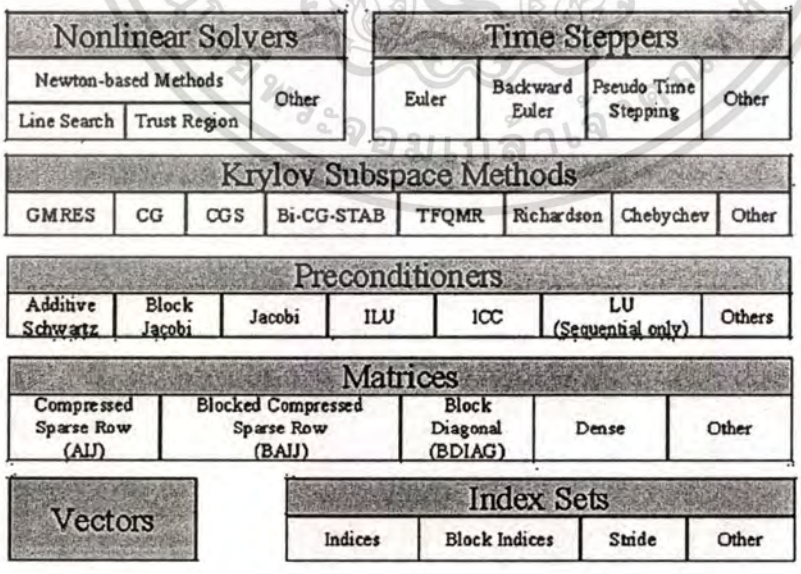
ภาษาคอมพิวเตอร์ที่ถูกนำมาใช้ในการเขียนโปรแกรมบนระบบคลัสเตอร์ ที่ได้รับความนิยมโดยส่วนใหญ่คือ C/C++ และ FORTRAN เพราะเนื่องจากมีผู้พัฒนาฟังก์ชั่นและไลบรารีต่าง ๆ ขึ้นมากมาย ทั้งตัว MPICH, LAM หรือ PVM (Parallel Virtual Machine) เอง ก็มีฟังก์ชั่นสนับสนุนภาษาเหล่านี้โดยตรง

เมื่อจะทำการสร้างโปรแกรมขนานจะต้องมีการคอมไพล์โปรแกรมที่สร้างขึ้นมา และเนื่องจากฟังก์ชั่นที่ถูกเตรียมไว้ให้โดย MPICH หรือ LAM เป็น C/C++ หรือ FORTRAN ดังนั้นในระบบคลัสเตอร์จึงต้องมีตัวแปรภาษาเหล่านี้อยู่ในระบบก่อนแล้ว คอมไพเลอร์ที่นิยมนำมาใช้ตัวอย่างเช่น GCC ของ GNU Project (http://gcc.gnu.org) เป็นต้น

5.7.3 ชุดคำสั่งทางคณิตศาสตร์

มีผู้พัฒนาชุดคำสั่งทางคณิตศาสตร์ขึ้นมาเพื่อความสะดวกในการสร้างโปรแกรมการคำนวณต่าง ๆ หลายอย่าง ซึ่งชุดคำสั่งเหล่านี้ได้ช่วยให้การสร้างโปรแกรมขึ้นมาทำได้ง่ายขึ้นเพราะไม่ต้องเขียนโปรแกรมเองทั้งหมด และคำสั่งเหล่านี้ก็สามารถใช้งานร่วมกับระบบคลัสเตอร์ได้โดยไม่มีปัญหา และบางชุดคำสั่งยังสนับสนุนการทำงานแบบขนานในตัวชุดคำสั่งเลยอีกด้วย ตัวอย่างชุดคำสั่งเหล่านี้เช่น

- PETSc (http://acts.nersc.gov/petsc/) เป็นไลบรารีที่ถูกสร้างขึ้นมาเพื่อแก้ปัญหาทางด้านนิวเมอริคอล (Numerical Analysis) โดยเฉพาะสมการพวก Partial differential equation สมการเชิงเส้น และสมการไม่เชิงเส้น เป็นต้น



รูปที่ 5.9 ส่วนประกอบของชุดคำสั่ง PETSc

- PLAPACK (http://www.cs.utexas.edu/users/plapack/) หรือ Parallel Linear Algebra Package เป็นชุดคำสั่งที่สนับสนุนการประมวลผลแบบขนาน เพราะถูกสร้างขึ้นมาบนพื้นฐานของ MPI และชุดคำสั่งนี้ถูกสร้างขึ้นมาเพื่อการแก้ปัญหาของสมการพีชคณิตเชิงเส้น และการคำนวณเกี่ยวกับเมตริกซ์
- ScaLAPACK (http://www.netlib.org/scalapack/) เป็นชุดคำสั่งที่ถูกพัฒนาขึ้นมาเพื่อแก้ปัญหาเกี่ยวกับการคำนวณเมตริกซ์ และพีชคณิตเชิงเส้น การหาค่า eigenvalue เป็นต้น

5.8 การจัดการระบบคลัสเตอร์

ระบบคลัสเตอร์ที่ดีนั้นนอกจากจะทำงานได้อย่างมีประสิทธิภาพ และคำนวณงานต่าง ๆ ได้อย่างรวดเร็วแล้ว จะต้องมีการจัดการที่ดี สามารถรองรับการทำงานของผู้ใช้งานหลาย ๆ คน รวมทั้งสามารถตรวจสอบสถานะของแต่ละโหนดได้อีกด้วย การจัดการระบบที่ดีจะทำให้เกิดความสะดวกแก่ผู้ใช้งานโดยส่วนใหญ่รวมทั้งทำให้เกิดประสิทธิภาพในการทำงานมากขึ้นด้วย การจัดการระบบนั้นจะมีอยู่ด้วยกันสามส่วนคือ

- **การจัดการเกี่ยวกับผู้ใช้** (Users Management) จะเป็นการเพิ่มหรือลบผู้ใช้ในระบบ ให้สิทธิ์ต่าง ๆ ที่เหมาะสมกับผู้ใช้งานแต่ละคน กำหนดความเหมาะสมให้เกิดขึ้นสำหรับผู้ใช้งาน และเนื่องมาจากเหตุผลที่ว่าคอมพิวเตอร์แต่ละโหนดที่เชื่อมต่ออยู่ในคลัสเตอร์นั้นสามารถทำงานได้โดยอิสระ และมีระบบปฏิบัติการเป็นของตัวเอง การจัดการเรื่องผู้ใช้งานจึงเป็นสิ่งจำเป็นอันดับต้น ๆ เพราะเมื่อทำการเพิ่มหรือลบผู้ใช้ออกจากระบบ ทุกโหนดจะต้องรู้และรับทราบการเปลี่ยนแปลงนั้น อีกอย่างเมื่อผู้ใช้ทำการล็อกอินเข้ามายังระบบคงไม่ดีแน่หากผู้ใช้ต้องทำการล็อกอินไปยังทุกเครื่องก่อนจึงจะสามารถใช้งานได้ ดังนั้นทางที่ดีควรจะทำให้ผู้ใช้ล็อกอินเข้าสู่ระบบเพียงครั้งเดียวแล้วสามารถใช้ทรัพยากรภายในระบบได้อย่างเหมาะสม
- **การจัดการคิวงาน** (Jobs scheduler) เมื่อมีผู้ใช้งานระบบหลาย ๆ คน อาจจะมีการแย่งกันใช้ทรัพยากรของระบบที่มีอยู่ ดังนั้นจะต้องมีระบบหรือวิธีการที่จะมาเรียงลำดับหรือช่วยจัดการงานต่าง ๆ ตามความสำคัญหรือตามลำดับของงานที่ส่งให้มาประมวลผลในระบบ ระบบการจัดการคิวงานที่ดีนั้นต้องสามารถตรวจสอบทรัพยากรที่มีอยู่ของระบบทั้งหมดได้ เช่น มีโหนดออนไลน์ (Online) อยู่กี่โหนด หรือออฟไลน์ (Offline) ไปแล้วกี่โหนด ผู้ใช้คนไหนมีสิทธิ์ใช้หน่วยประมวลผลได้มากน้อยแค่ไหน ใช้หน่วยความจำได้เท่าไหร่ โปรแกรมของผู้ใช้สามารถทำงานได้ตั้งแต่เวลาเท่าไหร่ถึงเท่าไหร่ ซึ่งสิ่งเหล่านี้ล้วนแต่เป็นสิ่งที่ต้องมีในตัวจัดการคิวงาน

ตัวจัดการคิวงานนี้มีเครื่องมือให้ใช้บ้างบางส่วนนั่นคือ ระบบ Batch queues ซึ่งที่นิยมนำมาใช้งานเช่น OpenPBS (Open Portable Batch System)

- **การตรวจสอบและการจัดการ** (Monitoring and Management) ระบบคลัสเตอร์จะต้องทำการตรวจสอบสถานะทำงานได้ เช่น สถานะการใช้งานหน่วยประมวลผล ปริมาณการใช้หน่วยความจำ และสถานะของเครื่องแต่ละโหนด เพื่อจะได้ทำการตรวจสอบและแก้ไขได้อย่างถูกต้อง ระบบการตรวจสอบสถานะมีเครื่องมือ เช่น Ganglia (http://ganglia.sourceforge.net/) เป็นต้น ซึ่งเครื่องมือเหล่านี้ทำให้จัดการระบบได้ง่ายขึ้น เพราะมีการตรวจสอบระบบเป็นระยะ ส่วนระบบการจัดการนั้นต้องมีระบบการจัดการที่ดีเพื่อให้ระบบสามารถทำงานต่อเนื่องยาวนาน และมีประสิทธิภาพสูงสุด

5.9 สรุป

ในบทนี้ได้กล่าวถึงความรู้เบื้องต้นเกี่ยวกับการประมวลผลแบบขนาน รูปแบบและโครงสร้างของการประมวลผลแบบขนาน ทั้งนี้ได้กล่าวถึงรายละเอียดของคอมพิวเตอร์ความเร็วสูงแบบคลัสเตอร์ กริดคอมพิวเตอร์ และเครือข่ายคอมพิวเตอร์ประมวลผลแบบ P2P แล้วค่อยเจาะลึกลงในรายละเอียดของส่วนประกอบต่าง ๆ ของระบบคลัสเตอร์ ทั้งในส่วนของโครงสร้างพื้นฐานของระบบคลัสเตอร์ เครื่องมือและโปรแกรมอัตถะประโยชน์ต่าง ๆ และสุดท้ายคือการจัดการระบบให้ทำงานได้อย่างมีประสิทธิภาพ สามารถรองรับการใช้งานจากผู้ใช้จำนวนมาก ๆ ได้ ตลอดทั้งมีวิธีการที่ทำให้สามารถตรวจสอบระบบได้ง่ายขึ้นด้วย

ตลอดทั้งบทนี้ได้แนะนำให้เห็นรายละเอียดว่าระบบคลัสเตอร์ประกอบด้วยส่วนประกอบใดบ้าง และมีเครื่องมือและวิธีการในการจัดการระบบอย่างไร ส่วนในบทต่อไปจะเป็นการแนะนำให้รู้จักถึงขั้นตอนและวิธีการของการโปรแกรมแบบขนาน และการทำงานบนระบบคลัสเตอร์

บทที่ 6
โปรแกรมแบบขนาน

6.1 บทนำ

หลักการประมวลผลขนานแบบต่าง ๆ ทั้งระบบที่เป็นแบบหลายหน่วยประมวลผลหรือหลายเครื่องประมวลผล จะไม่สามารถทำงานได้อย่างมีประสิทธิภาพ ถ้าหากโปรแกรมที่นำมาใช้งานบนระบบเหล่านี้ไม่ได้ทำการออกแบบและสร้างขึ้นมาอย่างเหมาะสมหรือยังคงใช้วิธีการในการโปรแกรมแบบลำดับขั้น (Sequential Algorithm) เหมือนกับโปรแกรมอื่น ๆ ทั่วไป ดังนั้นในบทนี้จึงจะเป็นการกล่าวถึงความสำคัญและวิธีการในการออกแบบและสร้างโปรแกรมแบบขนาน โดยใช้เทคนิคและวิธีการต่าง ๆ เพื่อให้เกิดประสิทธิภาพในการทำงานของโปรแกรมมากยิ่งขึ้น

ในบทนี้จะเป็นการกล่าวถึงเนื้อหาทั้งหมดของการโปรแกรมแบบขนาน นับตั้งแต่กลวิธีที่ถูกนำมาใช้ในการโปรแกรม รูปแบบและวิธีการพัฒนาโปรแกรม การออกแบบวิธีการการทำงานของโปรแกรม ทั้งการแบ่งงาน การรวมกลุ่มงาน การสมดุลงาน และการสื่อสารระหว่างโปรเซสในขณะทำงาน หลังจากนั้นจะเป็นการกล่าวถึงการออกแบบวิธีการสร้างภาพเชิงปริมาตรทางการแพทย์บนระบบคลัสเตอร์โดยใช้การโปรแกรมแบบขนาน

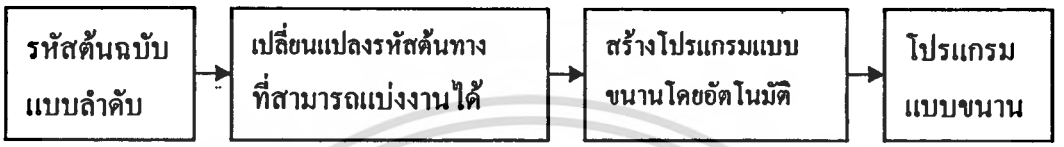
6.2 รูปแบบของการพัฒนาโปรแกรมแบบขนาน

ในหัวข้อนี้จะเป็นการกล่าวถึงรูปแบบหรือแนวทางในการพัฒนาโปรแกรมขนานแบบต่าง ๆ ก่อนที่จะกล่าวถึงทฤษฎีและการออกแบบเพื่อสร้างโปรแกรมขนานในหัวข้อถัดไป โดยลำดับแรกในการพัฒนาโปรแกรมขนานนั้น ส่วนใหญ่ผู้เขียนโปรแกรมมักจะสร้างโปรแกรมในแบบลำดับ (Sequential Program) ให้สามารถทำงานได้อย่างถูกต้องก่อน จากนั้นจึงจะพัฒนาโปรแกรมแบบลำดับนี้ไปเป็นโปรแกรมขนานได้ตามวิธีเหล่านี้

6.2.1 การสร้างโปรแกรมแบบขนานโดยอัตโนมัติ

การสร้างโปรแกรมด้วยวิธีนี้เป็นวิธีที่ง่ายที่สุด แต่ก็จะมีประสิทธิภาพต่ำที่สุด โดยหน้าที่ของการสร้างโปรแกรมแบบขนานจะเป็นหน้าที่ของตัวแปรภาษาซึ่งตัวแปรภาษาจะตรวจสอบรหัสต้นฉบับ (Source Code) ซึ่งอาจประกอบด้วยส่วนของรหัสที่มีการวนซ้ำ และการประมวลผลกับข้อมูลที่ซ้ำๆกัน

โดยตัวแปรภาษานี้จะทำการเปลี่ยนรหัสต้นฉบับไปเป็นรหัสที่สนับสนุนการประมวลผลแบบขนาน จากนั้นใช้ตัวแปรภาษาเปลี่ยนรหัสที่ได้ไปเป็นโปรแกรมแบบขนานเองโดยอัตโนมัติ แต่ข้อจำกัดของการสร้างโปรแกรมด้วยวิธีนี้จะขึ้นอยู่กับเทคโนโลยีที่ตัวแปรภาษาใช้ในการเปลี่ยนรหัสต้นฉบับเป็นรหัสที่สนับสนุนการประมวลผลแบบขนาน ขั้นตอนการทำงานของการสร้างโปรแกรมขนานโดยอัตโนมัติแสดงดังรูปที่ 6.1



รูปที่ 6.1 ขั้นตอนการทำงานของการสร้างโปรแกรมแบบขนานโดยอัตโนมัติ

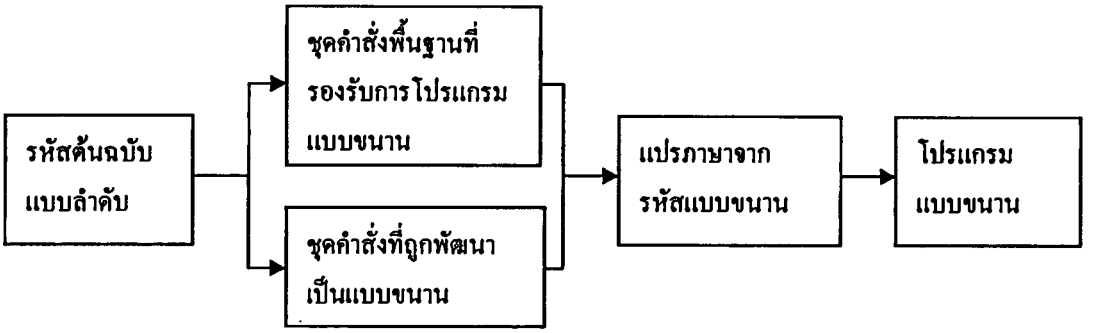
6.2.2 การสร้างโปรแกรมโดยใช้ชุดคำสั่งแบบขนาน

เป็นวิธีการที่มีประสิทธิภาพมากกว่าวิธีแรก โดยเมื่อทำการสร้างโปรแกรมจะใช้ชุดคำสั่งที่ถูกพัฒนาขึ้นมาให้สนับสนุนการทำงานแบบขนานแล้ว เช่น ชุดคำสั่งของ ScaLAPACK หรือ PLAPACK ซึ่งชุดคำสั่งเหล่านั้นสนับสนุนการประมวลผลแบบขนานอยู่แล้ว ฟังก์ชั่นที่ชุดคำสั่งเหล่านี้เตรียมไว้ให้ใช้งาน เช่น ชุดคำสั่งของการหาผลเฉลยสมการต่าง ๆ แบบขนาน และการคูณเมตริกแบบขนาน เป็นต้น

ชุดคำสั่งที่สนับสนุนการทำงานแบบขนานนี้จะมีอยู่สองกลุ่มคือ

- กลุ่มของชุดคำสั่งพื้นที่รองรับหรือทำให้เกิดสภาพแวดล้อมการประมวลผลแบบขนาน ตัวอย่างเช่น ฟังก์ชั่นที่ถูกเตรียมไว้โดย MPICH หรือ LAM ซึ่งจะเป็นคำสั่งหรือฟังก์ชั่นพื้นฐานที่ทำให้เกิดการประมวลผลแบบขนานได้ เพราะมีคำสั่งที่ทำให้เกิดการสื่อสารระหว่างโหนดต่อโหนด การส่งข้อความถึงโหนดอื่น ๆ ในระบบได้ และอื่น ๆ อีกมาก
- กลุ่มของชุดคำสั่งที่ได้รับการพัฒนาให้เป็นการประมวลผลแบบขนาน เช่น ScaLAPACK หรือ PLAPACK ซึ่งเป็นชุดคำสั่งทางคณิตศาสตร์ที่ได้รับการพัฒนาให้ทำงานแบบขนานได้โดยเรียกใช้ชุดคำสั่งของ MPI อีกที

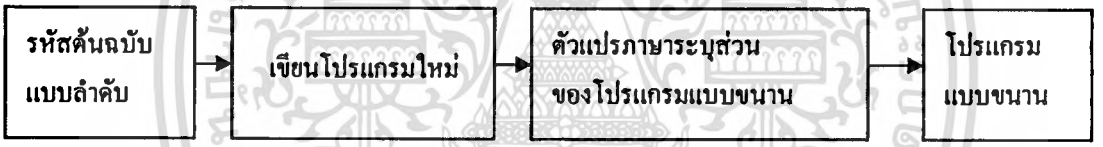
ในวิทยานิพนธ์ฉบับนี้ใช้วิธีการสร้างโปรแกรมแบบขนานโดยใช้เพียงแค่คำสั่งพื้นฐานเพื่อให้เกิดสภาพแวดล้อมในการโปรแกรมแบบขนานขึ้นเท่านั้น แต่ไม่ได้ใช้ชุดคำสั่งขนานแบบอื่นเข้ามาช่วย



รูปที่ 6.2 ขั้นตอนการสร้างโปรแกรมโดยใช้ชุดคำสั่งแบบขนาน

6.2.3 การสร้างโปรแกรมแบบขนานด้วยตัวเอง

การสร้างโปรแกรมแบบขนานด้วยวิธีนี้เป็นแบบที่ยืดหยุ่นมากที่สุด ผู้สร้างโปรแกรมสามารถเลือกตัวแปรภาษาใดก็ได้ที่ต้องการ รวมถึงสามารถเลือกรูปแบบและวิธีการที่ใช้ในการสื่อสาร แต่วิธีนี้นับเป็นวิธีที่ยากและเสียเวลามากที่สุด ขั้นตอนการสร้างโปรแกรมแบบขนานด้วยตัวเองสามารถแสดงได้ดังรูปที่ 6.3



รูปที่ 6.3 ขั้นตอนการสร้างโปรแกรมแบบขนานด้วยตัวเอง

6.3 กลวิธีการโปรแกรมแบบขนาน

การสร้างโปรแกรมแบบลำดับขั้น (Sequential Algorithm) บนเครื่องคอมพิวเตอร์ที่มีเพียงหน่วยประมวลผลเดียวเป็นเรื่องที่ไม่ยุ่งยากหรือสลับซับซ้อนมากมายนัก เพราะลำดับการทำงานของโปรแกรมเป็นไปอย่างตรงไปตรงมากับความต้องการหรือทฤษฎีและวิธีการที่อธิบายไว้แล้ว การทำงานของโปรแกรมใช้เพียงหน่วยประมวลเดียว การเข้าใช้งานหน่วยความจำก็สามารถที่เข้าถึงและใช้งานได้โดยไม่มีปัจจัยอย่างอื่นเข้ามาเกี่ยวข้อง การสื่อสารระหว่างโปรเซสที่ทำงานอยู่ก็มักจะไม่เกิดขึ้นในวิธีการการโปรแกรมแบบนี้ ทำให้การออกแบบโปรแกรมสามารถทำได้ง่าย ซึ่งแตกต่างจากกลวิธีการสร้างโปรแกรมแบบขนาน เพราะปัจจัยในเรื่องการใช้งานหน่วยประมวล การเข้าใช้งานหน่วยความจำและการสื่อสารระหว่างโปรเซสจะเริ่มเข้ามามีบทบาทในการทำงานของโปรแกรมมากขึ้น อีกทั้งถ้าหากต้องการให้โปรแกรมทำงานได้อย่างมีประสิทธิภาพแล้วต้องคำนึงถึงปัจจัยอื่น ๆ ที่มาเกี่ยวข้องด้วย

ดังนั้นในการสร้างโปรแกรมแบบขนานจึงได้มีผู้นำเสนอเทคนิคและวิธีการต่าง ๆ หลายอย่างในการสร้างโปรแกรมขึ้นมา เพื่อให้การทำงานของโปรแกรมเป็นไปอย่างเหมาะสมและมีประสิทธิภาพ Lester (1993:13) ได้นำเสนอวิธีการต่าง ๆ โดยรวมไว้ทั้งหมด 6 วิธีการดังนี้

- **การรวมกลุ่มชุดข้อมูลที่มีความเหมือนกัน** (Data Parallelism) ซึ่งเป็นวิธีการที่สามารถใช้งานได้โดยทั่วไป หลักการของการสร้างโปรแกรมแบบนี้ จะทำการรวมเอาข้อมูลอินพุท (Input) ที่มีความเหมือนกันเข้าไว้ด้วยกันเป็นกลุ่ม หลังจากนั้นทำการประมวลผลแยกเป็นกลุ่ม ๆ ไป เมื่อข้อมูลที่อยู่ในกลุ่มเดียวกันมีความสัมพันธ์กันแล้ว ทำให้เกิดความง่ายในการคำนวณมากยิ่งขึ้น
- **การแบ่งกลุ่มข้อมูลที่อยู่ใกล้กันไว้ด้วยกัน** (Data Partitioning) ข้อมูลอินพุทที่อยู่ใกล้กันจะมีความสัมพันธ์กันอยู่แล้วส่วนหนึ่ง ดังนั้นจึงสามารถที่จะรวมกลุ่มชุดของข้อมูลที่อยู่ใกล้กันไว้ด้วยกันได้ วิธีนี้ทำให้เกิดความสะดวกในการแบ่งกลุ่มของข้อมูลมากยิ่งขึ้น เพราะเพียงพิจารณาข้อมูลที่อยู่ติดกัน แยกเป็นส่วน ๆ เท่านั้น เมื่อทำการแยกข้อมูลได้แล้วก็ทำการประมวลผลข้อมูลกลุ่มต่าง ๆ ไปพร้อมกันตามหลักการของการประมวลผลแบบขนาน

  วิธีการนี้เหมาะกับการคำนวณบนระบบแบบหลายเครื่องประมวลผลอย่างเช่นระบบคลัสเตอร์เป็นอย่างมาก เพราะแต่ละเครื่องจะมีหน่วยความจำเป็นของตัวเองและเมื่อทำการคำนวณแต่ละเครื่องหรือแต่ละโหนดจะเกี่ยวข้องกับชุดข้อมูลที่ถูกแบ่งออกมาอย่างชัดเจน การเกี่ยวข้องกันกับชุดข้อมูลข้างเคียงมีน้อย ทำให้เกิดการแลกเปลี่ยนข้อมูลระหว่างโหนดน้อยลงตามไปด้วย มีผลทำให้เวลาที่สูญเสียไปในการสื่อสารน้อย เวลารวมในการทำงานก็ลดลงด้วย
- **การทำงานแยกกันโดยอิสระ** (Relaxed Algorithm) ถ้าหากแต่ละหน่วยประมวลผลในระบบสามารถทำงานแยกจากกันได้โดยอิสระ โดยที่ไม่ต้องรอข้อมูลผลลัพธ์จากหน่วยประมวลผลหรือโปรเซสข้างเคียง และไม่ต้องสื่อสารหรือตรวจทานความถูกต้องกับหน่วยประมวลผลอื่น ๆ ทำให้ไม่ต้องสูญเสียเวลาในการสื่อสารหรือการรอในระหว่างการทำงานเลย หลักการนี้สามารถนำไปประยุกต์ใช้ได้ทั้งระบบแบบหลายหน่วยประมวลและหลายเครื่องประมวลผล
- **การทำงานพร้อมกันในแต่ละรอบการทำงาน** (Synchronous Iteration) ในการคำนวณบางประเภทอาจจะต้องมีการทำงานไปพร้อม ๆ กันในแต่ละรอบการทำงาน วิธีการนี้จะให้แต่ละโปรเซสทำการตรวจสอบการทำงานของโปรเซสข้างเคียงอื่น ๆ ในแต่ละรอบเพื่อให้คำตอบที่ได้มีค่าถูกต้อง การทำงานไปพร้อม ๆ กันจะลดความเร็วในการประมวลผลลงเล็กน้อย และ

เหมาะกับระบบที่เป็นแบบหลายหน่วยประมวลผลมากกว่าระบบที่เป็นแบบหลายเครื่องประมวลผล เพราะจะต้องมีการสื่อสารระหว่างกันค่อนข้างมาก

- **การกระจายการทำงาน** วิธีการนี้จะทำการรวมศูนย์งานไว้ที่เดียวเรียกว่า Work Pool เครื่องอื่น ๆ หรือโหนดอื่น ๆ ในระบบจะถูกมองว่าเป็นผู้ทำงานหรือ Worker ซึ่งผู้ทำงานนี้จะต้องเข้ามารับงานจาก Work Pool ไปทำจนกระทั่งงานที่อยู่ใน Pool หมดลง การทำงานแบบนี้อาจกล่าวง่าย ๆ ว่าเป็นการทำงานแบบรวมศูนย์ เพราะทุก Worker จะทำการติดต่อและร้องของานจากที่เดียวกัน
- **การส่งต่อค่าผลลัพธ์** (Pipelined Computation) เทคนิคนี้ได้อาศัยรูปแบบของการส่งต่อค่าข้อมูลผลลัพธ์จากโปรเซสหนึ่ง ไปยังอีกโปรเซสหนึ่งในระบบเป็นลำดับเรื่อยไป ซึ่งการคำนวณงานบางชนิดต้องใช้รูปแบบการคำนวณงานแบบนี้ การส่งต่อค่าผลลัพธ์จะสามารถใช้ได้ดีทั้งระบบแบบหลายหน่วยประมวลผลและระบบแบบหลายเครื่องประมวลผล

เพื่อให้โปรแกรมทำงานอย่างมีประสิทธิภาพสูงสุด เมื่อจะทำการออกแบบโปรแกรมนั้น Lester (1993:15) ได้เสนอถึงปัญหาต่าง ๆ ที่ควรคำนึงถึง รวมทั้งปัญหาที่ควรหลีกเลี่ยงไม่ให้เกิดขึ้นในโปรแกรม ปัญหาสำคัญ ๆ มีดังนี้

- ความคับคั่งในการเข้าใช้งานหน่วยความจำ (Memory Contention) ปัญหานี้เกิดจากการเข้าใช้ตัวแปรร่วม (Global variable) ในหน่วยความจำพร้อมกันของหน่วยประมวลผล เมื่อหน่วยประมวลผลหนึ่งใช้งานหน่วยความจำอยู่ หน่วยประมวลผลอื่น ๆ จะไม่สามารถเข้าใช้งานได้ ทำให้เกิดการหยุดรอของหน่วยประมวลผลขณะที่กำลังคำนวณ ซึ่งจะทำให้สูญเสียเวลาไปกับการรอนี้ แต่ปัญหานี้มักจะเกิดในระบบหลายหน่วยประมวลผลที่ใช้หน่วยความจำร่วมมากกว่าส่วนระบบแบบหลายเครื่องประมวลผลมักไม่เป็นปัญหา
- การมีส่วนของโปรแกรมที่ทำงานแบบลำดับมากเกินไป (Excessive Sequential Code) ซึ่งอาจจะเกิดจากตอนที่ออกแบบโปรแกรมนั้น ผู้เขียนโปรแกรมอาจจะสร้างโปรแกรมขึ้นมาจากวิธีการแบบลำดับ และยังใช้วิธีการคิดแบบเดิมในการโปรแกรมแบบขนานอยู่ ทำให้โปรแกรมที่ถูกสร้างขึ้นไม่ได้ทำงานไปอย่างพร้อม ๆ กันอย่างแท้จริง และประสิทธิภาพของโปรแกรมก็ลดลง

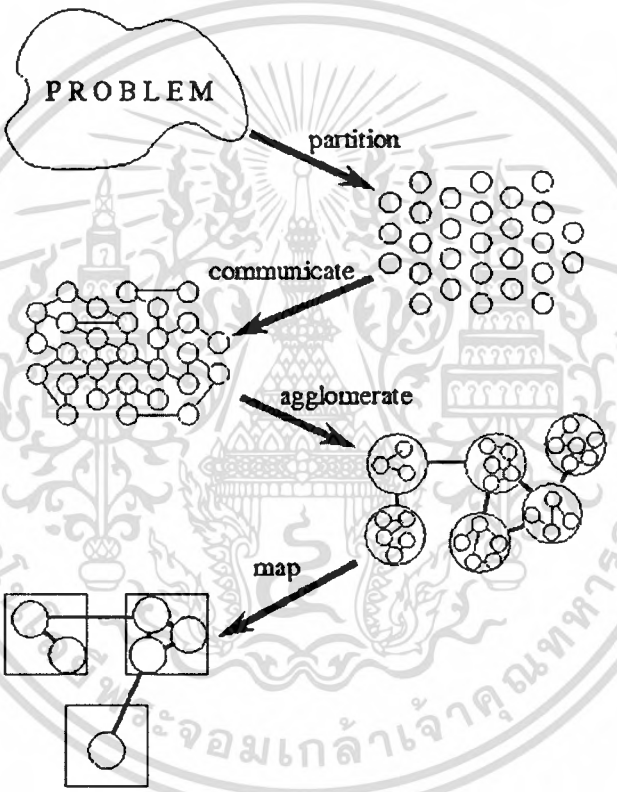
- การสูญเสียเวลาไปในการสร้างโปรเซส (Process Creation Time) ในการทำงานแบบขนานจะต้องมีเวลาส่วนหนึ่งที่สูญเสียไปเพราะการสร้างโปรเซสขึ้นมาใหม่ เพราะฉะนั้นการสร้างโปรเซสควรสร้างขึ้นมาเท่าที่จำเป็นเท่านั้น เพราะบางทีการสร้างโปรเซสขึ้นมามากเกินไปก็ไม่ได้ช่วยทำให้การทำงานของโปรแกรมเร็วขึ้นแต่อย่างใด
- การสูญเสียเวลาในการสื่อสาร (Communication Delay) ปัญหานี้จะเกิดขึ้นกับระบบแบบหลายเครื่องประมวลผลค่อนข้างมาก เพราะแต่ละหน่วยประมวลผลจะต้องทำการติดต่อกันผ่านทางการส่งข้อความข้ามไปมาระหว่างโปรเซสที่ทำงานอยู่ในระบบ ดังนั้นเมื่อทำการออกแบบโปรแกรมแล้วต้องให้มีการสื่อสารระหว่างโปรเซสน้อยที่สุด และขนาดของข้อมูลที่รับและส่งนั้นควรมีขนาดเล็กมากที่สุดด้วยเพื่อให้ เสียเวลาในการสื่อสารน้อยที่สุดนั่นเอง

- การสูญเสียเวลาในการตรวจทานสถานะ (Synchronization Delay) เมื่อมีการออกแบบโปรแกรมให้แต่ละโปรเซสต้องทำการตรวจทานสถานะซึ่งกันและกันแล้ว จะต้องมีเวลาส่วนหนึ่งสูญเสียไปเพื่อการนี้ ดังนั้น ทางที่ดีถ้าหากลดการตรวจทานสถานะไปได้หรือไม่มีเลยก็จะทำให้โปรแกรมที่ได้มีประสิทธิภาพดีขึ้นด้วย

- การกระจายงานไม่สมดุล หรือไม่เหมาะสม (Load Imbalance) เกิดจากการที่มีวิธีการกระจายการทำงานไม่เหมาะสม ทำให้มีหน่วยประมวลผลบางส่วนเท่านั้นที่มีโอกาสได้ทำงานส่วนหน่วยประมวลผลอื่นอาจจะอยู่ในสถานะรอเพียงอย่างเดียว ดังนั้นเพื่อแก้ปัญหานี้จะต้องมีวิธีการที่เหมาะสมในการกระจายงานไปยังหน่วยประมวลผลในระบบ

6.4 การออกแบบโปรแกรมขนาน

เมื่อจะทำการลงมือเขียนโปรแกรมนั้น จะต้องมีการออกแบบโปรแกรมที่ดีก่อนจึงค่อยลงมือเขียน โดยโปรแกรมที่ดีนั้นจะต้องเป็นไปตามหลักการและหลีกเลี่ยงปัญหาต่าง ๆ ที่ได้นำเสนอไปแล้วในหัวข้อที่ผ่านมา ซึ่งวิธีการออกแบบโปรแกรมขนานนี้ Foster (http://www-unix.mcs.anl.gov/dbpp/. 1995) ได้นำเสนอหลักการต่าง ๆ ไว้สี่ขั้นตอนด้วยกันคือ การแบ่งงาน (Partition) การพิจารณาเรื่องการสื่อสาร (Communication) การรวมกลุ่มงาน (Agglomerate) และการกำหนดงานไปยังหน่วยคำนวณที่เหมาะสม (Mapping) สามารถแสดงขั้นตอนทั้งหมดได้ดังรูปต่อไปนี้



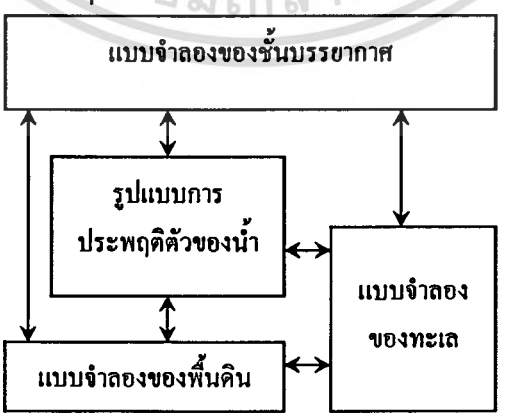
รูปที่ 6.4 ขั้นตอนการออกแบบการทำงานของโปรแกรมขนาน

จากรูปจะเห็นได้ว่าจากปัญหาขนาดใหญ่ที่ต้องการคำนวณนั้น ในขั้นตอนของการแบ่งงานจะทำการแยกแยะปัญหาออกให้เป็นส่วนย่อยตามความเหมาะสมของแต่ละปัญหา หลังจากนั้นจึงทำการพิจารณาความเกี่ยวพันของแต่ละส่วน โดยอาจจะพิจารณาจากขอบเขต การอยู่ติดกันของส่วนย่อยเล็ก ๆ เหล่านั้น รวมทั้งพิจาณาถึงการรับและส่งค่าของปัญหาต่าง ๆ อีกด้วย หลังจากนั้นจึงทำการรวมส่วนที่เกี่ยวข้องหรือมีความสัมพันธ์เข้าไว้ด้วยกันในขั้นตอนของการรวมกลุ่มงาน และสุดท้ายจะเป็นการกำหนดงานไปยังหน่วยคำนวณหรือ Processing Element ที่เหมาะสมต่อไป

6.4.1 ขั้นตอนการแบ่งงาน

การแบ่งงานเป็นจุดเริ่มต้นของการประมวลผลแบบขนาน เพราะจะต้องทำการแบ่งปัญหาขนาดใหญ่ที่ซับซ้อนออกมาเป็นส่วนย่อยที่สามารถคำนวณไปพร้อม ๆ กันให้ได้ก่อน จึงจะสามารถทำตามขั้นตอนอื่น ๆ ต่อไปได้ การแบ่งงานสามารถทำได้สองแบบคือ

- การแบ่งข้อมูลของปัญหาออกเป็นส่วนย่อย (Data Decomposition) ในส่วนนี้จะเป็นการพิจารณาที่ข้อมูลอินพุท ว่าสามารถแบ่งเป็นส่วนย่อย แล้วประมวลผลไปพร้อมกันได้หรือไม่ โดยอาจจะพิจารณาจากความสัมพันธ์ของชุดข้อมูล หรือเป็นกลุ่มของข้อมูลที่อยู่ติดกัน แล้วทำการประมวลผลโดยใช้หลักการ "โปรแกรมเดี่ยวหลายชุดข้อมูล" (Single Program Multiple Data: SPMD) ซึ่งโปรแกรมที่ทำงานอยู่บนแต่ละหน่วยประมวลผลจะเป็นโปรแกรมชุดเดียวกัน แต่ประมวลผลข้อมูลคนละชุดข้อมูล ซึ่งชุดข้อมูลที่ถูกส่งไปคำนวณจะเกิดจากการแบ่งปัญหาด้วยวิธีนี้ ตัวอย่างเช่น การบีบอัดข้อมูลภาพเคลื่อนไหวหรือภาพยนตร์ สามารถทำการแบ่งข้อมูลออกเป็นส่วน ๆ แล้วส่งไปประมวลผลยังแต่ละหน่วยประมวลผล เมื่อได้ผลลัพธ์จึงค่อยนำมารวมกันก็จะได้เป็นภาพยนตร์ที่ผ่านการเข้ารหัสที่สมบูรณ์ทั้งหมด ซึ่งจากตัวอย่างจะเห็นได้ว่า โปรแกรมที่ทำงานอยู่ในแต่ละหน่วยประมวลผลเป็นโปรแกรมเดียวกัน แต่ประมวลผลส่วนของข้อมูลต่างกัน ตามหลักการ SPMD
- การแบ่งปัญหาตามหน้าที่ของการคำนวณออกเป็นส่วนย่อย (Functional Decomposition) ซึ่งการแบ่งงานแบบนี้จะกระทำตั้งแต่เริ่มลงมือเขียนส่วนของโปรแกรม เพราะจะต้องทำการแยกฟังก์ชั่นการทำงานของแต่ละส่วนอย่างชัดเจนตั้งแต่ทีแรก ว่าส่วนไหนทำอะไร และเกี่ยวข้องกับข้อมูลชุดไหนบ้าง เพื่อให้เป็นไปตามหลักการ "หลายโปรแกรมหลายชุดข้อมูล" (Multiple Program Multiple Data: MPMD) เช่น การสร้างแบบจำลองของสภาพภูมิอากาศ ซึ่งประกอบด้วยงานย่อย ๆ คือ การสร้างแบบจำลองของทะเล พื้นดิน และชั้นบรรยากาศ



รูปที่ 6.5 การแบ่งขอบเขตหน้าที่ของปัญหา

6.4.2 ขั้นตอนออกแบบการสื่อสาร

ขั้นตอนการสื่อสารนี้จะพิจารณาความสัมพันธ์ในเรื่องของการสื่อสารระหว่างงานที่แบ่งออกไปในงานบางอย่าง งานย่อยที่ถูกแบ่งไม่สามารถประมวลผลได้ในเวลาเดียวกันได้ทั้งหมด เนื่องจากต้องอาศัยข้อมูลหรือผลลัพธ์จากงานย่อยอื่นเพื่อใช้ในการประมวลผล ดังนั้นจึงต้องมีการสื่อสารระหว่างงานย่อย การเลือกวิธีการสื่อสารที่เหมาะสมจะทำให้การปะมวลผลมีประสิทธิภาพมากยิ่งขึ้น การออกแบบขั้นตอนการสื่อสารจะแบ่งเป็นสองขั้นตอน ขั้นตอนแรกออกแบบโครงสร้างของการเชื่อมโยงว่างานใดต้องการข้อมูลและงานใดทำหน้าที่ให้ข้อมูล ขั้นตอนที่สองคือกำหนดโครงสร้างของข้อความ (Message) ที่ใช้ในการส่งและรับของแต่ละการเชื่อมโยง ลักษณะของการสื่อสารอาจแบ่งได้เป็นสี่แบบดังนี้

- **การสื่อสารกับโปรเซสข้างเคียง (Local Process Communication)**

การสื่อสารกับโปรเซสข้างเคียง คือ การสื่อสารที่เกิดขึ้นระหว่างงานจำนวนน้อย เช่นการวิเคราะห์เชิงเลข (Numerical Analysis) ในวิธีไฟไนต์ดิฟเฟอเรนต์ของ Jacobi ซึ่งการหาค่า $X_{i,j}^{(t+1)}$ ทำได้จากสมการ 6.1 โดยใช้ข้อมูลห้าจุดในการคำนวณ

$$X_{i,j}^{(t+1)} = \frac{4X_{i,j}^{(t)} + X_{i-1,j}^{(t)} + X_{i+1,j}^{(t)} + X_{i,j-1}^{(t)} + X_{i,j+1}^{(t)}}{8} \tag{6.1}$$

งานแต่ละงานทำหน้าที่หาค่าของ $X_{i,j}^{(t+1)}$ ที่ตำแหน่งต่างๆกัน และในขณะเดียวกันก็ทำหน้าที่ส่งค่าของตัวเองให้กับงานอื่นด้วยดังรูปที่ 6.6

รูปที่ 6.6 ลักษณะการสื่อสารในการคำนวณไฟไนต์ดิฟเฟอเรนต์ในสองมิติ

- **การสื่อสารแบบครอบคลุม (Global Communication)**

ส่วนการสื่อสารแบบครอบคลุมจะเป็นการสื่อสารในลักษณะที่งาน ๆ หนึ่งเป็นศูนย์กลางในการติดต่อกับงานอื่น ๆ ตัวอย่างเช่น การคำนวณหาค่าผลรวมดังสมการ 6.2

$$S = \sum_{i=0}^{N-1} X_i \tag{6.2}$$

การคำนวณขนานแบบนี้จะเกิดจากงานที่เป็นผู้จัดการ (Manager) รับค่า X_i ซึ่งเป็นผลลัพธ์จากการคำนวณของผู้ทำงาน (Worker) ต่าง ๆ เพื่อทำการหาค่าผลรวม S โดยการสื่อสารจะมีลักษณะดังรูปที่ 6.7

รูปที่ 6.7 การสื่อสารเพื่อรวมค่าโดยงานที่เป็นผู้จัดการ

- **การสื่อสารแบบไม่มีโครงสร้างและการสื่อสารแบบเปลี่ยนแปลงได้ (Unstructured and Dynamic Communication)**

ในหัวข้อที่ผ่านมาเป็นการสื่อสารแบบคงที่และมีโครงสร้างที่ไม่เปลี่ยนแปลง แต่มีงานบางรูปแบบที่มีโครงสร้างไม่แน่นอนและมีการเปลี่ยนแปลงตามเวลา เช่น การวิเคราะห์ไฟไนต์อีลีเมนต์ซึ่งแสดงดังรูปที่ 6.8

รูปที่ 6.8 ตาราง (Grid) ที่สร้างขึ้นเพื่อวิเคราะห์ไฟไนต์อีลีเมนต์

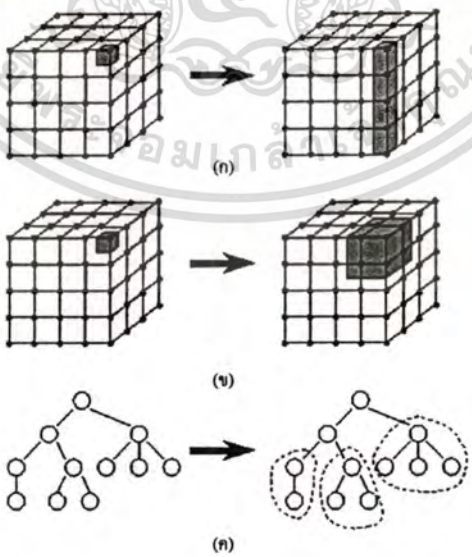
จากรูป ขนาดของตารางที่สร้างขึ้นมาเพื่อการวิเคราะห์ด้วยวิธีไฟไนต์อีลีเมนต์นี้จะไม่มีรูปแบบที่แน่นอน เพราะความละเอียดของตารางไม่เท่ากัน อาจจะขนาดใหญ่ในส่วนที่ไม่สำคัญมาก และจะมีขนาดเล็กลงมากขึ้นในส่วนที่เป็นจุดสนใจต้องการที่จะวิเคราะห์โดยละเอียด ดังนั้นจึงทำให้ไม่สามารถออกแบบการสื่อสารไว้ก่อนได้ วิธีแบ่งงานอาจทำได้โดยกำหนดให้แต่ละจุดต่อ (Vertex) ของเส้นตารางเป็นงานหนึ่งงาน และการสื่อสารจะทำตามจุดต่อตารางที่แบ่งไว้

- **การสื่อสารแบบไม่ประสานกัน (Asynchronous Communication)**

  การสื่อสารทั้งสามรูปแบบที่กล่าวมาล้วนแต่เป็นการสื่อสารแบบประสาน หรือตรวจทานการทำงานทั้งสิ้น (Synchronous Communication) แต่จะมีการสื่อสารอีกรูปแบบหนึ่งที่ไม่มีการตรวจทานจังหวะและเวลาในการส่งข้อมูล ข้อมูลจะต้องถูกร้องขอออกมาก่อนเท่านั้น จึงจะเกิดการสื่อสารขึ้น การสื่อสารแบบประสานกัน ผู้ส่งและผู้รับรู้ว่าจะเกิดการสื่อสารกันก่อนจะเกิดการสื่อสารกันจริง ส่วนการสื่อสารแบบไม่ประสานกันผู้ส่งจะไม่รู้ว่าเมื่อใดผู้รับจะต้องการข้อมูล ผู้รับข้อมูลจะต้องทำการร้องขอข้อมูลโดยตรงจากผู้ส่งข้อมูลเมื่อต้องการ

6.4.3 ขั้นตอนการรวมกลุ่มงาน

ขั้นตอนรวมกลุ่มงานจะถูกทำเพื่อลดเวลาที่ใช้ในการสื่อสารระหว่างงานให้ลดลงและทำให้หน่วยประมวลผล (Processors) ไม่ต้องเสียเวลาในการสื่อสาร และรอคอยการรับส่งข้อมูล ทำให้ประสิทธิภาพในการคำนวณมากขึ้นด้วย



รูปที่ 6.9 การรวมกลุ่มงานในการประมวลผลแบบขนานเพื่อลดเวลาในการสื่อสาร

การรวมกลุ่มงานอาจทำได้หลายลักษณะขึ้นอยู่กับประเภทของงานเช่นจากรูปที่ 6.9

ก. เป็นการรวมกลุ่มงานจากงานในตารางสามมิติเป็นงานในตารางสองมิติ

ข. เป็นการรวมกลุ่มงานจากงานในตารางสามมิติเป็นงานที่ใหญ่ขึ้นในสามมิติ

ค. เป็นการรวมกลุ่มงานในรูปแบบแผนภูมิต้นไม้เป็นต้น

6.4.4 ขั้นตอนการกำหนดงาน

ในขั้นตอนนี้จะเป็นการกำหนดงานไปยังหน่วยประมวลผลหรือหน่วยคำนวณที่เหมาะสม เพื่อเป็นการใช้ประโยชน์ของหน่วยประมวลผลสูงสุด และลดการสื่อสารให้มีน้อยที่สุด การกำหนดงานนี้สามารถทำเป็นแบบคงที่ที่กำหนดไว้ตายตัวตั้งแต่เริ่มสร้างโปรแกรม หรือทำการกำหนดงานตอนที่กำลังทำการคำนวณโดยใช้เทคนิคของการสมดุลงาน (Load Balancing) เข้ามาช่วยก็ได้

การกำหนดงานจะถูกทำเป็นขั้นตอนสุดท้ายในการออกแบบอัลกอริทึมแบบขนาน การกำหนดงานบนคอมพิวเตอร์แบบขนานจะต่างจากระบบคอมพิวเตอร์ที่มีหน่วยประมวลผลเดียว โดยในคอมพิวเตอร์ที่มีหน่วยประมวลผลเดียว ระบบปฏิบัติการจะเป็นผู้กำหนดและจัดลำดับงาน (Scheduling) ให้กับหน่วยประมวลผลเพื่อทำการประมวลผล ส่วนในคอมพิวเตอร์แบบขนานการกำหนดงานอาจทำโดยอัตโนมัติจากระบบปฏิบัติการแบบขนานเช่นในระบบคลัสเตอร์ที่ใช้คลัสเตอร์มิดเดิลแวร์อย่าง OpenMosix เป็นต้น หรืออาจกำหนดจากผู้สร้างโปรแกรมโดยตรง การกำหนดงานเป็นขั้นตอนที่ซับซ้อนซึ่งวิธีการกำหนดงานที่ดีจะทำให้ใช้เวลาในการประมวลผลงานน้อยและมีความยืดหยุ่นในการใช้งาน หลักการในการกำหนดงานที่ดีมีสองข้อคือ

- กำหนดงานที่สามารถประมวลผลพร้อมๆกันได้ให้ทำงานในหน่วยประมวลผลต่างกัน
- กำหนดงานที่ต้องมีการสื่อสารกันบ่อยให้ทำงานอยู่ในหน่วยประมวลผลเดียวกัน

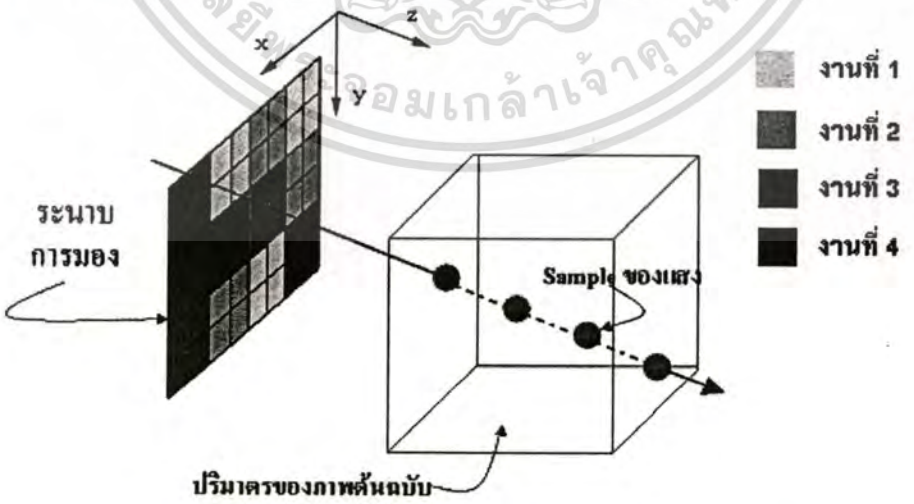
การกำหนดงานไม่มีรูปแบบที่แน่นอนขึ้นอยู่กับประเภทของงาน รูปแบบของแต่ละปัญหาและลักษณะคอมพิวเตอร์แบบขนาน เช่น ในระบบคอมพิวเตอร์คลัสเตอร์แบบเนื้อผสม (Heterogeneous) ซึ่งความสามารถในการประมวลผลของแต่ละหน่วยประมวลผลอาจไม่เท่ากันจะทำให้วิธีการกำหนดงานมีความยากและซับซ้อนขึ้นไปอีก

6.5 การสร้างภาพเชิงปริมาตรแบบฉายแสงบนระบบคลัสเตอร์

การสร้างภาพเชิงปริมาตรแบบฉายแสงบนระบบคลัสเตอร์ ได้ถูกนำเสนอไว้โดยนนท์ บัณฑิตวงษ์ (2545) เพื่อเป็นการเพิ่มความเร็วในการสร้างภาพเชิงปริมาตรในแต่ละมุมมองให้เร็วขึ้น ในวิธีการที่นำเสนอใช้หลักการที่ว่ากระบวนการสร้างภาพเชิงปริมาตรสามารถแบ่งการทำงานออกได้เป็นกลุ่มๆ ซึ่งไม่ขึ้นต่อกัน และไม่มีความจำเป็นต้องมีการสื่อสารระหว่างงานย่อย (Embarrassingly Parallel Computations) ซึ่งนับเป็นข้อได้เปรียบของงานประมวลผลภาพ และงานด้านคอมพิวเตอร์กราฟิคส่วนใหญ่ งานประเภทอื่นอาจต้องใช้วิธีการแบ่งงานที่ต่างออกไป

6.5.1 การแบ่งงานในการสร้างภาพเชิงปริมาตร

การแบ่งงานในการสร้างภาพเชิงปริมาตรใช้วิธีแบ่งข้อมูลของปัญหาออกเป็นส่วนย่อย (Data Decomposition) โดยหน่วยของข้อมูลที่เล็กที่สุดคือจุดภาพ (Voxel) ตามแนวลำแสงดังแสดงในรูปที่ 6.10 การแบ่งข้อมูลของปัญหาออกเป็นส่วนย่อยมีข้อดีคือสามารถทำงานได้ไม่ว่าจะมีจำนวนโพรเซสอยู่ในระบบคลัสเตอร์เท่าใดก็ตาม (Scalable) แต่ต้องไม่มากกว่าจำนวนกลุ่มข้อมูลที่แบ่งไว้โดยสามารถแบ่งได้ละเอียดที่สุดถึงหนึ่งแนวลำแสงต่อหนึ่งงาน ปัญหาที่พบคือถ้าหากกำหนดให้ข้อมูลของปริมาตรภาพไปที่แต่ละโพรเซสในระบบคลัสเตอร์ เมื่อต้องการหมุนภาพจะต้องทำการกระจายชุดข้อมูลที่แบ่งนั้นไปให้แต่ละโพรเซสในระบบใหม่ ซึ่งข้อมูลภาพอาจมีขนาดใหญ่มากทำให้สิ้นเปลืองเวลาในการส่งข้อมูล วิธีการแก้ไขคือให้แต่ละโพรเซสทำการอ่านข้อมูลภาพทั้งปริมาตรไปเก็บไว้ในหน่วยความจำของตัวเอง เมื่อมีการหมุนภาพก็เพียงทำการหมุนตามขั้นตอนของการสร้างภาพเชิงปริมาตรดังที่กล่าวมาแล้วในบทที่ 3 แต่ข้อเสียของวิธีนี้คือทำให้สิ้นเปลืองหน่วยความจำ



รูปที่ 6.10 การแบ่งข้อมูลของปัญหาออกเป็นส่วนย่อยตามแนวลำแสง

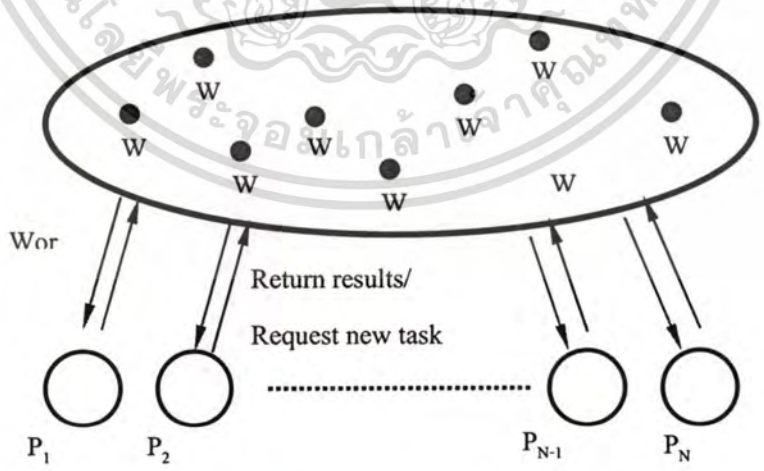
6.5.2 การรวมกลุ่มงาน

การรวมกลุ่มงานของการสร้างภาพเชิงปริมาตรนี้สามารถทำได้ง่ายโดยให้แนวลำแสงที่อยู่ติดกันรวมเป็นกลุ่มงานเดียวกัน การกำหนดให้งานหนึ่งงานเป็นการสร้างภาพในแนวหนึ่งลำแสงอาจเป็นงานที่ต้องอาศัยการคำนวณน้อยเกินไปและทำให้มีการสื่อสารมากเกินความจำเป็นซึ่งอาจทำให้เวลาที่ใช้ในการสร้างภาพส่วนใหญ่เสียไปกับการสื่อสารในขณะที่แต่ละโพรเซสใช้เวลาในการคำนวณของแต่ละงานน้อยมาก ตัวอย่างขนาดของกลุ่มงานที่สามารถใช้ได้ เช่น กลุ่มงานขนาด 32x32 หรือ 64x64 แนวลำแสง เป็นต้น

6.5.3 การสื่อสารระหว่างงานในการสร้างภาพเชิงปริมาตร

เนื่องจากงานที่ถูกแบ่งในการสร้างภาพเชิงปริมาตรไม่จำเป็นต้องมีการสื่อสารระหว่างกลุ่มงานด้วยกัน การออกแบบการสื่อสารจึงไม่มีความยุ่งยาก ซึ่งในวิทยานิพนธ์ฉบับนี้ใช้หลักการของศูนย์รวมงาน (Work Pool) ซึ่งเป็นการสื่อสารแบบสองทิศทางระหว่างโปรเซสที่เป็นผู้จัดการและโปรเซสที่เป็นผู้ทำงาน โดยมีลักษณะดังรูปที่ 6.11 โครงสร้างของข้อมูลที่ใช้ส่งเป็นอินพุทให้แก่โปรเซสผู้ทำงานประกอบด้วยตำแหน่งของจุดเริ่มของบล็อกตามแนวแกน X และ Y ส่วนโครงสร้างของข้อมูลที่เป็นผลลัพธ์กลับมาจากโปรเซสผู้ทำงานจะประกอบด้วยตำแหน่งของจุดเริ่มของบล็อกที่ประมวลผลเสร็จและค่าสีของข้อมูลภาพภายในบล็อก

6.5.4 การกำหนดงานในการสร้างภาพเชิงปริมาตร



รูปที่ 6.11 การกำหนดงานแบบศูนย์รวมงาน (Work Pool)

การกำหนดงานแบบศูนย์รวมงานอาศัยหลักการที่ว่าให้โปรเซสที่สร้างขึ้นในระบบคลัสเตอร์ไปรับงานที่ต้องประมวลผลจากศูนย์รวมงาน แทนที่จะให้ผู้สร้างโปรแกรมเป็นผู้กำหนดงานที่ต้องทำให้แก่โปรเซสโดยตรงซึ่งทำให้ขาดความยืดหยุ่นในกรณีที่ต้องการให้มีจำนวนโปรเซสในระบบมากขึ้นหรือต้องการเปลี่ยนแปลงขนาดของกลุ่มงาน

6.6 สรุป

ในบทนี้ได้กล่าวถึงภาพกว้าง ๆ ของการโปรแกรมแบบขนาน โดยเริ่มจากรูปแบบหรือวิธีการในการพัฒนาโปรแกรม ว่าสามารถที่จะสร้างโปรแกรมได้จากวิธีไหนบ้าง แล้วตามด้วย เทคนิค,วิธีการที่ได้ถูกนำมาใช้ และปัญหาต่าง ๆ ที่ควรหลีกเลี่ยงในการสร้างโปรแกรม หลักจากนั้นเป็นการกล่าวถึงวิธีการในการออกแบบโปรแกรมแบบขนาน โดยได้นำเสนอขั้นตอนต่าง ๆ สี่ขั้นตอนดังนี้คือ ขั้นตอนการแบ่งงาน ขั้นตอนการสื่อสาร ขั้นตอนการรวมกลุ่มงาน และขั้นตอนการกำหนดงาน และในหัวข้อสุดท้ายของบทนี้ได้กล่าวถึงการสร้างภาพเชิงปริมาตรแบบฉายแสงบนระบบคอมพิวเตอร์คลัสเตอร์ ซึ่งได้อธิบายให้เห็นรายละเอียดต่าง ๆ โดยคร่าวแล้ว

บทที่ 7

การทดลองและผลการทดลอง

7.1 บทนำ

การสร้างภาพสามมิติสามารถแบ่งเป็นสองวิธีใหญ่ ๆ คือ การสร้างภาพเชิงพื้นผิว และการสร้างภาพเชิงปริมาตร การสร้างภาพเชิงพื้นผิวมีข้อดีคือสามารถสร้างภาพได้รวดเร็ว แต่ภาพที่ได้จะไม่สามารถมองเห็นรายละเอียดด้านในวัตถุได้เลย ภาพเชิงปริมาตรจึงถูกสร้างขึ้นมาเพื่อแก้ไขข้อจำกัดของภาพเชิงพื้นผิว เพราะนอกจากจะแสดงรายละเอียดที่พื้นผิวได้แล้ว ยังสามารถที่จะแสดงรายละเอียดภายในวัตถุ ที่อาจจะมีขอบเขตไม่แน่นอนอย่าง เช่น โครงสร้างของเนื้อเยื่อ โครงสร้างของอวัยวะภายในร่างกาย เป็นต้น ทำให้เป็นการขยายขอบเขตการรับรู้และมองเห็นของมนุษย์ให้ดียิ่งขึ้นไปอีก แต่การสร้างภาพเชิงปริมาตรมีข้อเสียคือใช้เวลาในการสร้างภาพแต่ละมุมมองมาก ทำให้มีการพยายามนำการสร้างภาพเชิงปริมาตรนี้มาประมวลผลบนคอมพิวเตอร์แบบขนาน เพื่อทำให้ได้ภาพผลลัพธ์ที่รวดเร็วขึ้น การเพิ่มความเร็วในการสร้างภาพเชิงปริมาตรโดยใช้คอมพิวเตอร์แบบขนานนี้สามารถจำแนกได้สามวิธี (Giertsen and Petersen. 1993) คือ

1. การประมวลผลบนฮาร์ดแวร์ (Hardware) ที่ถูกออกแบบมาเพื่อรองรับการประมวลผลแบบขนานโดยเฉพาะ (Pfister et. al. 1995)
2. ปรับปรุงอัลกอริธึม (Algorithm) ของการแบ่งงานในการสร้างภาพแบบขนาน และอัลกอริทึมที่ใช้ในการสร้างภาพเชิงปริมาตร (Lacroute and Levoy, 1994)
3. การนำเอาโปรแกรมที่ใช้สร้างภาพไปประมวลผลบนระบบคอมพิวเตอร์ที่รองรับการประมวลผลแบบขนาน (Schroder, 1991) เช่น ระบบคลัสเตอร์ (Cluster) หรือคอมพิวเตอร์แบบกริด (Grid Computing)

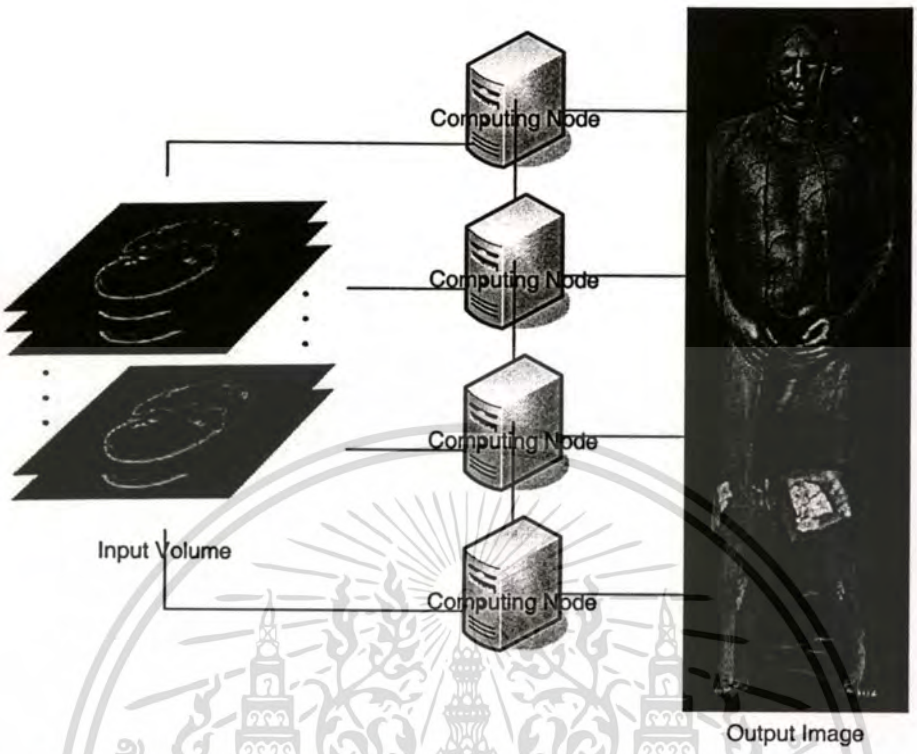
วิธีแรกอาจทำได้ง่ายในปัจจุบันเนื่องจากอุปกรณ์คอมพิวเตอร์มีราคาถูกลงมาก แต่ไม่สามารถรองรับงานที่อาจมีมากขึ้นในอนาคตได้ วิธีที่สองเป็นวิธีที่ดีที่สุดแต่อาจทำได้ยาก และวิธีสุดท้ายเป็นวิธีที่กำลังได้รับความสนใจเป็นอย่างมากในปัจจุบัน เนื่องจากสามารถใช้ร่วมกับวิธีที่หนึ่งและสอง เพื่อเพิ่มความเร็วในการประมวลผลให้มากขึ้นไปอีกได้

ซึ่งวิทยานิพนธ์ฉบับนี้ได้ใช้ข้อดีของการประมวลผลบนระบบคลัสเตอร์ ร่วมกับวิธีการในการสร้างภาพเชิงปริมาตรที่สามารถทำงานได้อย่างรวดเร็ว คือ การสร้างภาพเชิงปริมาตรโดยวิธีการแปลงเฉือนและบิด อีกทั้งใช้เทคนิคการแบ่งงานในการประมวลผลแบบขนานเข้าช่วย เพื่อให้การประมวลผลมีประสิทธิภาพ สามารถสร้างภาพเชิงปริมาตรในแต่ละมุมมองได้อย่างรวดเร็วใช้เวลาในการประมวลผลน้อยที่สุด

7.2 การสร้างภาพเชิงปริมาตรโดยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์

วิธีการสร้างภาพเชิงปริมาตรเป็นวิธีการที่มีความมุ่งหมายที่จะแสดงรายละเอียดของชุดข้อมูลรูปภาพให้มีความเข้าใจได้ง่ายยิ่งขึ้น เช่น การสร้างภาพเชิงปริมาตรทางการแพทย์ จะเป็นการสร้างภาพผลลัพธ์ให้สามารถเห็นโครงสร้างและรายละเอียดที่มุมมองต่างกันของอวัยวะต่าง ๆ ภายในร่างกายได้โดยอาศัยชุดข้อมูลภาพเริ่มต้นเพียงชุดเดียว ซึ่งภาพผลลัพธ์ที่ได้จะมีความลึก แสงเงา ความสว่าง ความโปร่งแสง รายละเอียดเชิงพื้นผิว อันที่จะทำให้เกิดความเข้าใจเนื้อหาของภาพได้มากขึ้น แต่ปัญหาที่ประสบในปัจจุบันคือ การสร้างภาพแต่ละมุมมองต้องใช้เวลานาน เพราะฉะนั้นถ้าหากมีวิธีการที่ทำให้การสร้างภาพทำได้รวดเร็ว ก็จะทำให้วิธีการสร้างภาพเชิงปริมาตรนี้เหมาะกับการใช้งานจริงมากยิ่งขึ้น

ในวิทยานิพนธ์ฉบับนี้จึงได้นำเอาการสร้างภาพเชิงปริมาตรด้วยวิธีการแปลงเฉือนและบิด ซึ่งเป็นวิธีที่สามารถสร้างภาพได้อย่างรวดเร็วมากมาทำงานร่วมกับระบบการประมวลผลแบบหลายหน่วยประมวลที่เรียกว่าระบบคลัสเตอร์ โดยการสร้างโปรแกรมการสร้างภาพเชิงปริมาตรขึ้นมาให้มีความเหมาะสมที่จะประมวลผลบนคอมพิวเตอร์ระบบนี้ อีกทั้งมีวิธีการในการแบ่งงานที่ดีเพื่อให้ประสิทธิภาพโดยรวมของระบบดีมากขึ้น



รูปที่ 7.1 การสร้างภาพเชิงปริมาตรบนระบบคลัสเตอริ่งโดยวิธีการแปลงเฉือนและบิด

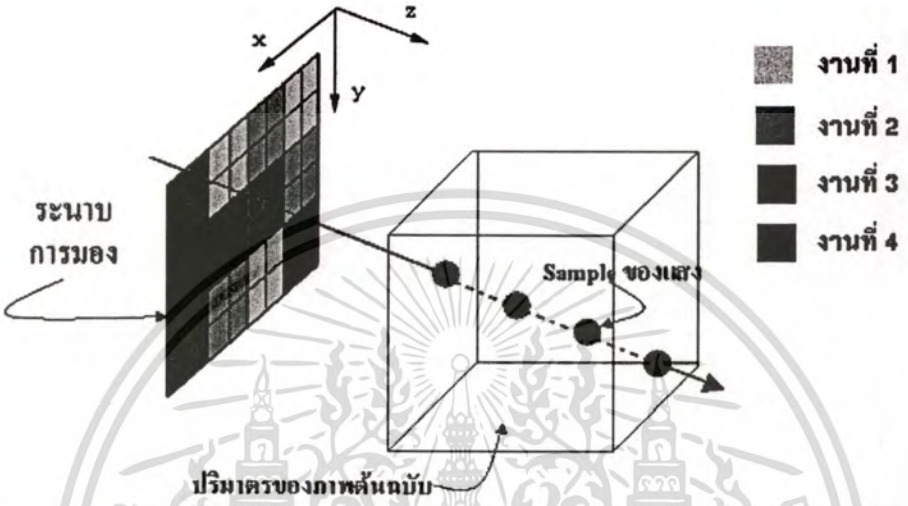
7.2.1 การออกแบบโปรแกรมการสร้างภาพแบบขนาน

การสร้างภาพเชิงปริมาตรแบบเฉือนและบิดบนระบบคลัสเตอร์นี้ได้ใช้วิธีการออกแบบโครงสร้างของโปรแกรมโดยยึดหลักการตามวิธีของ Foster (1995) อีกทั้งใช้เทคนิควิธีการ และคำนึงถึงผลกระทบที่จะเกิดขึ้นในการประมวลผลแบบขนานที่ได้ถูกนำเสนอไว้โดย Lester (1994) ดังมีรายละเอียดดังนี้

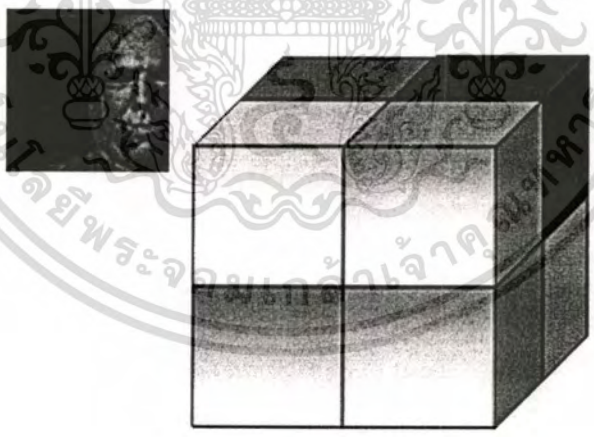
- **ขั้นตอนการแบ่งงาน**

นนท์ บัณฑิตวงษ์ (2545) ได้นำเสนอวิธีการแบ่งปัญหาในการสร้างภาพเชิงปริมาตรโดยใช้วิธีการฉายแสงว่า เราสามารถที่จะทำการแบ่งการทำงานออกเป็นส่วนย่อย ๆ ได้ถึงระดับของหนึ่งงานต่อการฉายแสงหนึ่งลำแสง แต่ในงานวิจัยนี้ใช้วิธีการสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิด ซึ่งเป็นการสร้างภาพเชิงปริมาตรแบบลำดับวัตถุที่พิจารณาที่ชุดของข้อมูลโดยตรง จึงไม่สามารถที่จะแบ่งงานออกเป็นส่วนย่อยตามลักษณะการฉายแสงแบบนั้นได้ ดังนั้นในวิทยานิพนธ์ฉบับนี้จึงใช้การแบ่งข้อมูลออกเป็นกลุ่ม ๆ แล้วค่อยทำการส่งไปยังแต่ละหน่วยประมวลผลเพื่อสร้างเป็นภาพผลลัพธ์ แล้วค่อยทำการรวมผลลัพธ์ที่ได้โดยโปรเซสที่ทำงานเป็น Manager เพราะฉะนั้นงานย่อยที่สุดของการสร้างภาพเชิงปริมาตรด้วย

การแปลงเฉือนและบิดนี้จึงเป็นการสร้างภาพเชิงปริมาตรต่อเส้นสแกนไลน์ (Scan Line) หนึ่งเส้นนั่นเอง แต่เพื่อไม่ให้เกิดการสื่อสารในระบบมากเกินไป จึงทำการแบ่งงานเป็นก้อนปริมาตรแบบลูกบาตรที่มีด้านกว้าง สูง และยาวเท่ากัน



รูปที่ 7.2 การแบ่งงานเป็นส่วนย่อยโดยพิจารณาที่ภาพผลลัพธ์ในการสร้างภาพแบบฉายแสง



รูปที่ 7.3 การแบ่งงานเป็นส่วนย่อยโดยแบ่งที่ปริมาตรโดยตรงในการสร้างภาพแบบแปลงเฉือนและบิด

- **ขั้นตอนสื่อสาร**

ในขั้นตอนของการสื่อสารนี้จะเป็นการสื่อสารแบบครอบคลุม (Global Communication) หรือ One-to-Many เพราะในขณะที่ทำการสร้างภาพเชิงปริมาตรนั้นจะมีหนึ่งโปรเซสที่ทำหน้าที่เป็น Manager คอยกำหนดงาน และรับค่าผลลัพธ์ที่ได้จากโปรเซสอื่น ๆ ที่ทำหน้าที่เป็น Worker ในระบบ ตามวิธีการกระจายงานแบบ Work Pool

การสื่อสารจะเป็นแบบสองทาง เพราะมีทั้งการส่งและการรับค่าระหว่าง Manager และ Work แต่จะไม่เกิดการสื่อสารระหว่าง Worker ด้วยกันเอง (Local Communication) เพราะแต่ละโปรเซสไม่ต้องการข้อมูลอื่น ๆ เพิ่มเติมจากโปรเซสข้างเคียง เมื่อทำการสร้างภาพผลลัพธ์เสร็จแล้วก็จะส่งค่ากลับไปให้แก่ Manager เพื่อสร้างเป็นภาพผลลัพธ์ที่สมบูรณ์ต่อไป รูปแบบของข้อมูลที่ใช้ในการสื่อสารกันมีรายละเอียดดังนี้

(ก)
```
typedef struct {
   int X,Y,Z;  /* x, y, z origin */
   } Manager_Block;
```

(ข)
```
typedef struct {
   int X,Y,Z;  /* x, y, z origin */
   unsigned char image[IMAGE_DATA_SIZE]; /* image data */
   } Worker_Block;
```

โดยที่

(ก) โครงสร้างข้อมูลที่เป็นอินพุทให้แก่โปรเซสผู้ทำงาน (Worker)

(ข) โครงสร้างข้อมูลเป็นผลลัพธ์จากโปรเซสที่เป็นผู้ทำงาน

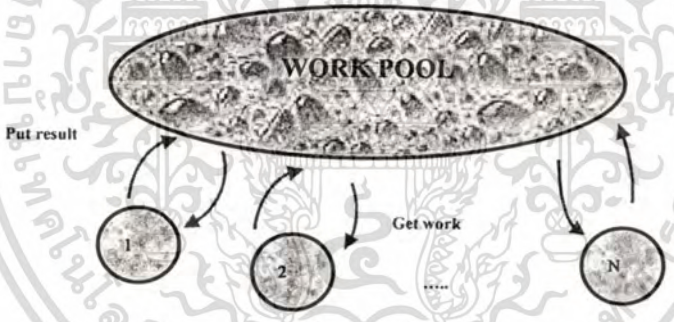
- **ขั้นตอนการรวมกลุ่มงาน**

เนื่องจากว่าในการสร้างภาพเชิงปริมาตรนี้ไม่มีการสื่อสารระหว่าง Worker ด้วยกันเอง (Local Communication) ดังนั้นในขั้นตอนของการรวมกลุ่มงานนี้จึงไม่มีผลกระทบต่อประสิทธิภาพของโปรแกรม

- **ขั้นตอนการกำหนดงาน**

เนื่องจากระบบคอมพิวเตอร์แบบคลัสเตอร์ประกอบขึ้นมาจากคอมพิวเตอร์หลายเครื่อง หรืออาจกล่าวได้ว่าประกอบด้วยหลายโหนด (Node) การทำงาน ซึ่งคอมพิวเตอร์เหล่านี้ อาจแตกต่างกันในหลาย ๆ ส่วน เช่น ความเร็วของหน่วยประมวลผล จำนวนของหน่วยประมวลผล ปริมาณของหน่วยความจำ ทำให้ความสามารถในการประมวลผลแตกต่างกัน ถ้าหากกระจายงานให้แต่ละเครื่องเท่า ๆ กันจะทำให้เครื่องที่ประมวลได้เร็วทำงานเสร็จก่อนและต้องรอเครื่องอื่น ๆ ที่ช้ากว่า ซึ่งจะเห็นได้ว่าการแบ่งงานแบบนี้ไม่ใช่วิธีที่ดีที่สุด ถ้าหากมีวิธีการแบ่งงานให้เครื่องที่ประมวลผลได้เร็วรับงานไปมากกว่าเครื่องที่ช้าก็จะทำให้ระบบนี้มีประสิทธิภาพที่มากขึ้น

สำหรับวิธีการกำหนดงานให้แก่เครื่องลูกในระบบคลัสเตอร์เพื่อให้แต่ละโปรเซสได้รับปริมาณงานที่เหมาะสมไปประมวลผลนั้นได้ใช้วิธีการกระจายงานแบบรวมศูนย์ (Work Pool) โดยที่แต่ละโหนดสามารถประมวลผลให้เสร็จในเวลาที่ใกล้เคียงกันถึงแม้ว่าปริมาณงานที่ได้รับในเครื่องคอมพิวเตอร์แต่ละเครื่องบนระบบคลัสเตอร์จะไม่เท่ากันก็ตาม



รูปที่ 7.4 วิธีการกำหนดงานแบบ Work Pool

ในขั้นตอนการกำหนดงาน โปรเซส Manager จะทำการตรวจสอบสถานะของโปรเซส Worker ทุกตัว ว่ากำลังทำงานอยู่ หรือว่ามีสถานะว่าง ถ้าหากโปรเซสใดมีสถานะว่างและเตรียมพร้อมสำหรับการรับค่างานใหม่ โปรเซส Manger จะกำหนดงานให้ใหม่ทันที จะทำอย่างนี้ไปเรื่อย ๆ จนกว่างานใน Pool จะหมดลง และได้ค่าผลลัพธ์ในขั้นตอนสุดท้าย

7.2.2 ระบบคลัสเตอร์ที่ใช้ในการทดลอง

ระบบคลัสเตอร์เป็นระบบที่มีโครงสร้างแบบหลายหน่วยประมวลผล และมีหน่วยความจำกระจาย (Distributed Memory) เป็นระบบที่เหมาะกับการสร้างภาพเชิงปริมาตรเป็นอย่างมาก เพราะสำหรับปริมาตรข้อมูลขนาดใหญ่ ถ้าหากเป็นการประมวลผลบนคอมพิวเตอร์เครื่องเดียว แทบจะเป็นไปไม่ได้เลย เพราะคอมพิวเตอร์เครื่องนั้นจะต้องมีสมรรถนะที่สูงมาก ไม่ว่าจะเป็นหน่วยการประมวลผลความเร็วสูง หรือหน่วยความจำขนาดใหญ่ เนื่องจากชุดข้อมูลที่นำมาใช้ในการสร้างภาพเชิงปริมาตรนั้นมักมีขนาดใหญ่มาก เช่น ข้อมูลภาพถ่ายเชิงปริมาตรของทั้งตัวของมนุษย์ซึ่งที่ใช้ในการทดลองของวิทยานิพนธ์ฉบับนี้ เป็นต้น โดยข้อมูลภาพต้นฉบับจะมีขนาด 587x341x1877 Voxel หรือคิดเป็นขนาดของข้อมูลประมาณ 358.3 Megabyte และเมื่อทำการคำนวณสร้างภาพผลลัพธ์นั้น ต้องการพื้นที่ในหน่วยความจำอีกอย่างน้อย 4 byte/Voxel ดังนั้นในขณะที่ทำการสร้างภาพผลลัพธ์นั้น จะต้องการพื้นที่ในหน่วยความจำทั้งหมดประมาณ 1.433 Gigabyte เป็นอย่างน้อย ซึ่งจะเห็นได้ว่า ด้วยความต้องการหน่วยความจำที่มากขนาดนี้ จึงแทบจะเป็นไปไม่ได้เลย สำหรับการสร้างภาพเชิงปริมาตรแบบทั้งตัวของมนุษย์บนเครื่องคอมพิวเตอร์ส่วนบุคคลทั่ว ๆ ไป

- **โครงสร้างทางฮาร์ดแวร์**

  โครงสร้างทางด้านฮาร์ดแวร์ของระบบที่ใช้ในการคำนวณนั้น เป็นคลัสเตอร์แบบโครงสร้างเหมือนกัน หรือ Homogenous Cluster คือจะมีหน่วยประมวลผล ปริมาณหน่วยความจำ และส่วนประกอบอื่น ๆ เหมือนกัน ระบบที่สร้างขึ้นมานี้ประกอบด้วยคอมพิวเตอร์ PC Server จำนวน 4 เครื่อง โดยใช้เครื่องคอมพิวเตอร์ชนิดที่มีสองหน่วยประมวลผลในหนึ่งเครื่อง (Dual CPU) หน่วยประมวลผล Intel Xeon 2.4 GHz หน่วยความจำชนิด ECC จำนวน 1 GB เชื่อมต่อกันผ่านทางเครือข่ายภายในความเร็ว 100 Mbps

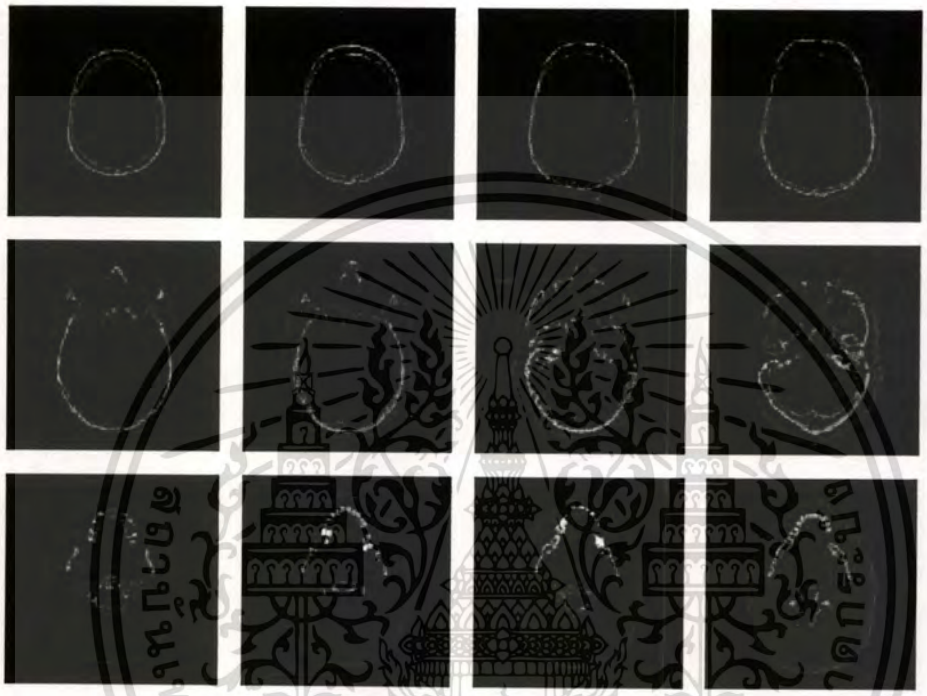
- **โครงสร้างทางซอร์ฟแวร์**

  ระบบปฏิบัติการของคลัสเตอร์ใช้ระบบปฏิบัติการ Linux ของบริษัท RedHat รุ่น Fedora และใช้ gcc เป็นตัวแปรภาษาเนื่องจากมีมาพร้อมกับ Linux อยู่แล้ว และเป็นที่นิยมใช้กันโดยทั่วไป

  ในส่วนของซอร์ฟแวร์ที่ทำหน้าที่เป็นตัวสร้างสภาวะแวดล้อมแบบขนานนั้นในวิทยานิพนธ์ฉบับนี้เลือกใช้ MPICH ซึ่งเป็นชุดคำสั่งที่ถูกพัฒนาขึ้นมาตาม มาตรฐาน MPI และมีชุดคำสั่งต่าง ๆ ให้ใช้งานอย่างเพียงพอ อีกทั้งสามารถใช้งานได้กับระบบปฏิบัติการที่หลากหลาย เช่น Linux, Sun Solaris, HP-UX หรืออื่น ๆ

7.2.3 ข้อมูลเชิงปริมาตรที่ใช้ในการทดลอง

ข้อมูลที่ใช้ในการทดลองนั้น เป็นข้อมูลจากเครื่องถ่าย CT ของโครงการ Visible Human Project (http://www.nlm.nih.gov/) มีขนาด 587x341x1877 จุดภาพ ชนิดของข้อมูลภาพเป็นภาพระดับเทา 256 ระดับ ดังรูป



รูปที่ 7.5 ภาพที่ใช้ในการทดลอง

7.3 การวัดประสิทธิภาพของการสร้างภาพเชิงปริมาตรบนระบบคลัสเตอร์

การวัดเวลาที่ใช้ในการสร้างภาพเชิงปริมาตรในวิทยานิพนธ์ฉบับนี้จะทำการวัดเวลาที่ใช้ในการสร้างภาพจริงๆเท่านั้นซึ่งจะไม่รวมไปถึงเวลาที่ใช้ในการอ่านภาพมาเก็บไว้ในหน่วยความจำและการตั้งค่าระบบ (Initialization) ของ MPI ซึ่งกระบวนการทั้งสองจะถูกทำครั้งเดียวในการสร้างภาพ เมื่อต้องทำการสร้างภาพในมุมมองใหม่จะไม่ต้องทำกระบวนการทั้งสองอีก แนวโน้มของเวลาที่ใช้ในการสร้างภาพด้วยวิธีการที่นำเสนอในวิทยานิพนธ์นี้เป็นดังสมการ 7.1

$$T = \frac{T_s}{p} \tag{7.1}$$

เมื่อ T คือ เวลาที่ใช้ในการสร้างภาพ

T_s คือ เวลาที่ใช้ในการสร้างภาพเชิงปริมาตร โดยใช้ Sequential algorithm

p คือ จำนวนโปรเซส (Process) ที่ใช้ในระบบคลัสเตอร์

จากสมการที่ 7.1 ถึงแม้ว่าจำนวนของหน่วยประมวลผล p ในระบบจะมีเพิ่มขึ้นมาก ๆ แต่เวลารวมที่ได้ของระบบจะไม่เป็นศูนย์ เพราะสาเหตุมาจากเวลาที่ใช้ในการประมวลผลจะมีลักษณะตามสมการ 7.2

$$T_p = t_{comp} + t_{comm} \tag{7.2}$$

เมื่อ T_p คือ เวลารวมที่ใช้ในการสร้างภาพ

t_{comp} คือ เวลาที่ใช้ในการประมวลผล

t_{comm} คือ เวลาที่ใช้ในการสื่อสารระหว่างโปรเซส

ดังนั้นจะทำให้สรุปได้ว่าเมื่อเราเพิ่มหน่วยประมวลผลเข้าไปในระบบมากขึ้น เวลาจะลดลงในช่วงหนึ่งเท่านั้น แต่เมื่อเพิ่มหน่วยประมวลผลเข้าไปอีก เวลาจะไม่ลดลง แต่อาจจะเพิ่มมากขึ้นด้วย เนื่องจากเวลาที่ใช้ในการสื่อสารระหว่างหน่วยประมวลผล (t_{comm}) มากขึ้น

ระบบคลัสเตอร์เป็นระบบที่การสื่อสารระหว่างโพรเซสจะทำผ่านระบบเครือข่ายซึ่งความเร็วของการสื่อสารจะขึ้นอยู่กับชนิดและคุณภาพของอุปกรณ์ระบบเครือข่าย อุปกรณ์เครือข่ายที่ดีควรจะทำให้ t_{comm} มีค่าน้อยๆ เพราะเมื่อเกิดการสื่อสารระหว่างหลาย ๆ โปรเซสในระบบคลัสเตอร์ เวลาที่สูญเสียไปเพราะการสื่อสารจะมีค่ามากขึ้นตามจำนวนของการสื่อสารเหล่านั้นด้วย ซึ่งจะส่งผลให้เวลาที่ใช้ในการสร้างภาพมากขึ้น

นอกจากการวัดเวลาที่ใช้ในการประมวลผลแล้วยังสามารถวัดอัตราการเพิ่มของความเร็ว (Speed up) ของการสร้างภาพได้จากสมการ 7.3 (Lakshmivarahan and Dhall, 1990:18)

$$S_p = \frac{T_s}{T_p} \tag{7.3}$$

เมื่อ S_p คืออัตราการเพิ่มของความเร็วที่ใช้ในการสร้างภาพ

T_s คือ เวลาที่ใช้ในการสร้างภาพเชิงปริมาตรโดยใช้ Sequential algorithm

T_p เวลาที่ใช้ในการสร้างภาพโดยใช้ p หน่วยประมวลผล บนระบบคลัสเตอร์

การวัดประสิทธิภาพของระบบทำได้โดยใช้สมการที่ 7.4 ดังนี้

$$E_p = \frac{S_p}{p} \leq 1 \tag{7.4}$$

เมื่อ E_p คือ ประสิทธิภาพของระบบ มีค่าสูงสุดคือ 1

S_p คือ ค่าอัตราการเพิ่มของความเร็วที่คำนวณได้จากสมการที่ 7.3

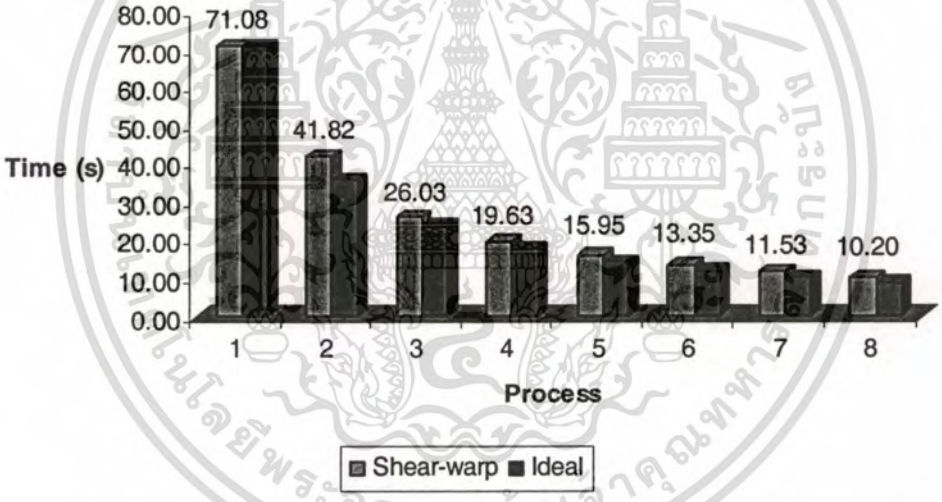
p คือ จำนวนหน่วยโปรเซส (Process) ที่ใช้ในระบบคลัสเตอร์

7.4 การทดลองและผลการทดลอง

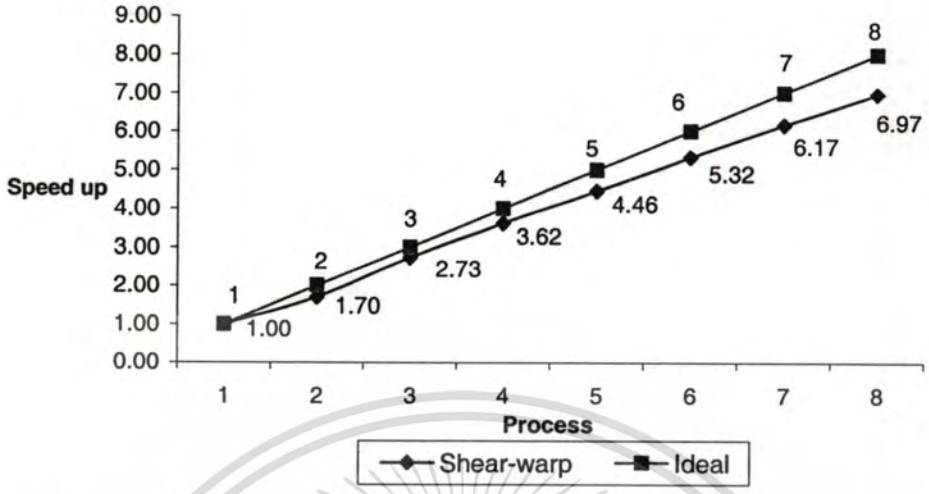
ในวิทยานิพนธ์นี้ได้ทำการทดลองโดยมุ่งความสนใจไปที่การสร้างภาพเชิงปริมาตรที่มีขนาดใหญ่มาก โดยปริมาตรของข้อมูลเริ่มต้นคือ 587x341x1877 Voxels เป็นข้อมูลแบบเทา 8 Bit และผลลัพธ์ที่ได้เป็นภาพสี 24 Bit และมีการแบ่งการทดลองออกเป็นตอนต่าง ๆ ทั้งหมด 4 การทดลองคือ การทดลองเปรียบเทียบระหว่างการเก็บข้อมูลแบบศูนย์กลางและแบบกระจาย การทดลองเปรียบเทียบระหว่างวิธีการฉายแสงกับการแปลงเฉือนและบิด การทดลองเปรียบเทียบเมื่อขนาดก้อนปริมาตรต่างกัน การทดลองเปรียบเทียบเมื่อค่า Threshold ต่างกัน ดังมีรายละเอียดดังนี้

7.4.1 ผลการทดลองเปรียบเทียบกับระบบอุดมคติ

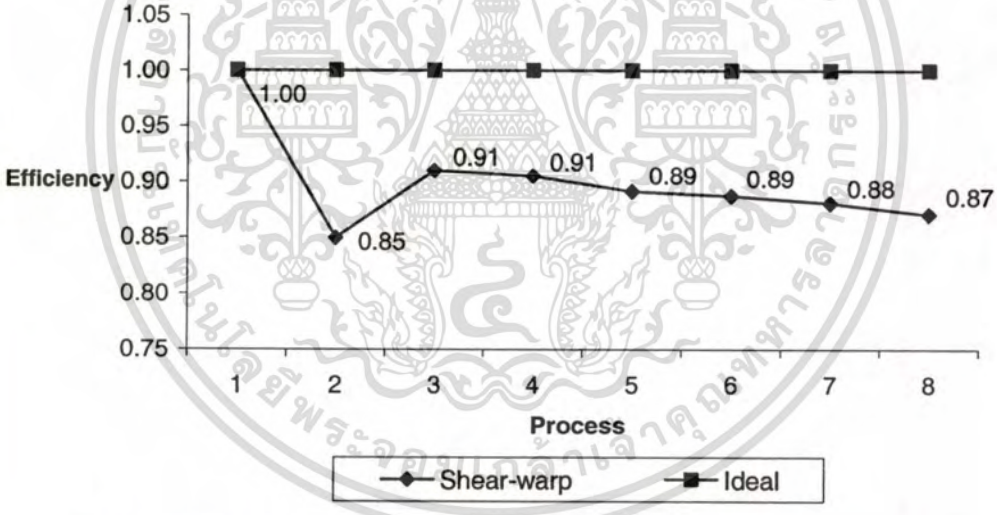
การทดลองในหัวข้อนี้จะเป็นการวัดประสิทธิภาพของระบบที่ใช้ในการทดลองเปรียบเทียบกับระบบที่เป็นอุดมคติ ซึ่งอธิบายโดยสมการที่ 7.1, 7.3 และ 7.4 ดังนี้



รูปที่ 7.6 แผนภูมิแสดงเวลาที่ลดลงของการทดลองเปรียบเทียบกับระบบอุดมคติ



รูปที่ 7.7 แผนภูมิแสดงอัตราการเพิ่มความเร็วของการทดลองเปรียบเทียบกับระบบอุดมคติ



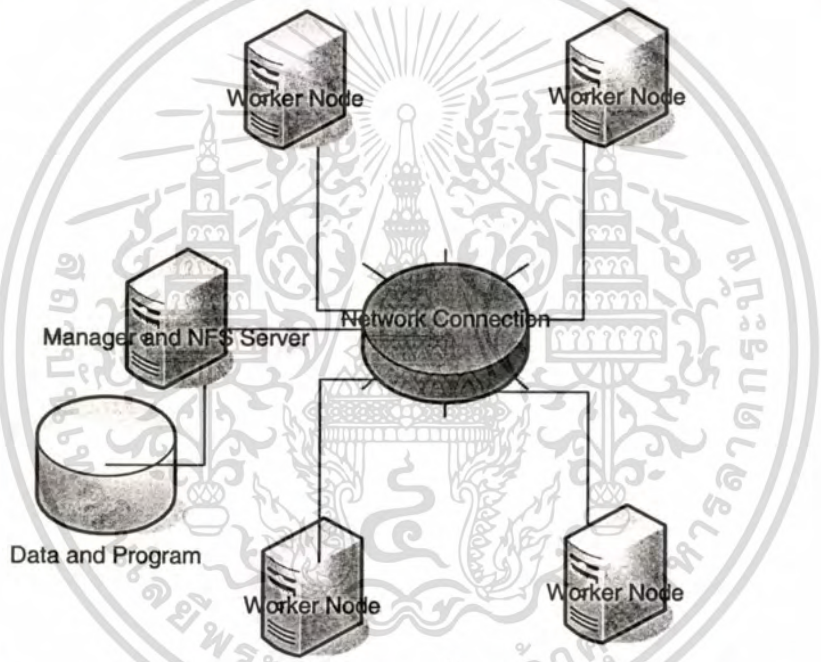
รูปที่ 7.8 แสดงประสิทธิภาพของระบบที่ใช้ทดลองเปรียบเทียบกับระบบอุดมคติ

แผนภูมิที่ 7.6 เวลาที่ Process เท่ากับ 1 นั้นหมายถึง เวลาที่ได้จากการคำนวณด้วยวิธี Sequential Algorithm บนคอมพิวเตอร์หน่วยประมวลผลเดี่ยว แต่มีคุณสมบัติทางฮาร์ดแวร์อื่น ๆ เหมือนกับระบบคอมพิวเตอร์ขนาน และจากผลการทดลองจะเห็นได้ว่าระบบที่สร้างขึ้นเพื่อใช้ในการทดลองนี้มีประสิทธิภาพค่อนข้างสูงใกล้เคียงกับระบบอุดมคติ ดังแสดงในแผนภูมิทั้งสามที่ผ่านมา

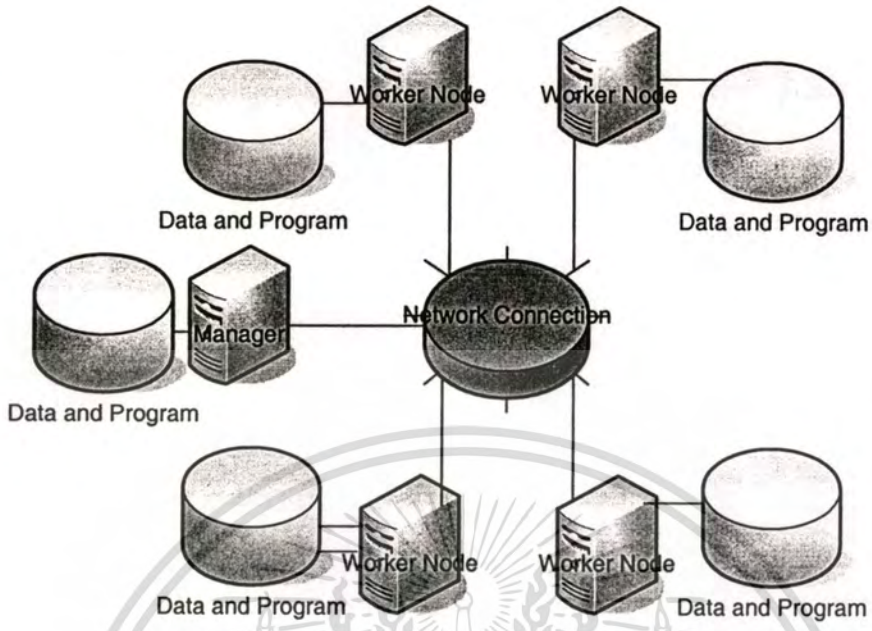
7.4.2 การทดลองเปรียบเทียบระหว่างการเก็บข้อมูลแบบศูนย์กลางและแบบกระจาย

ในการทดลองนี้จะทำการทดลองบนระบบสองระบบที่แตกต่างกันในส่วนของการเก็บข้อมูลปริมาตรและโปรแกรมการทำงาน โดยที่ระบบหนึ่งจะทำการเก็บข้อมูลทุกอย่างไว้ที่ศูนย์กลางคือ เครื่องที่ทำหน้าที่เป็น Manager และเครื่องอื่น ๆ จะทำการดึงเอาข้อมูลปริมาตรและโปรแกรมจากศูนย์กลางผ่าน NFS Service (Network File System) ซึ่งเป็นวิธีการแบ่งปันไฟล์ (File Sharing) แบบพื้นฐานของระบบปฏิบัติการแบบ UNIX อยู่แล้ว ส่วนอีกระบบหนึ่งจะเก็บโปรแกรมและข้อมูลปริมาตรไว้ที่แต่ละเครื่องเอง เมื่อระบบทำงานก็จะทำการอ่านค่าต่าง ๆ จากหน่วยความจำหลัก (Hard disk) โดยตรง

โครงสร้างของระบบที่ใช้ในการทดลองจริงสามารถแสดงได้ดังรูปที่ 7.9 และรูปที่ 7.10



รูปที่ 7.9 ระบบที่มีการเก็บข้อมูลปริมาตรและโปรแกรมการทำงานไว้ที่ศูนย์กลาง

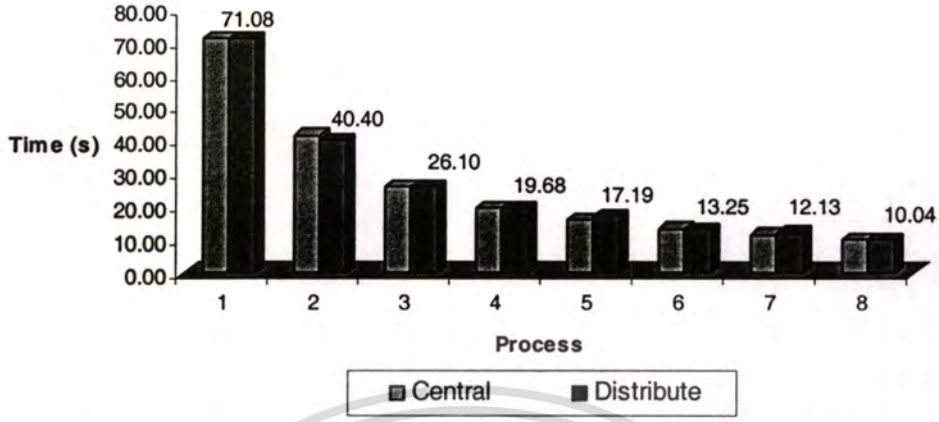


รูปที่ 7.10 ระบบที่มีการเก็บข้อมูลปริมาตรและโปรแกรมการทำงานแบบกระจาย

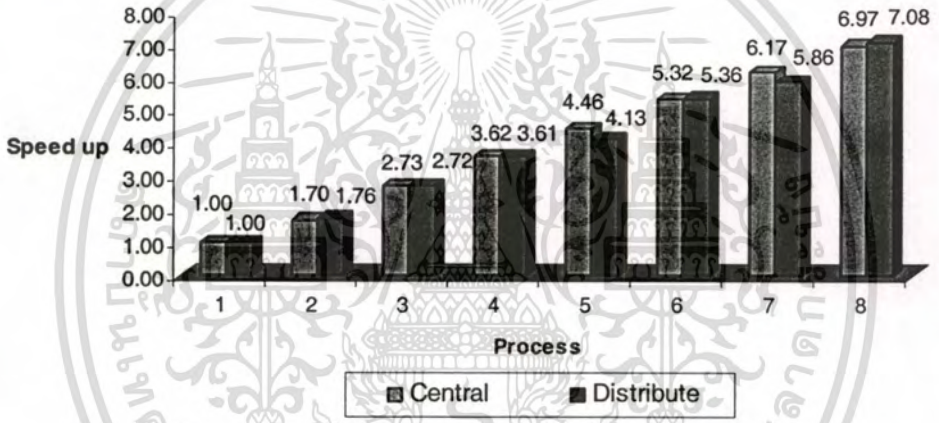
ดังนั้นสามารถกำหนดเงื่อนไขในการทดลองได้ดังนี้

ตัวแปรต้น	วิธีในการเก็บข้อมูลและโปรแกรม
ตัวแปรตาม	เวลาที่ใช้ในการประมวลผล
ตัวแปรควบคุม	Threshold = 0, Cubic Size = 128, Shear-warp

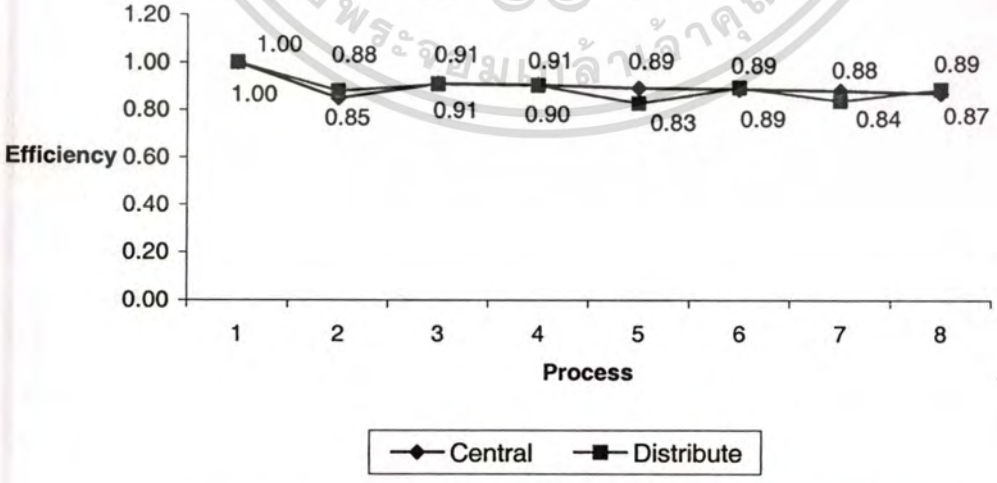
และจะได้ผลการทดลองดังแผนภูมิต่อไปนี้



รูปที่ 7.11 แผนภูมิแสดงเวลาที่ลดลงเปรียบเทียบสองระบบที่เก็บข้อมูลต่างกัน



รูปที่ 7.12 แผนภูมิแสดงอัตราความเร็วที่เพิ่มขึ้นของสองระบบที่เก็บข้อมูลต่างกัน



รูปที่ 7.13 แผนภูมิแสดงประสิทธิภาพของสองระบบที่เก็บข้อมูลต่างกัน

ดังนั้นจากผลการทดลอง สามารถสรุปผลการทดลองในหัวข้อนี้ได้ว่า การเก็บข้อมูลและโปรแกรมแบบกระจายนั้น จะทำให้ระบบสามารถทำงานได้รวดเร็วกว่าในช่วงที่ใช้โปรเซสทำงานจำนวนน้อย ๆ เท่านั้น แต่ถ้าหากจำนวนโปรเซสทำงานในระบบมากขึ้นความเร็วรวมของระบบจะใกล้เคียงกัน และเมื่อพิจารณาแผนภูมิของอัตราการเพิ่มของความเร็วแล้วจะเห็นได้ว่าการเพิ่มความเร็วของระบบที่มีการจัดเก็บข้อมูลที่ส่วนกลางนั้นจะมีอัตราการเพิ่มที่ดีกว่า ส่วนประสิทธิภาพของระบบทั้งสองในรูปที่ 7.13 นั้นมีค่าใกล้เคียงกัน แต่ของระบบที่เก็บโปรแกรมและข้อมูลอยู่ที่ศูนย์กลางจะมีค่าที่เป็นเชิงเส้นมากกว่า ดังนั้นการทดลองที่เหลือทั้งหมดต่อจากนี้จะเป็นการทดลองบนระบบที่มีการเก็บข้อมูลอยู่ที่ศูนย์กลางทั้งหมด

การเก็บข้อมูลที่ศูนย์กลางนี้ยังมีข้อดีอีกอย่างคือ สะดวกในการจัดการข้อมูลต่าง ๆ ในระบบคลัสเตอร์ เพราะทุก ๆ โหนดของระบบจะมองเห็นข้อมูลชุดเดียวกัน เหมือนกันทุกโหนด แต่จะใช้การเก็บโปรแกรมแบบศูนย์กลางไม่ได้ถ้าหากระบบปฏิบัติการของแต่ละโหนดไม่เหมือนกัน เช่น มีทั้งโหนดที่เป็น Linux, Solaris หรือ HP-UX แต่การเก็บข้อมูลที่ส่วนกลางยังเป็นไปได้เหมือนเดิม เพราะ NFS เป็น Service ที่ทุกระบบปฏิบัติการแบบ UNIX รู้จัก

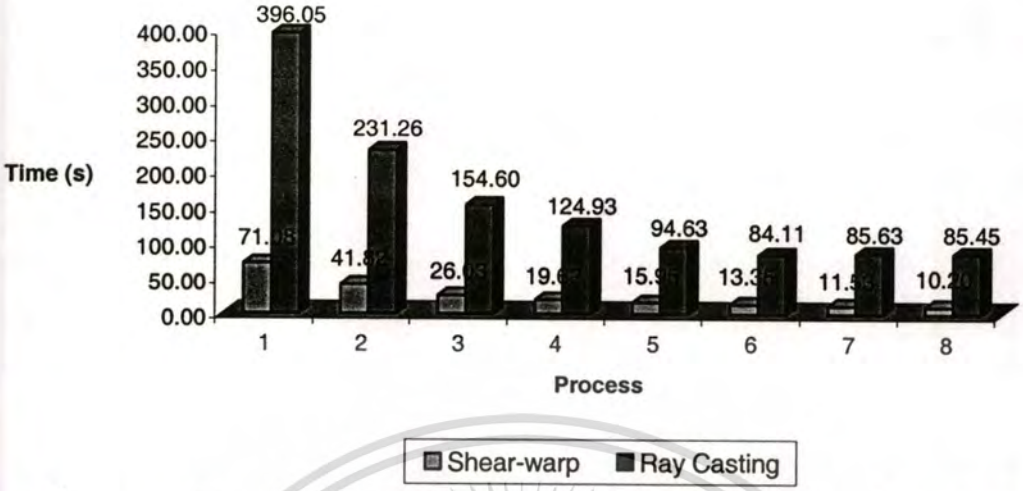
7.4.3 การทดลองเปรียบเทียบระหว่างวิธีการฉายแสงกับการแปลงเฉือนและบิด

ในการทดลองนี้จะเป็นการทดลองเพื่อให้เห็นความรวดเร็วที่เพิ่มขึ้นของการใช้วิธีการสร้างภาพเชิงปริมาตรแบบการแปลงเฉือนและบิด เพื่อทำการเปรียบเทียบกับวิธีการฉายแสง

สามารถกำหนดเงื่อนไขในการทดลองได้ดังนี้

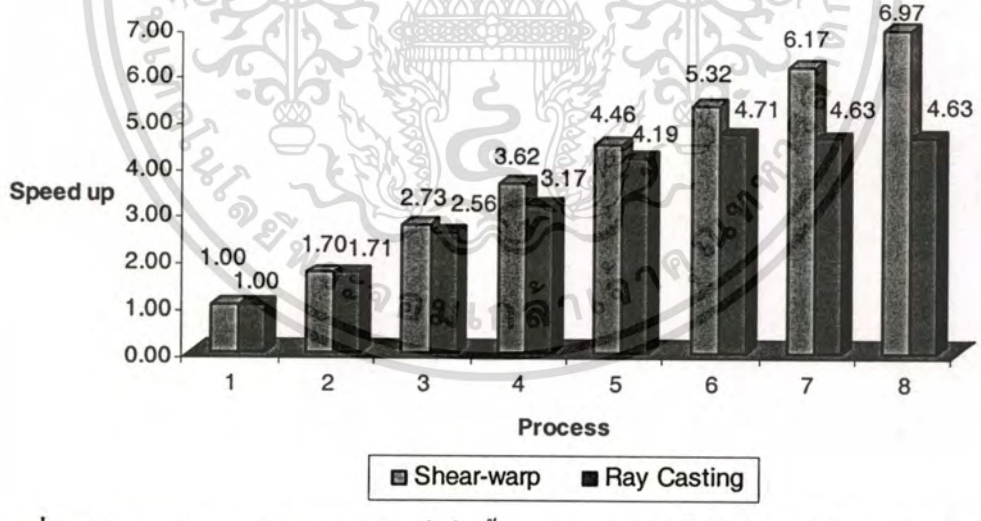
ตัวแปรต้น	วิธีในการสร้างภาพ
ตัวแปรตาม	เวลาที่ใช้ในการประมวลผล
ตัวแปรควบคุม	Threshold = 0, Cubic Size = 128, ข้อมูลและโปรแกรมอยู่ศูนย์กลาง

ได้ผลการทดลองดังแผนภูมิต่อไปนี้

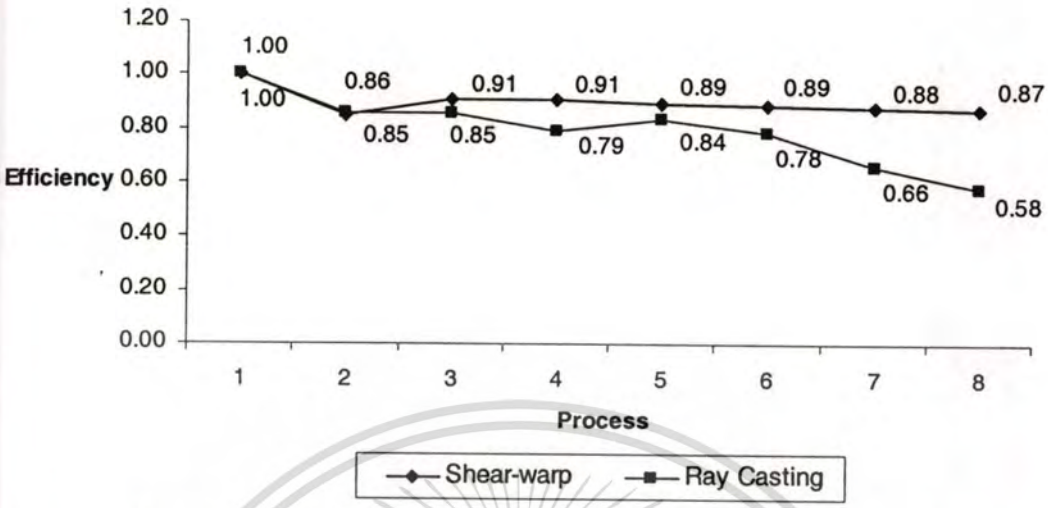


รูปที่ 7.14 แผนภูมิแสดงเวลาที่ใช้สร้างภาพผลลัพธ์ของสองระบบที่ใช้วิธีการสร้างภาพต่างกัน

ในรูปที่ 7.10 นั้น เวลาที่ Process เท่ากับ 1 นั้นหมายถึง เวลาที่ได้จากการคำนวณด้วยวิธี Sequential Algorithm บนคอมพิวเตอร์หน่วยประมวลผลเดี่ยวทั้งวิธีการแปลงเฉือนและบิด และวิธีการฉายแสง ส่วนเวลาที่โปรเซสตั้งแต่สองโปรเซสขึ้นไปนั้น คือ เวลาที่คำนวณบนคอมพิวเตอร์ขนาน



รูปที่ 7.15 แผนภูมิแสดงอัตราความเร็วที่เพิ่มขึ้นของสองระบบที่ใช้วิธีการสร้างภาพต่างกัน



รูปที่ 7.16 แผนภูมิแสดงประสิทธิภาพของระบบเมื่อใช้วิธีการสร้างภาพต่างกัน

จากผลการทดลองจะได้ว่า การสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิดนั้นสามารถทำให้การคำนวณรวดเร็วขึ้น 6.97 เท่า (เมื่อใช้โปรเซสทำงานจำนวน 8 โปรเซส) เมื่อเปรียบเทียบกับวิธีการสร้างภาพเชิงปริมาตรแบบฉายแสงบนคอมพิวเตอร์คลัสเตอร์ระบบเดียวกัน และทำให้การคำนวณรวดเร็วขึ้น 38.81 เท่า เมื่อเทียบกับการประมวลผลบนคอมพิวเตอร์เครื่องเดียวโดยใช้วิธีฉายแสง

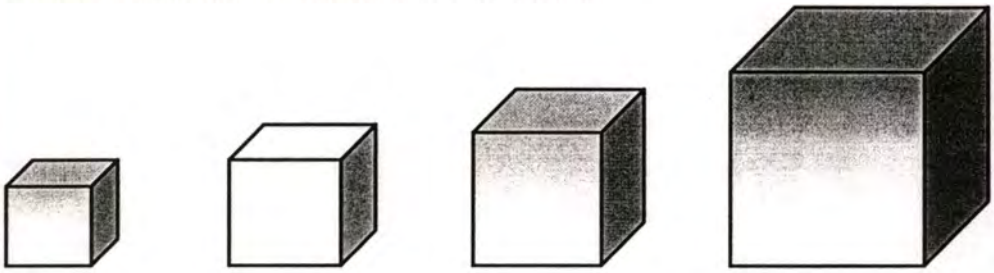
ดังนั้นจึงสามารถสรุปผลการทดลองได้ว่าการสร้างภาพเชิงปริมาตรโดยการแปลงเฉือนและบิดสามารถทำงานได้รวดเร็วกว่าการสร้างภาพโดยการฉายแสง และสามารถเพิ่มความรวดเร็วได้มากยิ่งขึ้นไปอีกเมื่อทำงานบนระบบคลัสเตอร์

7.4.4 การทดลองเปรียบเทียบเมื่อขนาดก้อนปริมาตรต่างกัน

เงื่อนไขอีกอย่างหนึ่งที่นำมาใช้ในการทดลองคือ ขนาดของก้อนปริมาตรการแบ่งงานที่มีขนาดแตกต่างกัน เพื่อจะทดสอบผลกระทบที่มีต่อความรวดเร็วในการทำงานของระบบ โดยมีเงื่อนไขในการทดลองดังนี้

ตัวแปรต้น	ขนาดของปริมาตรที่แบ่งงาน (Cubic Size)
ตัวแปรตาม	เวลาที่ใช้ในการประมวลผล
ตัวแปรควบคุม	Threshold = 0, จำนวน process = 8, Shear-warp, ข้อมูลและโปรแกรมอยู่ศูนย์กลาง

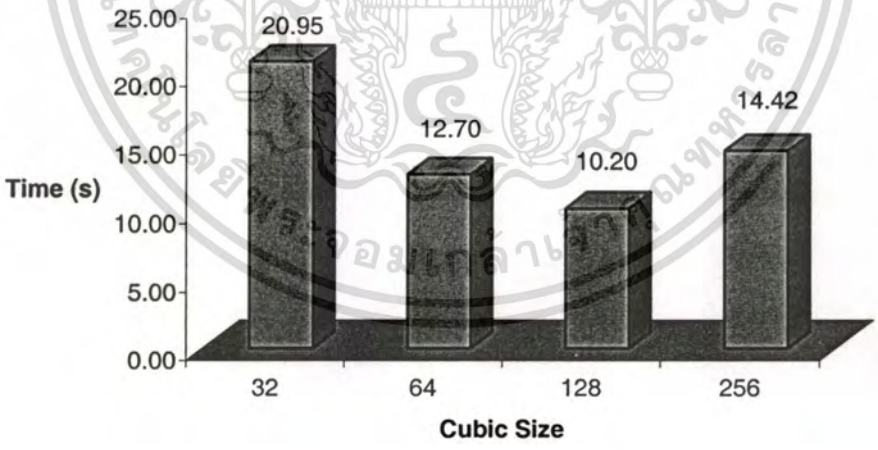
ขนาดของปริมาตรที่แบ่งจะมีทั้งหมด 4 ขนาด ดังตาราง



ขนาด^3 (Voxel)	Kbytes
32	32
64	256
128	2048
256	16384

รูปที่ 7.17 ขนาดของปริมาตรที่แบ่งในการทดลอง

จะได้แผนภูมิผลการทดลองดังนี้



รูปที่ 7.18 แผนภูมิแสดงเวลาที่ใช้ในการสร้างภาพเมื่อขนาดของก้อนปริมาตรแตกต่างกัน

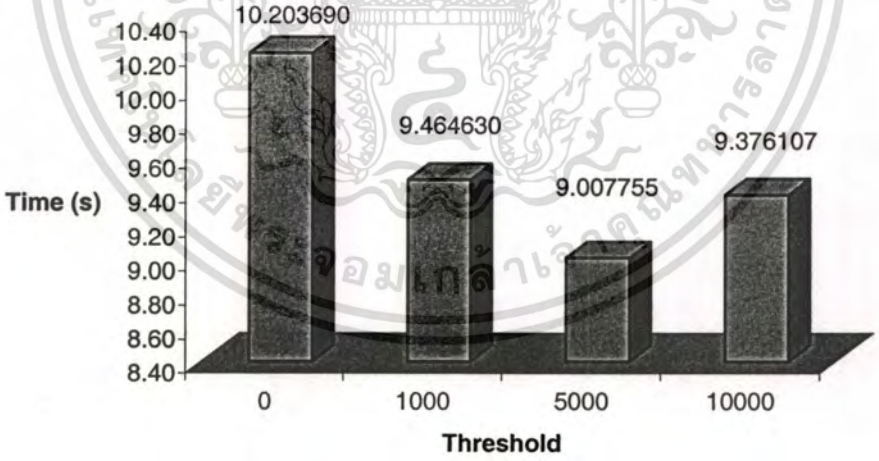
จากผลการทดลองจะได้ว่า เมื่อเราทำการแบ่งปริมาตรออกเป็นปริมาตรเล็ก ๆ ขนาดต่าง ๆ กัน จะมีผลต่อความเร็วในการสร้างภาพ โดยกรณีที่ดีที่สุดคือเมื่อปริมาตรเล็ก ๆ เหล่านี้มีขนาด

128x128x128 Voxels หรือคิดเป็นขนาด 2 MB ส่วนกรณีอื่น ๆ เวลาที่ใช้จะมากกว่า โดยเฉพาะเมื่อมีการแบ่งปริมาตรเป็นลูกบาศก์ขนาดเล็กมาก ๆ อย่าง 32x32x32 Voxels จะทำให้มีการสื่อสารระหว่างโปรเซสมากขึ้น เวลาที่ใช้ในการสื่อสารจึงมากขึ้นด้วย

7.4.5 การทดลองเปรียบเทียบเมื่อค่า Threshold ต่างกัน

การทดลองสุดท้ายจะเป็นการทดลองเพื่อทดสอบการใช้ค่า Threshold เข้ามาช่วยเพื่อให้การทำงานของระบบมีความรวดเร็วมากยิ่งขึ้น โดยเมื่อทำการอ่านค่าของปริมาตรเข้าเก็บในหน่วยความจำนั้น จะมีการหาค่าผลรวมของระดับเทา (Summation of the gray value) โดยหากก้อนปริมาตรใดมีค่าผลรวมระดับเทาน้อยกว่าค่า Threshold ที่ตั้งไว้จะไม่ถูกนำไปใช้ในขั้นตอนการสร้างภาพเชิงปริมาตร ทำให้ไม่ต้องสูญเสียเวลาโดยไม่จำเป็น เงื่อนไขในการทดลองมีดังนี้

ตัวแปรต้น	Threshold
ตัวแปรตาม	เวลาที่ใช้ในการประมวลผล
ตัวแปรควบคุม	Cubic Size = 128, จำนวน process=8, Shear-warp, ข้อมูลและโปรแกรมอยู่ศูนย์กลาง

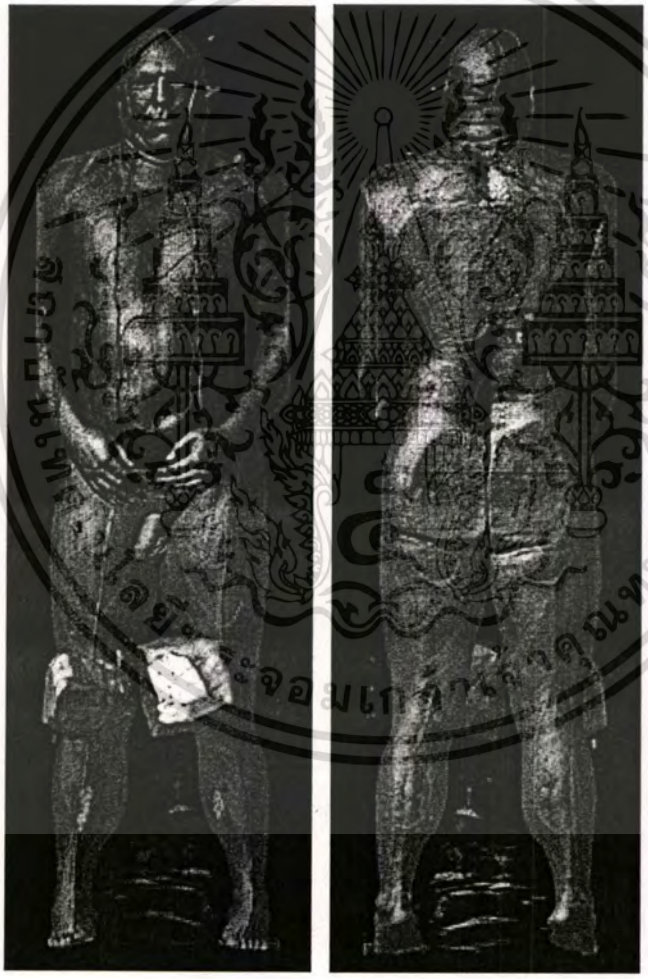


รูปที่ 7.19 แผนภูมิแสดงเวลาที่ใช้ในการสร้างภาพเมื่อใช้ค่า Threshold ต่างกัน

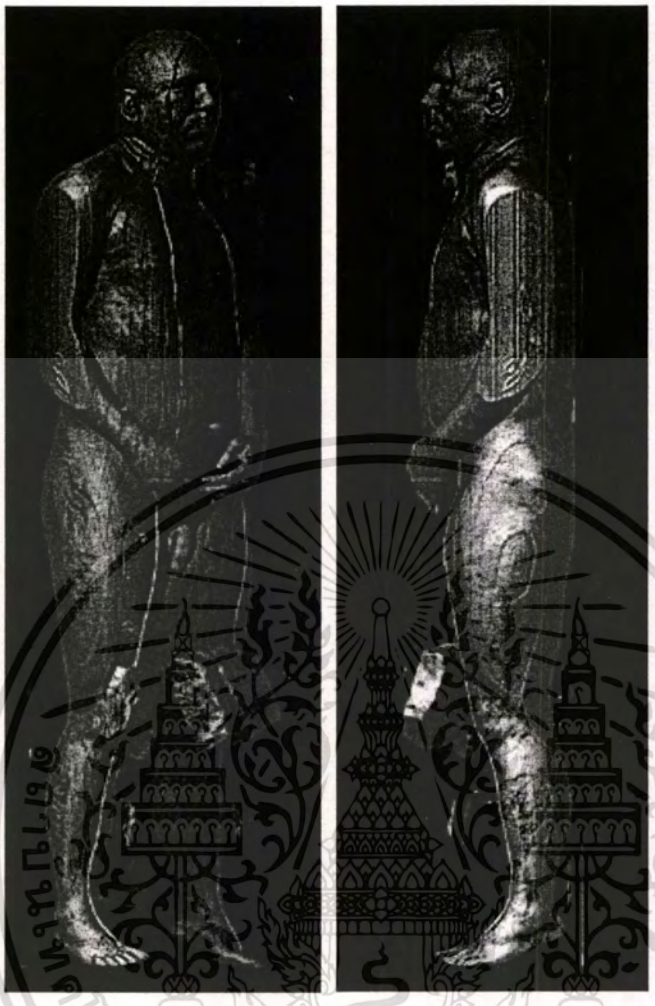
จากแผนภูมิสามารถสรุปผลการทดลองได้ว่า เมื่อใช้ค่า Threshold ที่ 5000 แล้วจะทำให้การสร้างภาพใช้เวลาน้อยที่สุด แต่เมื่อใช้ค่า Threshold สูงมากขึ้นก็ไม่ได้หมายความว่าความเร็วจะเพิ่มขึ้นด้วยเสมอไป

ในการทดลองนี้การพิจารณาค่า Threshold ทำการพิจารณาผลรวมของค่าระดับเทาเพียงอย่างเดียว ไม่ได้คำนึงถึงคุณภาพของภาพผลลัพธ์ ดังนั้นจึงควรมีความระมัดระวังในการใช้งานด้วย ถ้าหากภาพอินพุทไม่มีสัญญาณรบกวนอยู่เลย หรือต้องการคุณภาพของภาพผลลัพธ์ที่ต้องการความถูกต้องมากๆ

และในการทดลองได้ภาพผลลัพธ์ในแต่ละมุมมองแสดงได้ดังรูปที่ 7.20

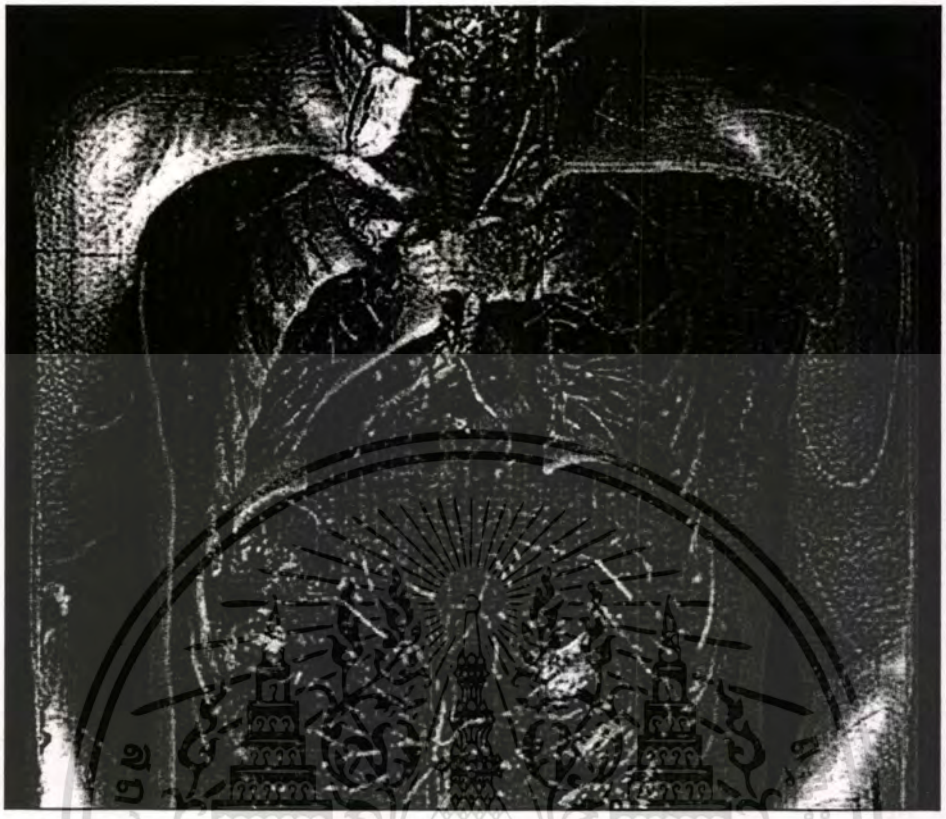


(a) (b)



(c) (d)

รูปที่ 7.20 ภาพผลลัพธ์ในแต่ละมุมมอง โดยที่ปริมาตรที่ใช้ในการสร้างภาพมีขนาด 587x341x1877 Voxel และภาพผลลัพธ์เป็นภาพสี 24 bit มีขนาด 1920x512 pixels



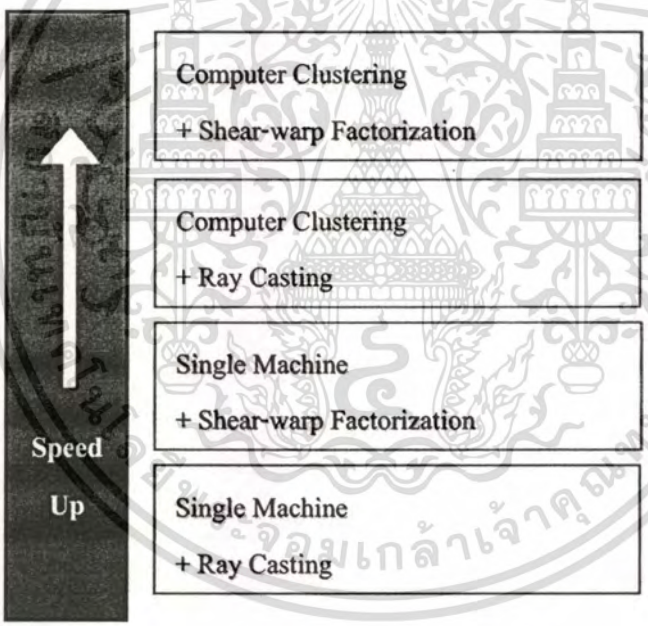
รูปที่ 7.21 ภาพผลลัพธ์ที่เลือกมาเฉพาะบางส่วนเพื่อให้เห็นรายละเอียดมากยิ่งขึ้น

รูปที่ 7.22 ภาพผลลัพธ์ในมุมมองอื่น ๆ

บทที่ 8
สรุปผลและแนวทางการพัฒนา

8.1 สรุปและวิจารณ์ผลการทดลอง

จากผลการทดลองที่ผ่านมาจะเห็นได้ว่าการสร้างภาพเชิงปริมาตรบนระบบคลัสเตอร์โดยใช้วิธีการแปลงเฉือนและบิดนี้ ได้ช่วยให้การสร้างภาพในแต่ละมุมมองสามารถทำได้รวดเร็วขึ้นกว่าวิธีการสร้างภาพแบบเดิมที่เคยมีมา ซึ่งถ้าหากทำการเปรียบเทียบกับงานวิจัยอื่นแล้วจะได้ดังแผนภูมิด้านล่างคือ



รูปที่ 8.1 แผนภูมิแสดงการเปรียบเทียบความเร็วในการสร้างภาพเชิงปริมาตรแบบต่าง ๆ

และจากที่กล่าวมาทั้งหมดนั้น ทำให้สามารถที่จะสรุปได้เป็นข้อ ๆ ดังต่อไปนี้

- วิทยานิพนธ์นี้เป็นการทดลองให้เห็นขบวนการเร่งความเร็วในการสร้างภาพเชิงปริมาตรขนาดใหญ่มากบนคอมพิวเตอร์แบบคลัสเตอร์

- การทดลองนี้แสดงให้เห็นว่าการสร้างภาพโดยใช้คอมพิวเตอร์แบบคลัสเตอร์ประมวลผลจะทำให้เวลาที่ใช้ในการคำนวณลดน้อยลง เป็นอีกทางเลือกหนึ่งในการสร้างภาพเชิงปริมาตร และยังสามารถที่จะพัฒนาความสามารถของระบบให้เพิ่มขึ้นได้อีก
- ระบบนี้ประกอบไปด้วยคอมพิวเตอร์หลาย ๆ เครื่อง เชื่อมต่อกันผ่านทางเครือข่ายความเร็วสูง โดยจะมีคอมพิวเตอร์หนึ่งเครื่องทำหน้าที่เป็นผู้กระจายงานเรียกว่า Manager และส่วนเครื่องอื่น ๆ ที่เหลือจะทำหน้าที่รับงานมาประมวลผลเรียกว่า Worker ซึ่งการแบ่งงานด้วยวิธีการนี้ทำให้ง่ายในการออกแบบโปรแกรมและออกแบบการสื่อสาร เพราะมีเพียงหนึ่ง Manager แล้วนอกจากนั้นเป็น Worker ทั้งหมด จะมีกี่ Worker ก็ทำได้โดยไม่ต้องลงมือเขียนโปรแกรมใหม่
- วิทยานิพนธ์ฉบับนี้ใช้วิธีการที่มีประสิทธิภาพในการสร้างภาพเชิงปริมาตร ทำให้ผลลัพธ์ที่ได้รวดเร็วมากขึ้น ทำให้สามารถลดเวลาในการคำนวณลงได้มาก
- วิธีการในการกระจายงานเรียกว่า Work Pool Load Balancing ทำให้สามารถใช้งานกับระบบคลัสเตอร์ได้ทั้งแบบที่เป็น Homogeneous และ Heterogeneous และไม่ต้องออกแบบโปรแกรมใหม่เพราะวิธีการนี้ใช้ได้ดีกับทั้งสองระบบอยู่แล้วนั่นเอง

8.2 แนวทางการพัฒนา

ในทางการพัฒนาที่น่าสนใจคือ การสร้างภาพเชิงปริมาตรบนระบบกริดคอมพิวเตอร์ เพราะเนื่องจากอัตราความก้าวหน้าของเทคโนโลยีนี้มีมากขึ้นทุกวัน และเทคโนโลยีเครือข่ายเองก็มีความรวดเร็วในการส่งข้อมูลมากขึ้นเช่นกัน ดังนั้นหากสามารถให้ระบบการสร้างภาพเชิงปริมาตรนี้ให้สามารถทำงานบนระบบกริดได้แล้ว จะทำให้การได้ภาพผลลัพธ์ในแต่ละมุมมองเป็นไปได้ง่ายดายมากยิ่งขึ้น

แต่ทั้งนี้การพัฒนาวิธีการสร้างภาพเชิงปริมาตรแบบใหม่ที่มีความรวดเร็วมากกว่าเดิม ก็เป็นอีกทางเลือกหนึ่งที่ยังมีความสำคัญอยู่ เพราะจะเป็นวิธีการที่ทำให้การสร้างภาพเชิงปริมาตรง่ายและรวดเร็วมากยิ่งขึ้นไปอีกเมื่อใช้ร่วมกับระบบคลัสเตอร์หรือกริดคอมพิวเตอร์

บรรณานุกรม

Image Processing and Volume Rendering:

Fletcher, P. A.&Robertson, P. K. 1993. "Interactive shading for surface]
and volume visualization on graphics workstations". Proceedings of Visualization '93. San Jose, pp. 291–298.

Lacroute, P. & Levoy, M. 1994. "Fast volume rendering using a shear-warp factorization of the viewing transformation", Computer Graphics Proceedings, Annual Conference Series (SIGGRAPH '94). Orlando, pp. 451–458.

Lacroute, P. 1995. "Fast volume rendering using a shear-warp factorization of the viewing Transformation" Ph.D. Dissertation, Stanford University.

Levoy, M. 1990. "Efficient ray tracing of volume data". ACM Transactions on Graphics. 9(3): 245–261.

Levoy, M. 1988. "Display of surfaces from volume data". IEEE Computer Graphics & Applications 8(3): 29–37.

Meagher, D. J. 1982. "Efficient synthetic image generation of arbitrary 3-D objects". Proceeding of the IEEE Conference on Pattern Recognition and Image Processing. pp. 473–478.

Nikolaidis, N. and Pitas, I. 2001. 3-D Image processing algorithm. John Wiley & Sons, Inc.

Reynolds, R. A., Gordon, D. & Chen, L.-S. 1987. "A dynamic screen technique for shaded graphics display of slice-represented objects". Computer Vision, Graphics, and Image Processing. 38(3): 275–298.

Subramanian, K. R. & Fussell, D. S. 1990. "Applying space subdivision techniques to volume Rendering". Proceedings of Visualization '90. San Francisco. California. pp. 150–159.

เกษมสุข เสพศิริสุข. 2544 "การสร้างภาพเชิงปริมาตรทางการแพทย์แบบเร็ว
โดยใช้การแปลงเฉือนและบิด"
วิทยานิพนธ์ วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

นนท์ บัณฑิตวงษ์. 2545. "การศึกษาการประมวลผลแบบขนานบนระบบคลัสเตอร์
ในการสร้างภาพเชิงปริมาตร"
วิทยานิพนธ์ วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมไฟฟ้า
บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

พิทยา อึงพินิจพงษ์. 2545. "การปรับปรุงวิธีการสร้างภาพกลับโดยใช้วิธีการทางพีชคณิต"
วิทยานิพนธ์ วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

วิทวัส วิทยชำนาญกุล. 2545. "การหาโครงร่างของวัตถุโดยใช้หลักการโทโมกราฟฟี
กับภาพถ่าย" วิทยานิพนธ์ วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

วรเทพ ไพบูลย์รัตนากร. 2544. "การเพิ่มความเร็วให้การสร้างภาพเชิงปริมาตรทางการแพทย์
โดยใช้การแปลงระยะทาง." วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิศวกรรมไฟฟ้า บัณฑิตวิทยาลัย,
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

Clustering System and Parallel Programming:

Buyya R. 1999 . "High Performance Cluster Computing: Architectures and Systems",
Volume 1. Prentice-Hall International, Inc.

Giertsen and Petersen. 1993. "Introduction to Programming on Distributed Memory
Multiprocessor", Computer Physics Communications, vol. 73, no. 1-3, pp.72-92

Kaufman, A. and Bakalash, R. 1988. "Memory and Processing Architectures
for 3D Voxel-Based Imagery", IEEE Computer Graphics and Applications.,
vol. 8, no. 11, pp. 10-23.

Lakshmivarahan, S. and Dhall, S K. 1990 . "Analysis and Design of Parallel Algorithm."
Prentice-Hall International, Inc.

Lester, B.P. 1993. "The art of parallel programming." Prentice-Hall International, Inc.

Potmesil, M. and Hoffert, E. M. 1989. "The Pixel Machine: A parallel Image Computer". Computer Graphics, vol. 23, no. 3, pp. 69-78.

Spector, D HM. 2000. "Building Linux Cluster." O'Reilly & Associates, Inc.

Pfister, H. Kaufman, A. and F. Wessels. 1995 . "Towards a Scalable Architecture for Real-Time Volume Rendering", Eurographics Workshop on Graphics Hardware, p. 123-130.

DesignDrug@Home. [Online] Available: http://www.gridbus.org/vlab

Earth quake simulation. [Online] Available: http://www.neesgrid.org

Foster, I. 1995. [Online] Available: http://www-unix.mcs.anl.gov/dbpp

Globus Project. [Online] Available: http://www.globus.org

SETI@home. [Online] Available: http://setiathome.berkeley.edu

TeraGrid . [Online] Available: http://www.teragrid.org

Network:

Gigabit Ethernet. [Online] Available: http://www.gigabit-ethernet.org/

InfiniBand . [Online] Available: http://www.infinibandta.org

Myrinet. [Online] Available: http://www.myri.com/myrinet/

Quadrics. [Online] Available: http://www.quadrics.com

Tools and utilities:

Ganglia. [Online] Available: http://ganglia.sourceforge.net

GNU Project. [Online] Available: http://gcc.gnu.org

LAM. [Online] Available: http://www.lam-mpi.org

MPI. [Online] Available: http://www.mpi-forum.org

MPICH. [Online] Available: http://www-unix.mcs.anl.gov/mpi/mpich

OpenMosix. [Online] Available: http://openmosix.sf.net

PETSc. [Online] Available: http://acts.nersc.gov/petsc

PLAPACK. [Online] Available: http://www.cs.utexas.edu/users/plapack

ScaLAPACK. [Online] Available: http://www.netlib.org/scalapack

Visible Human Project. [Online] Available: http://www.nlm.nih.gov

ตารางที่ ก.1 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์ (Cubic Size = 32)

Threshold	Worker	การทดลอง							
		โปรแกรมและข้อมูลอยู่ที่ศูนย์กลาง				โปรแกรมและข้อมูลกระจายอยู่ทุกโหนด			
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย
0	1								
	2	86.073199	80.658786	77.947865	81.559950	83.425293	81.480116	77.581542	80.828984
	3	51.940254	51.557016	51.584746	51.694005	50.371430	50.567853	50.008776	50.316020
	4	39.693623	38.996654	39.099524	39.263267	37.732425	37.674480	37.519608	37.642171
	5	32.106237	32.170570	31.765979	32.014262	32.834727	32.087827	31.123133	32.015229
	6	26.752034	26.860431	26.669488	26.760651	30.843976	25.778701	26.850272	27.824316
	7	23.594472	23.015353	23.048691	23.219505	22.112435	22.426803	22.118447	22.219228
	8	20.764939	21.457641	20.627004	20.949861	19.420283	19.492644	19.371527	19.428151
1000	1								
	2	63.397311	55.585892	51.927066	56.970090	52.278228	48.862388	48.718506	49.953041
	3	34.744939	34.938046	36.457945	35.380310	33.122485	32.826219	32.731129	32.893278
	4	27.735378	27.503366	26.190204	27.142983	24.598846	24.707099	24.634042	24.646662
	5	21.819526	21.398653	21.218530	21.478903	20.434156	20.303323	20.287756	20.341745
	6	17.948413	17.962010	17.924564	17.944996	17.274938	17.210731	17.110264	17.198644
	7	15.597043	16.442244	15.792140	15.943809	15.220367	14.687427	14.670122	14.859305
	8	13.973541	13.766324	13.905603	13.881823	13.296091	12.945072	12.986046	13.075736
5000	1								
	2	50.058908	49.855388	49.727078	49.880458	51.470585	47.841531	47.621544	48.977887
	3	33.682521	33.575481	33.583188	33.613730	32.439888	33.515203	31.975881	32.643657
	4	25.430912	25.483000	25.432400	25.448771	24.051155	24.220159	24.130469	24.133928
	5	20.821745	20.783757	20.781370	20.795624	19.824156	20.073026	20.098301	19.998494
	6	17.687621	17.721417	17.670424	17.693154	16.811581	16.666830	16.645386	16.707932
	7	15.579941	15.715278	15.389690	15.561636	14.376480	14.922445	14.483168	14.594031
	8	13.981240	13.533420	13.532238	13.682299	12.681103	12.680410	12.686166	12.682560
10000	1								
	2	57.603849	51.944720	56.763879	55.437483	50.246619	47.296181	47.391541	48.311447
	3	36.864071	38.873790	35.569916	37.102592	33.520491	31.759452	31.664664	32.314869
	4	26.817565	25.495113	25.162170	25.824949	24.112516	24.081116	24.491000	24.228211
	5	20.779006	20.518647	20.575488	20.624380	19.702403	19.581215	19.899084	19.727567
	6	17.317624	17.341124	17.370968	17.343239	16.693765	16.671151	16.794673	16.719863
	7	15.085671	15.540086	15.857649	15.494469	14.423311	14.217039	14.183258	14.274536
	8	13.616993	13.914027	13.774640	13.768553	12.617080	12.507064	13.154753	12.759632

ตารางที่ ก.2 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์ (Cubic Size = 64)

Threshold	Worker	การทดลอง							
		โปรแกรมและข้อมูลอยู่ที่ศูนย์กลาง				โปรแกรมและข้อมูลกระจายอยู่ทุกโหนด			
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย
0	1								
	2	62.187242	53.387279	53.749716	56.441412	53.148428	46.104132	48.094452	49.115671
	3	32.092286	51.791011	32.406468	38.763255	31.140260	30.894980	30.902108	30.979116
	4	23.751323	24.248471	23.554638	23.851477	23.248290	23.318689	23.256069	23.274349
	5	19.104218	19.070177	19.281964	19.152120	19.606299	18.849438	18.934266	19.130001
	6	16.952612	15.918695	16.080481	16.317263	15.742564	15.734419	15.752077	15.743020
	7	14.318771	14.361182	14.461101	14.380351	13.502272	13.563806	13.646182	13.570753
	8	12.501226	13.096641	12.500016	12.699294	11.920505	11.900803	11.882129	11.901146
1000	1								
	2	38.784192	37.546807	35.624640	37.318546	36.238400	34.974914	35.074010	35.429108
	3	23.840552	24.911551	23.966893	24.239665	23.497851	23.459844	23.486104	23.481266
	4	18.084486	18.505511	19.334661	18.641553	17.697886	17.647070	17.659435	17.668130
	5	15.374932	14.540881	14.546301	14.820705	14.356363	14.379772	14.317552	14.351229
	6	12.217452	12.216496	12.211574	12.215174	12.026464	12.009745	12.039660	12.025290
	7	11.558529	11.506648	11.479717	11.514965	10.335133	10.372427	10.371014	10.359525
	8	9.850962	9.677983	9.751391	9.760112	9.073679	9.204699	9.066481	9.114953
5000	1								
	2	34.551338	34.687342	34.388480	34.542387	37.189102	34.107135	33.970457	35.088898
	3	22.997043	23.086678	23.656622	23.246781	22.756958	22.763128	22.771814	22.763967
	4	18.080498	17.594318	125.605050	53.759955	17.125202	17.088510	17.106542	17.106751
	5	14.108302	14.172915	14.377869	14.219695	15.047284	13.992605	13.934387	14.324759
	6	65.416756	11.884480	11.919866	29.740367	11.653992	11.657007	11.678596	11.663198
	7	10.212978	10.164818	10.216784	10.198193	10.149744	9.992601	10.021049	10.054465
	8	8.991818	9.082542	8.960786	9.011715	8.784915	8.805091	8.877512	8.822506
10000	1								
	2	94.556416	34.158683	34.151477	54.288859	36.552003	33.755061	33.722426	34.676497
	3	22.920672	22.984508	23.840676	23.248619	22.708605	23.492197	22.621421	22.940741
	4	18.183255	17.321352	17.287975	17.597527	17.188563	17.020284	17.009337	17.072728
	5	14.074548	14.086866	14.063562	14.074992	13.848500	13.933891	13.834921	13.872437
	6	11.907064	11.923937	12.741357	12.190786	11.586025	11.634980	11.629329	11.616778
	7	10.112384	10.151186	10.153790	10.139120	9.937399	9.957220	9.928113	9.940917
	8	9.026512	8.944208	8.937506	8.969409	8.770553	8.747404	8.744951	8.754303

ตารางที่ ก.3 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์ (Cubic Size = 128)

Threshold	Worker	การทดลอง							
		โปรแกรมและข้อมูลอยู่ที่ศูนย์กลาง				โปรแกรมและข้อมูลกระจายอยู่ทุกโหนด			
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย
0	1				71.079300				71.079300
	2	46.772334	39.335757	39.366388	41.824826	38.943094	39.984831	42.275144	40.401023
	3	26.150477	25.846264	26.084642	26.027128	25.912517	25.940164	26.450222	26.100968
	4	19.619896	19.541113	19.718805	19.626605	19.691864	19.842270	19.491247	19.675127
	5	15.923973	15.879627	16.041616	15.948405	15.737742	19.318542	16.516253	17.190846
	6	13.320024	13.366688	13.366214	13.350975	13.332071	13.155841	13.258841	13.248918
	7	11.457066	11.562292	11.563470	11.527609	11.408980	11.503305	13.475828	12.129371
	8	10.423446	10.130455	10.057170	10.203690	10.089009	9.956980	10.060337	10.035442
1000	1								
	2	41.088978	35.657648	35.660917	37.469181	38.453815	42.011824	35.668591	38.711410
	3	23.827436	23.811992	24.181967	23.940465	23.863755	27.883132	24.035422	25.260770
	4	18.130560	17.916304	18.417233	18.154699	17.936992	17.965246	17.993437	17.965225
	5	15.066185	14.625494	14.758579	14.816753	14.659942	14.572000	14.484263	14.572068
	6	12.144372	12.621301	12.162636	12.309436	12.093132	12.226853	12.352405	12.224130
	7	10.403130	10.492250	10.517935	10.471105	10.507809	10.540960	10.439193	10.495987
	8	9.131314	9.200992	10.061584	9.464630	9.246026	9.222574	9.381085	9.283228
5000	1								
	2	40.678900	34.818352	34.871352	36.789535	37.261036	34.606659	34.718444	35.528713
	3	23.302479	23.585122	23.350473	23.412691	23.298157	23.318931	23.323207	23.313432
	4	17.551000	17.590914	17.666563	17.602826	17.564876	17.813849	18.450497	17.943074
	5	14.208307	14.293870	14.097459	14.199879	14.180943	14.103174	14.293471	14.192529
	6	11.931231	11.950976	11.840241	11.907483	12.654990	12.095685	11.931488	12.227388
	7	10.158790	10.211079	10.171633	10.180501	10.278337	10.190427	10.259671	10.242812
	8	9.092521	8.974858	8.955887	9.007755	8.950039	8.988981	9.006484	8.981835
10000	1								
	2	41.194553	34.683597	34.620919	36.833023	34.745992	34.930271	34.698605	34.791623
	3	23.281637	23.237792	23.325563	23.281664	25.513382	23.444219	23.315962	24.091188
	4	17.706622	17.798920	17.554625	17.686722	17.510644	17.459188	17.516264	17.495365
	5	15.192473	14.152561	14.104129	14.483054	14.117076	14.547367	14.356343	14.340262
	6	11.850339	11.814297	11.858332	11.840989	11.865549	11.810274	12.983461	12.219761
	7	10.163416	10.136461	10.149674	10.149850	10.148264	10.122570	10.219871	10.163568
	8	9.881151	9.155740	9.091431	9.376107	8.906002	9.079773	8.972290	8.986022

ตารางที่ ก.4 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการแปลงเฉือนและบิดบนระบบคลัสเตอร์ (Cubic Size = 256)

Threshold	Worker	การทดลอง							
		โปรแกรมและข้อมูลอยู่ที่ศูนย์กลาง				โปรแกรมและข้อมูลกระจายอยู่ทุกโหนด			
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย
0	1								
	2	67.274107	59.733722	53.468645	60.158825	54.578360	54.570635	54.703156	54.617384
	3	36.819057	36.073957	36.080650	36.324555	36.702516	36.222177	36.115492	36.346728
	4	27.487276	28.027426	28.013091	27.842598	27.957712	27.796320	26.843285	27.532439
	5	22.418056	22.695675	22.303967	22.472566	22.589032	22.756046	22.888862	22.744647
	6	18.394137	18.622365	18.456005	18.490836	19.875376	19.504698	18.789245	19.389773
	7	16.214082	16.186962	16.230925	16.210656	16.229593	16.232098	16.068331	16.176674
	8	14.328114	14.434366	14.482715	14.415065	13.871848	14.635074	14.376005	14.294309
1000	1								
	2	51.317245	50.802143	50.807757	50.975715	50.967622	50.979441	50.960079	50.969047
	3	34.986349	34.753862	35.053058	34.931090	34.751167	34.792175	34.733573	34.758972
	4	26.377278	27.222584	26.351364	26.650409	26.281078	26.294303	27.335180	26.636854
	5	21.296042	20.669991	21.785883	21.250639	21.777254	22.301645	21.841625	21.973508
	6	17.917321	17.943478	17.456899	17.772566	17.878799	17.869452	17.456705	17.734985
	7	15.592006	16.128209	15.503650	15.741288	15.407755	15.855086	15.421584	15.561475
	8	14.000396	13.935960	13.829668	13.922008	13.814062	13.836612	14.040664	13.897113
5000	1								
	2	47.886966	48.294189	49.258558	48.479904	47.871001	48.991967	48.761334	48.541434
	3	32.313396	32.285522	32.346086	32.315001	32.298005	33.604502	32.822321	32.908276
	4	24.455352	24.398001	24.420784	24.424712	24.410375	24.426237	24.416599	24.417737
	5	19.864085	19.955032	19.979780	19.932966	19.941480	19.902278	19.951270	19.931676
	6	17.191973	17.169076	17.215173	17.192074	17.213276	17.205456	17.272243	17.230325
	7	14.079362	14.851998	14.104492	14.345284	14.397449	14.715811	14.049254	14.387505
	8	12.947754	14.247743	13.973169	13.722889	12.919901	12.539115	12.908727	12.789248
10000	1								
	2	48.318709	48.538261	47.542798	48.133256	47.645340	47.942224	48.057777	47.881780
	3	32.827778	33.333439	32.282262	32.814493	32.861492	32.250083	33.058384	32.723320
	4	24.343852	24.690504	24.383936	24.472764	24.853150	24.348264	25.451911	24.884442
	5	19.945571	20.165206	19.950917	20.020565	19.960704	19.899499	19.975103	19.945102
	6	17.130884	17.232395	17.071854	17.145044	17.088496	17.157943	17.170511	17.138983
	7	14.306724	16.239209	14.092290	14.879408	14.057243	14.017387	14.572726	14.215785
	8	12.584616	12.502705	12.531666	12.539662	12.543484	12.500743	12.570124	12.538117

ตารางที่ ก.5 เวลาที่ใช้ในการสร้างภาพด้วยวิธีการฉายแสงบนระบบคลัสเตอร์

Block Size	Worker	การทดลอง							
		โปรแกรมและข้อมูลอยู่ที่ศูนย์กลาง				โปรแกรมและข้อมูลกระจายอยู่ทุกโหนด			
		ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ค่าเฉลี่ย
32	1								
	2	242.189402	237.129884	238.120989	239.146758	235.301478	232.345642	233.015973	233.554364
	3	158.596687	155.227356	155.160472	156.328172	158.812762	155.726010	155.479341	156.672704
	4	119.242178	117.283094	117.074513	117.866595	119.798800	116.902295	116.915265	117.872120
	5	93.628397	94.689040	95.454571	94.590669	95.063694	94.333889	95.287101	94.894895
	6	81.274375	87.364283	83.571328	84.069995	80.495307	79.947945	79.696375	80.046542
	7	84.217551	91.305321	100.194461	91.905778	70.223868	82.348853	75.516382	76.029701
	8	86.908591	90.170140	87.897495	88.325409	73.937405	67.837511	67.495817	69.756911
64	1								
	2	231.903511	232.160283	231.721665	231.928486	233.653890	233.964400	235.965637	234.527976
	3	155.289548	155.187335	158.686657	156.387847	156.797276	156.364000	156.199076	156.453451
	4	135.857264	119.857267	118.053298	124.589276	118.472983	117.941691	117.681953	118.032209
	5	94.717288	93.629193	95.007948	94.451476	95.131200	94.218815	94.331202	94.560406
	6	80.438646	83.262320	85.106078	82.935681	81.345750	80.741782	81.097688	81.061740
	7	72.730513	94.291400	90.335451	85.785788	75.565646	70.088379	70.471367	72.041797
	8	75.265087	83.664446	81.206728	80.045420	94.980991	85.665992	91.780262	90.809082
128	1				396.047000				
	2	232.456485	230.444126	230.868287	231.256299	235.838804	230.255542	230.726640	232.273662
	3	154.663005	154.521582	154.601998	154.595528	159.448966	157.400676	154.739382	157.196341
	4	138.626632	116.653684	119.497234	124.925850	118.812142	116.657014	117.324712	117.597956
	5	95.233714	93.877751	94.764798	94.625421	93.000477	93.200572	94.153528	93.451526
	6	79.238159	87.034969	86.058430	84.110519	82.846951	79.983729	82.069396	81.633359
	7	77.093820	86.156817	93.639430	85.630022	69.821922	73.410267	73.784685	72.338958
	8	82.972796	85.818568	87.550415	85.447260	63.926715	67.503796	66.233608	65.888040
256	1								
	2	236.772597	219.877899	219.724128	225.458208	223.607599	219.954506	219.769397	221.110501
	3	156.514350	157.932536	158.680040	157.708975	156.924086	157.996070	157.343194	157.421117
	4	146.207905	125.669533	126.238837	132.705425	125.281230	125.327000	126.633430	125.747220
	5	96.688252	97.577909	97.592690	97.286284	98.940345	96.694032	96.640900	97.425092
	6	94.794421	99.852798	94.531503	96.392907	93.598245	99.046560	94.580400	95.741735
	7	82.254703	80.452987	84.457126	82.388272	76.735436	77.135186	71.395385	75.088669
	8	77.512687	76.589650	76.021548	76.707962	74.960410	71.443124	72.562883	72.988806

ภาคผนวก ก.

ตารางผลการทดลอง

ภาคผนวก ข.

ผลงานที่ได้รับการตีพิมพ์

"Image Compression Using Vector Quantization"

Nopparat Pantsaena, Chinnapat Nantajiwakornchai, Manas Sangworasil and Thanasak Phanprasit

The 2002 International Technical Conference on Circuits / Systems, Computers and Communications (ITC-CSCC 2002). July 16-19,2002, Phuket, Thailand.

"A Large Scale Medical Volume Rendering on Clustering System"

Nopparat Pantsaena, Nont Banditwong, Chuchart Pintaviruj, Surapan Airphaiboon and Manas Sangworasil

The 2003 International Technical Conference on Circuits / Systems, Computers and Communications (ITC-CSCC 2003). July 7-9, 2003 Phoenix Park, Kang-Won Do, Korea.

"Fast Volume Rendering on Clustering System"

Nopparat Pantsaena, Akkapob Ngamlamiad, Chuchart Pintavirooj,Manas Sangworasil and Kasemsuk Sepsirisuk

World Congress on Medical Physics and Biomedical Engineering (WC2003), August 24-29, 2003. Sydney Australia.

"Fast Volume Rendering on Clustering System using A Shear-warp Factorization"

Nopparat Pantsaena, Prakit Engkakitti, Boonwat Attachoo, Chuchart Pintavirooj and Manas Sangworasil

The 3 rd International Symposium on Communications and Information Technologies (ISCIT-2003), 3-5 September 2003, BP Samila Beach Hotel and Resort, Songkhla, Thailand

ประวัติผู้เขียน

นายนพรัตน์ พันธุ์เสนา เกิดเมื่อวันที่ 24 สิงหาคม 2524 ที่จังหวัดขอนแก่น สำเร็จการศึกษาวิศวกรรมศาสตรบัณฑิต (อิเล็กทรอนิกส์) จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2544 หลังจากนั้นเข้าศึกษาต่อในระดับปริญญาโททันที และเข้าทำงานในตำแหน่งนักวิชาการคอมพิวเตอร์ สำนักวิจัยและบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ตั้งแต่วันที่ 1 พฤศจิกายน 2545 จนกระทั่งปัจจุบัน โดยรับผิดชอบงานทางด้าน UNIX/Linux Server