รูปบนพิกัดแกนคู่ขนาน

Image on Parallel Coordinate

นายคมสัน ว่องเขตรการณ์ รหัส 46360012

นายสุพจน์ ทาช้าง รหัส 46362141

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2549

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	รูปบนพิกัดแกนคู่ขนาน
ผู้ดำเนินโครงงาน	นายคมสัน ว่องเขตรการณ์ รหัส 46360012
	นายสุพจน์ ทาช้าง รหัส 46362141
อาจารย์ที่ปรึกษา	ดร.สุรเดช จิตประไพกุลศาล
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2549

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะกรรมการการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ
(ดร.สุรเดช จิตประไพกุลศาล)

.....กรรมการ
(ดร.พนมขวัญ ริยะมงคล)

.....กรรมการ
(ดร.ฮัครพันธ์ วงศ์กังแห)

หัวข้อโครงงาน	รูปบนพิกัดแกนคู่ขนาน		
ผู้ดำเนินโครงงาน	นายคมสัน	ว่องเขตรการณ์	รหัส 46360012
	นายสุพจน์	ทาช้าง	รหัส 46362141
อาจารย์ที่ปรึกษา	ดร.สุรเดช	จิตประไพกุลศาล	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2549		

บทคัดย่อ

โครงงานนี้ศึกษา และพัฒนาโปรแกรมการวาดรูปหลายมิติ โดยใช้การหลักการพิกัดแกนขนาน (Parallel Coordinate) ที่ทำให้สามารถวาดรูปที่มีมิติมากกว่า 3 มิติได้ ซึ่งโปรแกรมที่พัฒนาขึ้นสามารถวาดรูปเส้นตรง, วงกลม, วงรี, พาราโบลา, ไฮเปอร์โบลา, ซึ่งมีขนาด 2 มิติบนแกนคู่ขนาน โปรเจคไตล์, ลูกบาศก์, ทรงกลม ซึ่งมีขนาด 3 มิติบนแกนคู่ขนาน และการชนกันของเครื่องบิน ซึ่งมีขนาดมากกว่า 3 มิติบนแกนคู่ขนานได้ เพื่อหาพิกัดของเวลาที่เครื่องบินจะชนกัน และยังได้นำหลักการ Parallel Coordinate มาพัฒนาโปรแกรมเกี่ยวกับการทำ Image Processing อีกด้วย การเขียนโปรแกรมจะเน้นการใช้หลักการง่ายๆ เพื่อให้การแก้ไขปรับปรุงเป็นไปด้วยความสะดวก และเหมาะกับการนำไปศึกษาต่อ ซึ่งโปรแกรมที่พัฒนาขึ้นนี้จะเขียนด้วยภาษา C++ โดยใช้ Microsoft Visual Studio 6 และ Microsoft Visual Studio 2005

Project title	Image on Parallel Coordinate		
Name	Mr. Komsan	Wongkethgarn	ID. 46360012
	Mr. Supot	Thachang	ID. 46362141
Project advisor	Dr. Suradet	Jitprapaikulsan	
Major	Computer Engineering.		
Department	Electrical and Computer Engineering.		
Academic year	2006		

.....

Abstract

This project studies an alternative method the parallel coordinate system for drawing multi-dimensional images. Two programs are developed under this study: 1) a program to draw a linear line, circle, parabola, hyperbola, ellipse, projectile motion, on the parallel coordinate system. The program allows us to rearrange an axis to compare the impact between any two dimensions. 2) a program to apply the parallel coordinate system to study the color spectrum RGB and HIS of color images. Both programs are written using Microsoft Visual C++ and run on Microsoft Windows XP operating system.

กิตติกรรมประกาศ

ขอขอบพระคุณ ดร. สุรเดช จิตประไพกุลศาล อาจารย์ที่ปรึกษา และดร.พนมขวัญ ริยะมงคล และ ดร.อัครพันธ์ วงศ์กังแห อาจารย์ที่ปรึกษาร่วม ที่คอยให้คำปรึกษา ความช่วยเหลือตลอดจนคำแนะนำต่างๆในการทำโครงงานชิ้นนี้ สุดท้ายต้องขอขอบพระคุณอาจารย์ทุกท่านและเพื่อนๆทุกคนที่ยังไม่ได้เอ่ยนามที่ให้การสนับสนุนผู้จัดทำโครงงานให้สามารถทำโครงงานชิ้นนี้จนสำเร็จลุล่วงไปได้ด้วยดี

คณะผู้จัดทำ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญ

ในปัจจุบันการศึกษา การศึกษารูปทรงต่างๆ ในระบบพิกัดฉาก (Orthogonal Coordinate) มักจะจำกัดอยู่เพียงแค่ 1 – 3 มิติเท่านั้น เนื่องจากยากต่อการที่จะวาดรูปลงไปในมิติที่สูงกว่า ซึ่งการศึกษาในมิติที่สูงกว่า มักจะใช้วิธีการ Projection รูปนั้นลงบน 1 – 3 มิติ ซึ่งทำให้ข้อมูลบางส่วนได้สูญหายไป ระบบพิกัดขนาน (Parallel Coordinate) แก้ปัญหานี้ได้โดยการจัดให้แกน (axis) ต่างๆ ขนานกัน ดังนั้นจึงไม่จำกัดอยู่ที่ 3 มิติ แต่ทว่าการวาดรูปด้วยมือในระบบพิกัดขนาน เกิดความผิดพลาดได้ง่ายโดยเฉพาะเมื่อมีจำนวนมิติมาก ๆ

ดังที่กล่าวมา จึงทำให้เกิดการสร้างและพัฒนาโปรแกรมคอมพิวเตอร์สำหรับสร้างภาพบนระบบพิกัดขนาน โดยอาศัยหลักการทางคณิตศาสตร์มาใช้ในการพัฒนาโปรแกรมเพื่อง่าย รวดเร็วและแม่นยำกว่าการวาดเองด้วยมือ การศึกษาในเรื่องนี้เกิดประโยชน์มากมาย โดยเฉพาะในส่วนของการบิน ซึ่งมีความสำคัญต่อชีวิตมนุษย์ เช่น ระบบพิกัดขนานสามารถ plot จุดให้เห็นว่าตำแหน่งของเครื่องบิน 2 ลำ เมื่อบินสวนกัน ณ ตำแหน่งใดๆ เป็นอย่างไร จะชนกันหรือไม่ เป็นต้น

1.2 วัตถุประสงค์

1.2.1 นำความรู้ทางระบบพิกัดขนานมาประยุกต์ใช้

1.2.2 เพื่อนำความรู้ทางการเขียนโปรแกรมมาใช้

1.2.3 เพื่อพัฒนาโปรแกรมที่สามารถสร้างภาพบนระบบพิกัดขนานได้

1.3 ขอบข่ายของโครงงาน

1.3.1 ศึกษารายละเอียดของระบบพิกัดขนาน

1.3.2 เข้าใจและอธิบายถึงการคำนวณหาความสัมพันธ์ระหว่างระบบพิกัดฉากกับระบบพิกัดขนาน

1.3.3 สร้างซอฟต์แวร์ที่สามารถแสดงภาพหลายมิติ บนระบบพิกัดขนานได้

1.4 ขั้นตอนการดำเนินงาน

1.4.1 รวบรวมข้อมูลเกี่ยวกับ C++ และหลักการของระบบพิกัดขนาน

1.4.2 ศึกษารายละเอียดเกี่ยวกับ C++ และหลักการของระบบพิกัดขนาน

1.4.3 เขียนโปรแกรม

1.4.4 ทดสอบหาจุดบกพร่องของโปรแกรม

1.4.5 แก้ไขข้อผิดพลาด

1.4.6 ตรวจสอบรอบที่สอง

1.4.7 สรุปผลการทดลอง จัดทำรูปเล่มโครงงาน

1.5 แผนการดำเนินงาน

กิจกรรม	ปี 2548		ปี 2549								
	พ.ย.	ธ.ค.	ม.ค	ก.พ.	มี.ค.	เม.ย	พ.ค.	มิ.ย.	ก.ค.	ส.ค.	ก.ย.
1. รวมรวมข้อมูลเกี่ยวกับ C++ และหาหลักการของระบบพิกัดขนาน	↔	↔									
2. ศึกษารายละเอียดเกี่ยวกับ C++ และหลักการของระบบพิกัดขนาน			↔	↔	↔	↔					
3. เขียนโปรแกรม											
4. ทดสอบหาจุดบกพร่องของโปรแกรม							↔				
5. แก้ไขข้อผิดพลาดในส่วนต่างๆ ของโปรแกรม								↔			
6. ตรวจสอบรอบที่สอง									↔		
7. สรุปผลการทดลองและจัดทำรูปเล่มโครงงาน										↔	↔

1.6 ผลที่คาดว่าจะได้รับ

1.6.1 ความรู้ความเข้าใจเรื่องระบบพิกัดขนาน

1.6.2 ได้โปรแกรมที่สามารถใช้งานได้จริง

1.6.3 มีความรู้ความเข้าใจในการเขียนโปรแกรมมากขึ้น

1.6.4 เรียนรู้การวางแผนการทำงาน

1.6.5 ได้ประสบการณ์ในการทำงานร่วมกับผู้อื่น

1.7 งบประมาณ

1.7.1	ค่าเอกสารต่างๆ	1,000	บาท
1.7.2	ค่ารูปเล่มรายงาน	400	บาท
1.7.3	ค่าอุปกรณ์ อื่นๆ	600	บาท
	รวม	2,000	บาท

บทที่ 2

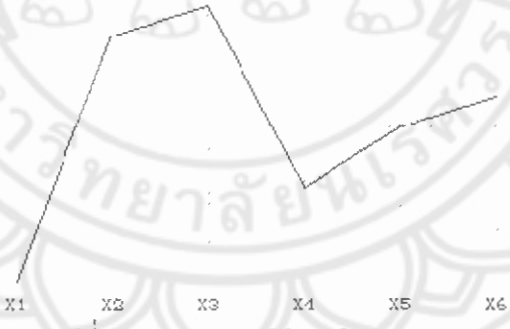
หลักการและทฤษฎี

ในบทนี้จะเป็นส่วนของหลักการและทฤษฎีต่างๆ ทั้งหมดที่ได้ศึกษาและนำมาใช้ในโครงงานนี้ โดยจะแบ่งออกเป็น 4 ส่วนหลักๆ คือ หลักการของระบบพิกัดขนาน สมการที่ใช้ในการวาดรูปต่างๆ หลักการของโหมดสีแบบ RGB และหลักการของโหมดสีแบบ HSI ซึ่งจะมีเนื้อหาดังนี้

2.1 หลักการของระบบพิกัดขนาน (Parallel Coordinate)

Alfred Inselberg เป็นผู้คิดค้นระบบพิกัดขนาน (Parallel Coordinate) ขึ้นมาในปี 1981 เพื่อเป็นแนวทางใหม่ในการแสดงข้อมูลที่มีขนาดหลายมิติ ซึ่งตั้งแต่ที่ได้มีการคิดค้นระบบพิกัดขนาน ขึ้นมานั้น สามารถทำให้งานหลายๆงาน ที่จำเป็นต้องใช้แกนมากกว่า 3 มิติ ในการทำงานนั้นๆ สำเร็จลงได้ ซึ่งในลักษณะของระบบพิกัดฉากแกนจะวางตั้งฉากกัน แต่ในลักษณะของระบบพิกัดขนานนั้น จะนำแกนมาวางขนานกัน โดยที่มีระยะห่างระหว่างแกนแต่ละแกนเท่าๆกัน โดยที่จะสมมุติให้ระยะห่างระหว่างแกนเป็น 1 หน่วย ซึ่งระบบพิกัดขนานนี้ สามารถแสดงได้ทั้งจุด เส้น และสามารถแสดงรูปได้มากกว่า 3 มิติ ซึ่งเป็นข้อดีของระบบพิกัดขนานโดยเฉพาะ

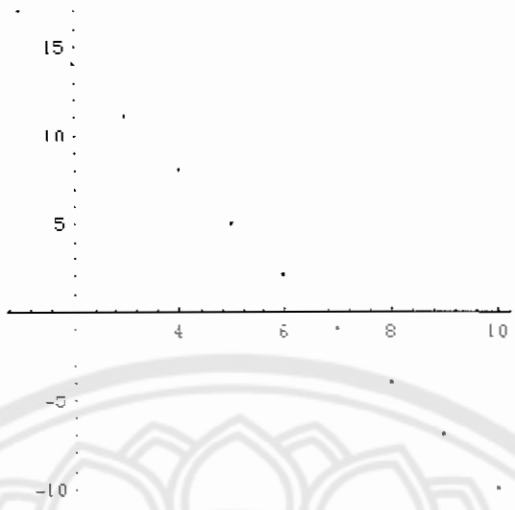
เช่นดังตัวอย่างเป็นภาพของจุดขนาด 6 มิติ (-5, 3, 4, -2, 0, 1) ในลักษณะของพิกัดขนาน ดังรูปที่ 2.1 โดยจะสังเกตได้ว่าระยะห่างระหว่างแกน ตั้งแต่ X1 ถึง X6 จะเท่าๆกัน



รูปที่ 2.1 จุดขนาด 6 มิติบนพิกัดขนาน [1]

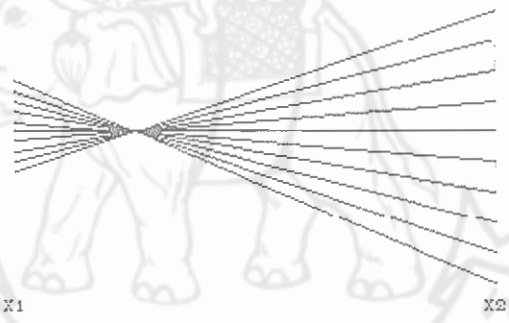
สำหรับจุดใดๆ $(x_1, x_2, \ldots, x_n)$ จะสามารถวาดได้โดยการลากเส้นจากตำแหน่ง x_1 ของแกน X1 เชื่อมต่อไปยังจุด x_2 บนแกน X2 แล้วลากจากจุด x_2 ไปยัง x_3 บนแกน X3 และลากเชื่อมต่อไปยังจุดต่อๆไปบนแกนถัดไป เรื่อยๆจนครบ แล้วจะได้จุดทั้งหมดโดยแสดงเป็นเส้นเชื่อมต่อกัน

การที่จะแสดงเส้นต่างๆในลักษณะของระบบพิกัดขนาน นั้นจะพิจารณาจากจุดบนแกน 2 มิติ เช่น สมการของเส้นตรง $x_2 = -3x_1 + 20$ สร้างรูปแบบพิกัดฉากจะได้ดังรูปที่ 2.2



รูปที่ 2.2 จุดของสมการเส้นตรง $x_2 = -3x_1 + 20$ บนพิกัดฉาก [1]

และสามารถแสดงบนระบบพิกัดขนาน ของจุดเหล่านั้นได้ดังรูปที่ 2.3



รูปที่ 2.3 จุดของสมการเส้นตรง $x_2 = -3x_1 + 20$ บนพิกัดขนาน [1]

เส้นทุกเส้น (เป็นจุดบนพิกัดฉาก) ในพิกัดขนาน จะตัดกันที่จุดตัดร่วม ซึ่งโดยปกติเส้นบนแกน 2 มิติ จะมีสมการเป็น $x_2 = mx_1 + b$ และจะได้จุดตัดที่ $(1/(1-m), b/(1-m))$ บนพิกัดขนาน ที่ m ไม่เท่ากับ 1 แต่ถ้า m = 1 จะไม่เหมาะที่จะแสดงบนระบบพิกัดขนาน เนื่องจากว่าเมื่อ m = 1 รูปที่ออกมาบนพิกัดขนานจะเป็นรูปเส้นตรงขนานกัน ซึ่งไม่สามารถหาจุดตัดได้

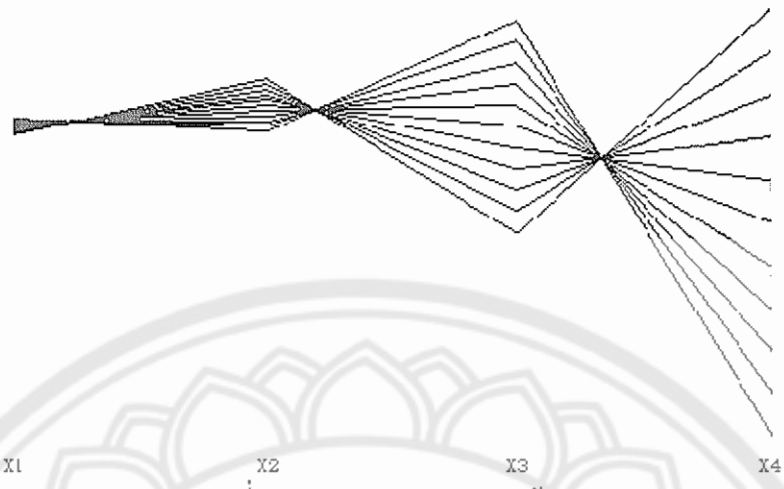
สำหรับ แกนที่เป็น n มิติ จะสามารถเขียนเป็นสมการอนุกรมได้ดังนี้

$$x_i = m_i x_{i-1} + b_i \quad \text{โดยที่ } i = 2,\ldots, n$$

แต่ละเส้นนั้นจะสามารถแสดงได้บนแกน 2 มิติ ดังนั้นจึงสามารถแสดงได้บนระบบพิกัดขนานโดยเซตของจุด n-1 ดังตัวอย่างนี้ พิจารณาแกน 4 มิติ โดยที่

$x_2 = -3x_1 + 12$ $\quad x_3 = -4x_2 + 48$ $\quad x_4 = -2x_3 - 54$

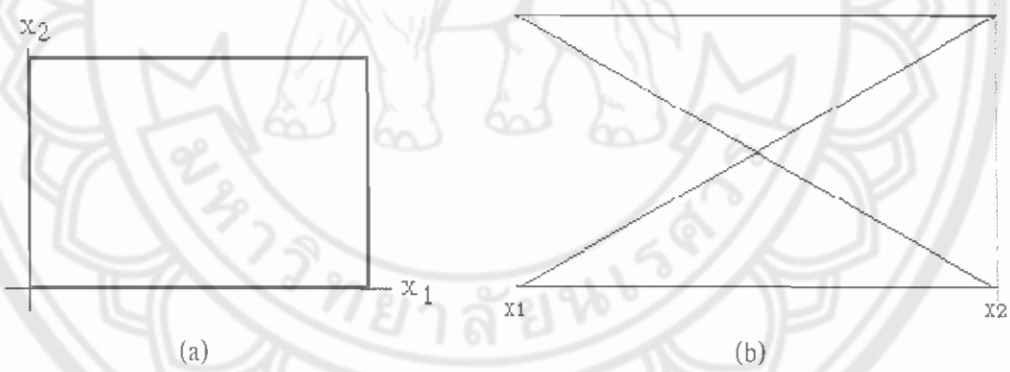
ซึ่งจะสามารถแสดงเป็นอนุกรมของจุด อยู่บนระบบพิกัดขนาน 4 มิติดังรูป 2.4



รูปที่ 2.4 รูป 4 มิติ ของสมการเส้นตรง

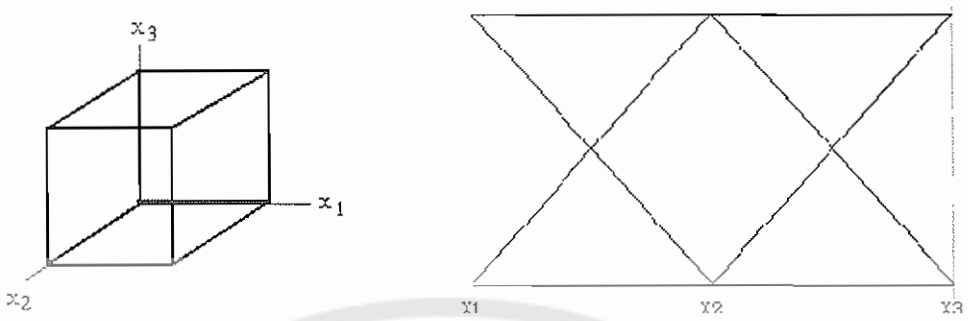
$x_2 = -3x_1 + 12$, $x_3 = -4x_2 + 48$ และ $x_4 = -2x_3 - 54$ บนพิกัดขนาน [1]

การใช้ระบบพิกัดขนานนั้น จะทำให้สามารถแสดงรูปลูกบาศก์ ที่มีหลายๆมิติได้อย่างง่ายดาย ดังเช่น รูปสี่เหลี่ยมบนแกน 2 มิติ แล้วนำมุมทั้ง 4 มุม ไป plot ลงบนระบบพิกัด ดังรูปที่ 2.5



รูปที่ 2.5 (a) รูปสี่เหลี่ยมบนฉาก (b) รูปสี่เหลี่ยมบนพิกัดขนาน [1]

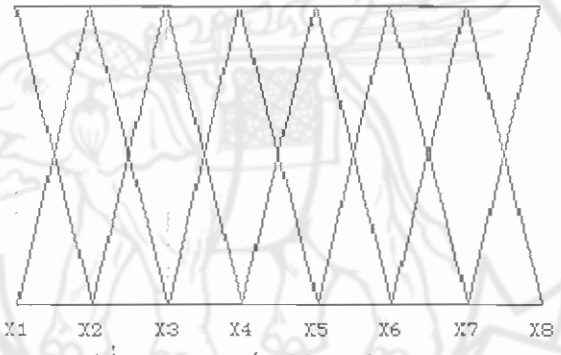
ลูกบาศก์ 3 มิติ ซึ่งมีมุม 8 มุม แสดงบนพิกัดขนาน ดังรูปที่ 2.6



(a) (b)

รูปที่ 2.6 (a) ลูกบาศก์บนพิกัดฉาก (b) ลูกบาศก์ 3 มิติ บนพิกัดขนาน [1]

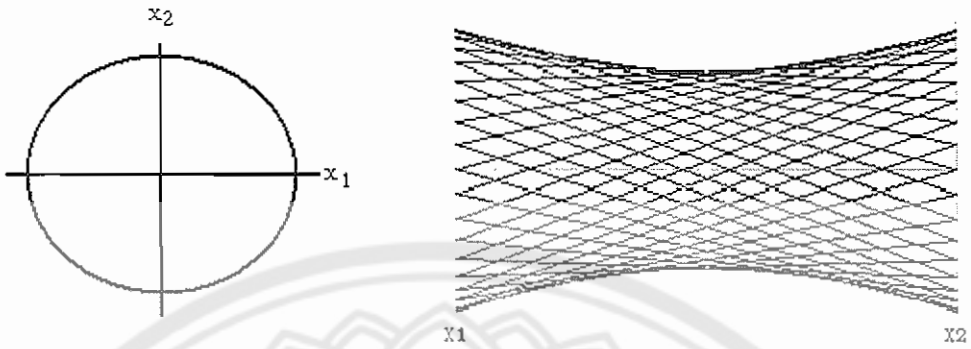
ลูกบาศก์ 8 มิติ ซึ่งมีมุม 256 มุม แสดงบนพิกัดขนาน ดังรูปที่ 2.7



รูปที่ 2.7 ลูกบาศก์ 8 มิติ บนพิกัดขนาน [1]

มีวิธีหนึ่งที่สามารถวาดรูปลูกบาศก์ที่มีหลายมิติได้ โดยการนำรูปสี่เหลี่ยม 2 มิติบนพิกัดขนานมาเรียงต่อกันในแนวนอน

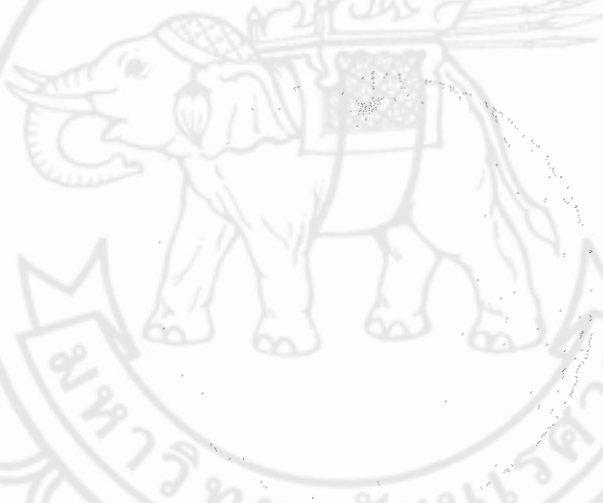
รูปวงกลม (Circle) สามารถวาดบนระบบพิกัดขนานได้ จากการนำจุดบนวงกลมในพิกัดฉาก มาวาดเป็นเส้นบนระบบพิกัดขนาน 2 มิติ ดังรูปที่ 2.8



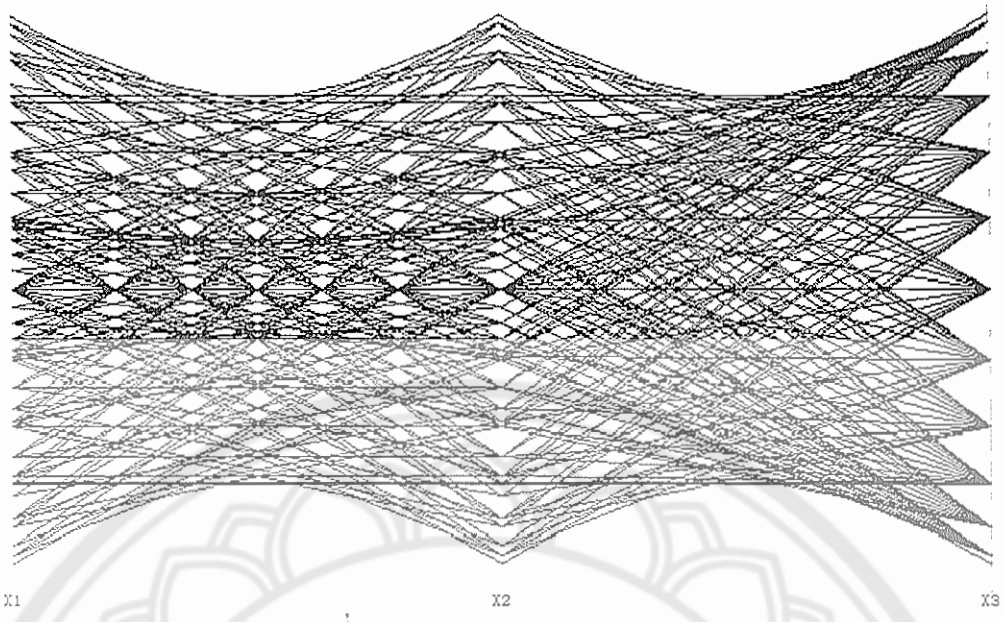
(a) (b)

รูปที่ 2.8 (a) รูปวงกลมบนพิกัดฉาก (b) รูปวงกลมบนพิกัดขนาน [1]

จุดบนทรงกลม สามารถแสดงบนระบบพิกัดขนาน ดังรูปที่ 2.10



รูปที่ 2.9 รูปทรงกลมบนพิกัดฉาก [4]



รูปที่ **2.10** รูปทรงกลมบนพิกัดขนาน [1]

ส่วนรูปกลม (Hyperball) ที่มีมิติมากกว่านี้ก็ทำได้โดยเพิ่มแกนเข้าไปอีก

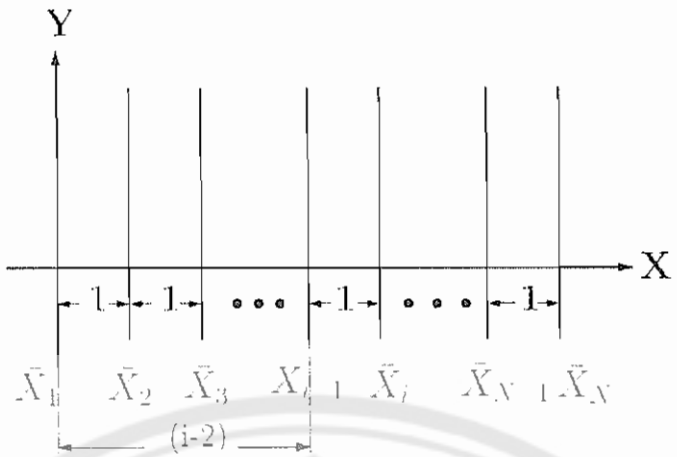
ใน R^3 เส้นคือ การตัดกันของ 2 แนวแกน ดังนั้น เส้นใน R^N มันก็คือ การตัดกันของ N-1 แนวแกนที่มากมายกลายเป็นกลุ่มของจุด ที่อิสระเชิงเส้นต่อกัน สามารถเขียนเป็นรูปแบบได้ดังนี้

$$\ell : \begin{cases} \ell_{1,2} & : \; x_2 = m_2 x_1 + b_2 \\ \ell_{2,3} & : \; x_3 = m_3 x_2 + b_3 \\ & \cdots \\ \ell_{i-1,i} & : \; x_i = m_i x_{i-1} + b_i \\ & \cdots \\ \ell_{N-1,N} & : \; x_N = m_N x_{N-1} + b_N \, . \end{cases}$$

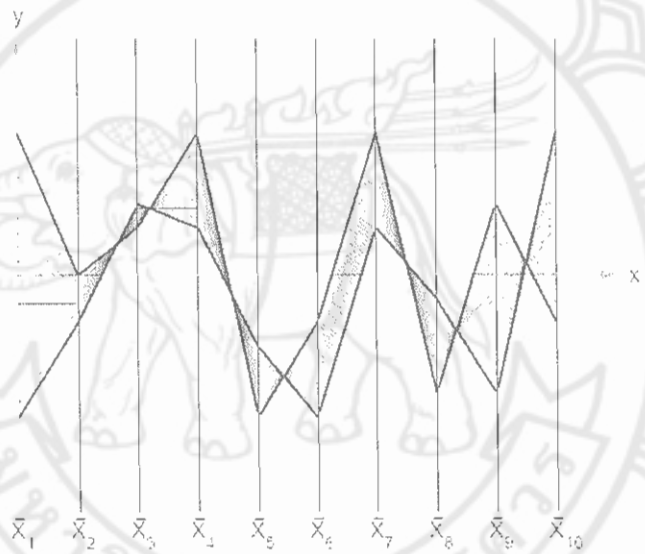
แต่ละสมการจะเป็นลักษณะของเส้นแต่ละเส้นที่มีการกระทำต่อกันที่ความชัน ในฐาน $x_{i-1}x_i$ จะได้ความสัมพันธ์ของเส้น $l_{i-1,i}$ โดยก่อนที่จะแสดง จุด <=> เส้น เราสามารถแสดงได้โดยจุด $l_{i-1,i}$ เขียนเป็นสมการได้ดังนี้

$$\bar{\ell}_{i-1,i} = ((i-2)(1-m_i)+1 \, , \, b_i \, , \, 1-m_i). \quad (1)$$

โดยที่ N-1 คือจุดที่ i = 2, …, N

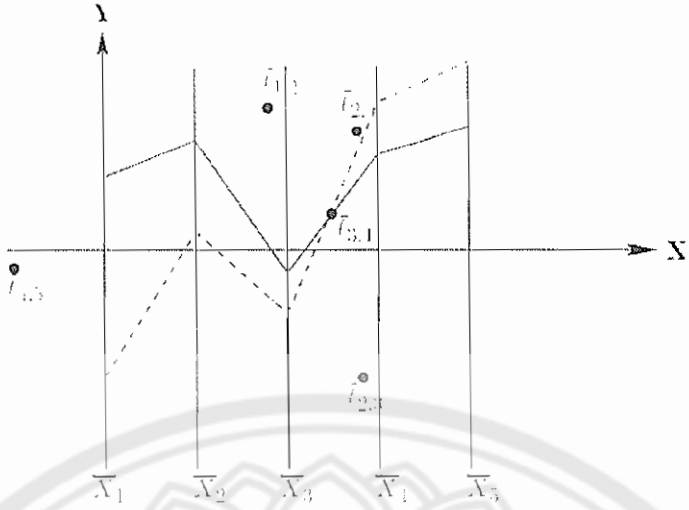


รูปที่ 2.11 แสดงระยะห่างมาตรฐานระหว่างแกน เท่ากับ 1 หน่วย [2]



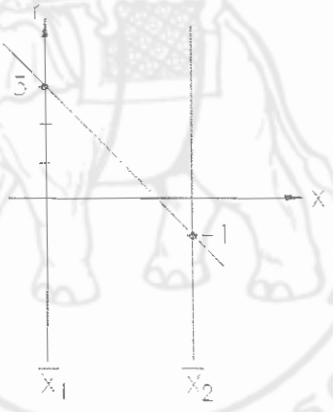
รูปที่ 2.12 รูปขนาด 10 มิติ [2]

จากรูปที่ 2.11 เป็นเส้นขนาด 10 มิติ ความหนาของเส้นที่แสดงให้เห็นจุดสิ้นสุดของมัน การชิดกันของเส้นตัวแปรที่แสดงให้เห็นประกอบด้วยตัวบอกตำแหน่งจุด 9 ตัว เป็นการนำเอาลำดับที่ดัดกันของแต่ละแกนที่ต่อเนื่องกัน เช่น $l_{1,2}$ ทางขวาของแกน X_2 และ $l_{6,7}$ คือจุดที่ดีมาก จุดที่แสดงให้เห็นคือ จุดที่มีการใช้ร่วมกันระหว่างแกน

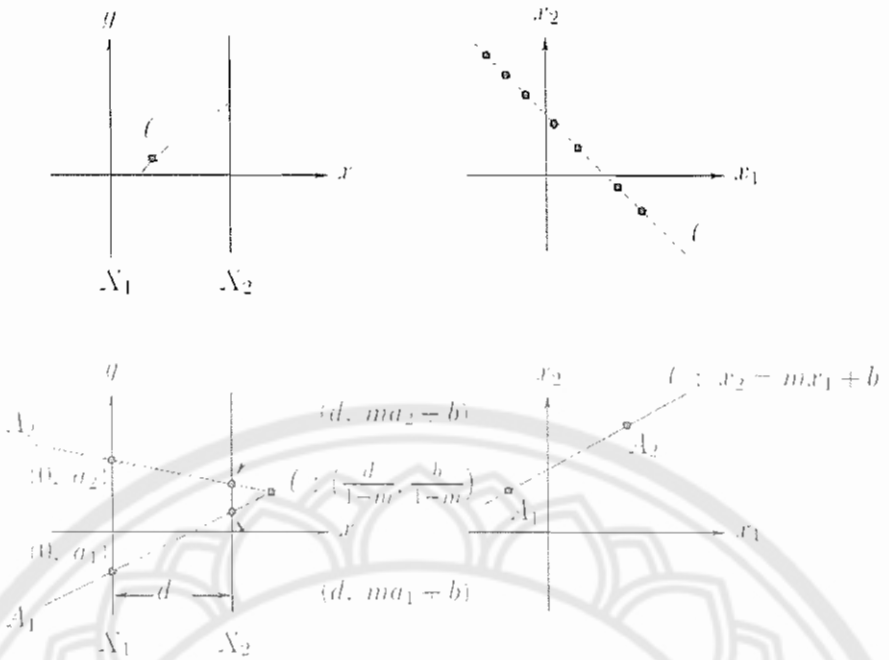


รูปที่ 2.13 รูปแสดงความสัมพันธ์ระหว่างจุด [2]

รูปนี้เป็น Algorithm สำหรับรูปทรงเส้นที่สัมพันธ์กัน ในกรณีของ $l_{2,5}$ แสดงให้เห็น N-1 จุด $l_{i+1,1}$ แสดงดังรูปที่ 2.13



รูปที่ 2.14 จุดบนพิกัดฉาก (3, -1) แสดงเป็นเส้นบนระบบพิกัดขนาน [3]



รูปที่ **2.15** ลักษณะความสัมพันธ์ระหว่างจุดกับเส้น [3]

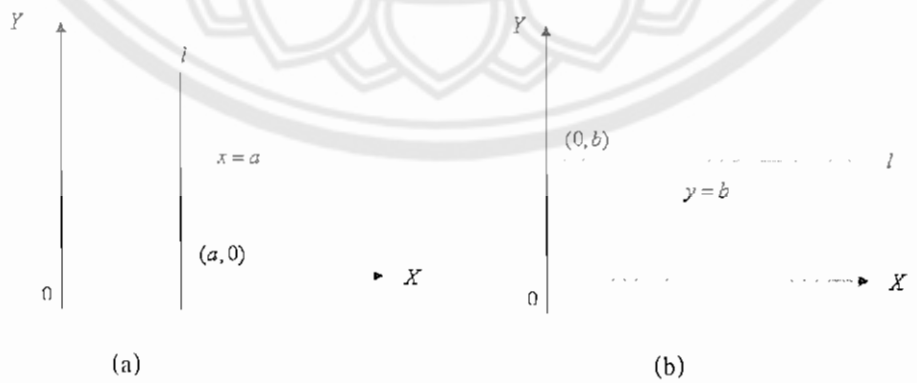
จะเห็นว่าลักษณะของมันจะตรงข้ามกัน โดยที่จุดบนพิกัดฉากจะกลายเป็นเส้นบนพิกัดขนาน และเส้นบนพิกัดฉากจะกลายเป็นจุดบนพิกัดขนาน

2.2 สมการที่ใช้ในโปรแกรมวาดรูปบนพิกัดขนาน

2.2.1 สมการเส้นตรง

สมการของเส้นตรง *l* คือ ความสัมพันธ์ระหว่าง x กับ y เมื่อ P(x, y) เป็นจุดใดๆ บนเส้นตรง *l* การเขียนเส้นตรงคือการหาความสัมพันธ์ระหว่าง x กับ y ตามลักษณะหรือเงื่อนไขที่กำหนดซึ่งอาจเขียนได้หลายแบบดังนี้

1. สมการเส้นตรงที่ขนานกับแกนพิกัด



(a) (b)

รูปที่ **2.16** (a) เส้นตรงขนานแกน y และ (b) เส้นตรงขนานแกน x

ถ้า l ขนานกับแกน y เป็นระยะ |a| หน่วย ดังรูปที่ 2.15 (a) สมการของเส้นตรง l คือ

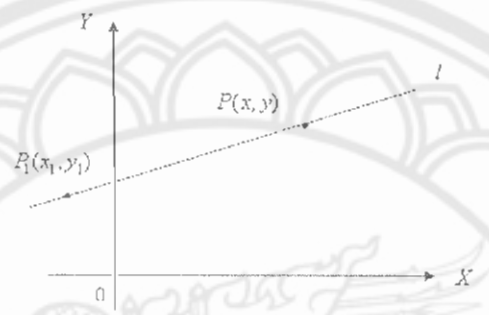
$$x = a$$

ถ้า l ขนานกับแกน x เป็นระยะ |b| หน่วย ดังรูปที่ 2.15 (b) สมการของเส้นตรง l คือ

$$y = b$$

2. สมการเส้นตรงแบบจุด – ความชัน (point – slope form)

ให้ l เป็นเส้นตรงที่ผ่านจุด $P_1(x_1, y_1)$ และมีความชันเป็น m และให้ P(x, y) เป็นจุดๆหนึ่งบนเส้นตรง l ดังรูปที่ 2.16



รูปที่ **2.17** เส้นตรงแบบจุด – ความชัน

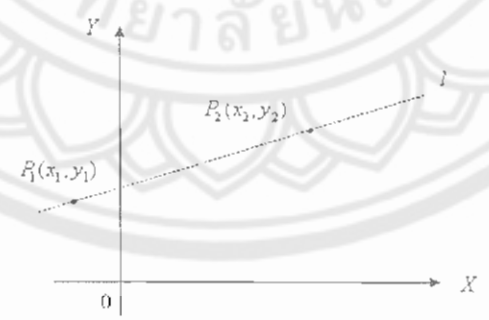
$$m = \frac{y - y_1}{x - x_1}$$

$$m(x - x_1) = y - y_1$$

ดังนั้น $$y - y_1 = m(x - x_1)$$

3. สมการเส้นตรงแบบสองจุด (two – point form)

ให้ l เป็นเส้นตรงที่ผ่านจุด $P_1(x_1, y_1)$ และ $P_2(x_2, y_2)$ เมื่อ $x_1 \neq x_2$ ดังรูปที่ 2.16



รูปที่ 2.18 เส้นตรงแบบสองจุด

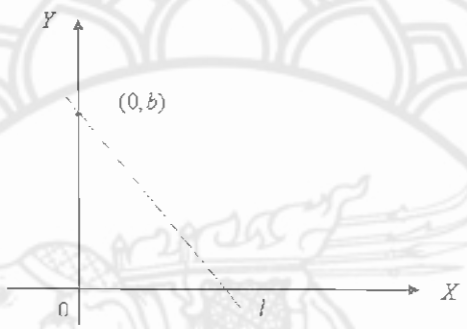
จากรูป $m = \frac{y_2 - y_1}{x_2 - x_1}$

จากสมการในแบบที่ 2 $y - y_1 = m(x - x_1)$

แทนค่า m ได้ $y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$

4. สมการเส้นตรงแบบความชัน – จุดตัดแกน (Slope – intercept form) ให้ l เป็นเส้นตรงที่ไม่ขนานกับแกน y

m เป็นความชันของ l ซึ่งตัดแกน y ที่จุด (0,b) ดังรูปที่ 2.17



รูปที่ 2.19 แบบความชัน – จุดตัดแกน

จากสมการเส้นตรงแบบจุดความชัน

$$y - y_1 = m(x - x_1)$$

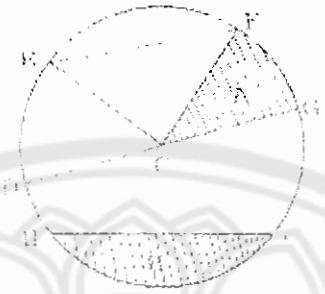
จะได้ $y - b = m(x - 0)$

$$y - b = mx$$

$$y = mx + b$$

2.2.2 วงกลม

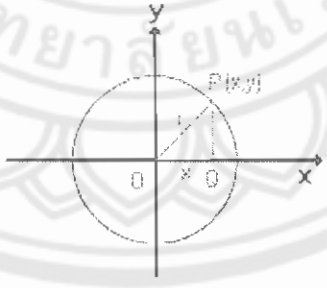
วงกลม คือ เซตของจุดบนระนาบซึ่งอยู่ห่างจากจุดคงที่จุดหนึ่งบนระนาบ เป็นระยะทางเท่ากันเสมอเรียกจุดคงที่ว่า จุดศูนย์กลางและเรียกระยะทางที่เท่ากันว่ารัศมีของวงกลม



รูปที่ **2.20** ส่วนประกอบของวงกลม

โดยที่ C คือ จุดศูนย์กลาง
DG คือ เส้นผ่านศูนย์กลาง
CF, CE, CG, CD คือ รัศมี
HI, EF คือ คอร์ด
พื้นที่ A เรียกว่า เซคเตอร์ (Sector)
พื้นที่ B เรียกว่า เซคเมนต์ (Segment)

วงกลมที่มีจุดศูนย์กลางอยู่ที่ (0, 0) รัศมียาว r หน่วย จะมีสมการดังนี้ $x^2 + y^2 = r^2$



รูปที่ **2.21** วงกลมที่มีจุดศูนย์กลางอยู่ที่ (0, 0)

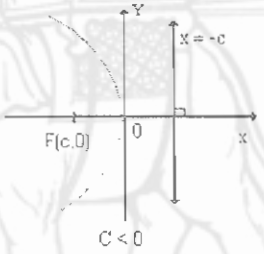
การวาดรูปวงกลมเมื่ออยู่ในระบบพิกัดขนาน จะใช้สมการเหมือนกับในระบบพิกัดฉาก แต่จะแปลงให้มาอยู่ในรูปของ $y = \pm\sqrt{r^2 - x^2}$ ก่อน เพื่อให้สะดวกเมื่อต้องการนำค่าที่ได้ไป plot

2.2.3 พาราโบลา

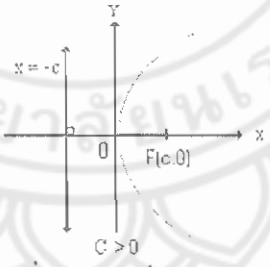
พาราโบลา คือ เซตของจุดบนระนาบซึ่งอยู่ห่างจากจุดคงที่จุดหนึ่งและอยู่ห่างจากเส้นตรงคงที่เส้นหนึ่งเป็นระยะทางเท่ากันเสมอ โดยที่พาราโบลามีส่วนประกอบต่างๆ ดังนี้

- จุดคงที่เรียกว่า จุดโฟกัส (Focus) เขียนแทนด้วย F
- เส้นตรงคงที่เรียกว่า เส้นไดเรกตริกซ์ (Directrix)
- เส้นตรงที่ลากผ่านโฟกัสและตั้งฉากกับเส้นไดเรกตริกซ์ เรียกว่าแกน (axis) ของพาราโบลา
- จุดที่กราฟตัดกับแกนพาราโบลาเรียกว่า จุดยอด (Vertex)
- ระยะทางจากจุดยอดไปยังโฟกัส เขียนแทนด้วย c
- เส้นตรงตั้งฉากกับแกนของพาราโบลาและผ่านจุดโฟกัสเรียกว่า ลาตัสเรกตัม (Latus rectum)

พาราโบลาที่มีจุดยอดอยู่ที่ (0, 0) มีแกน x เป็นแกนสมมาตร จะมีรูปสมการดังนี้ $y^2 = 4cx$ มีลักษณะดังรูปที่ 2.20 และรูปที่ 2.21

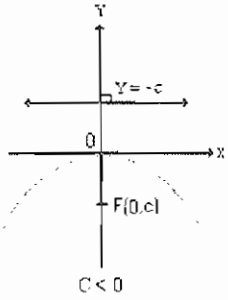


รูปที่ **2.22** พาราโบลาที่มีแกน x เป็นแกนสมมาตร โดยที่ $C < 0$

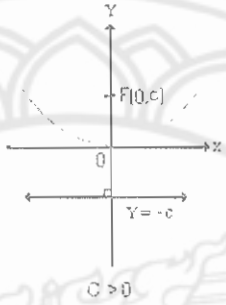


รูปที่ **2.23** พาราโบลาที่มีแกน x เป็นแกนสมมาตร โดยที่ $C > 0$

พาราโบลาที่มีจุดยอดอยู่ที่ (0, 0) มีแกน y เป็นแกนสมมาตร จะมีรูปสมการดังนี้ $x^2 = 4cy$ มีลักษณะดังรูปที่ 2.22 และ รูปที่ 2.23



รูปที่ 2.24 พาราโบลาที่มีแกน y เป็นแกนสมมาตร โดยที่ C < 0

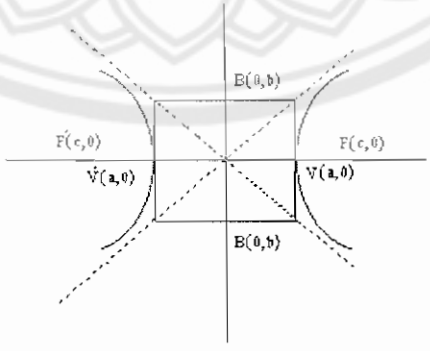


รูปที่ 2.25 พาราโบลาที่มีแกน y เป็นแกนสมมาตร โดยที่ C > 0

การวาดรูปของพาราโบลาเมื่ออยู่ในระบบพิกัดขนาน จะใช้สมการเหมือนกับการวาดในระบบพิกัดฉาก โดยที่จะต้องแปลงสมการให้มาอยู่รูปของ $x = \frac{y^2}{4c}$ เมื่อมีแกน x เป็นแกนสมมาตร และในรูป $y = \frac{x^2}{4c}$ เมื่อมีแกน y เป็นแกนสมมาตร

2.2.4 ไฮเพอร์โบลา

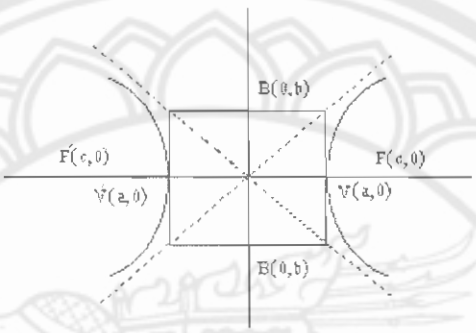
ไฮเพอร์โบลา คือ เซตของจุดบนระนาบซึ่งผลต่างของระยะทางจากจุดนี้ไปยังจุดคงที่สองจุดบนระนาบจะมีค่าคงตัวเสมอ



รูปที่ 2.26 ส่วนประกอบต่างๆ ของไฮเพอร์โบลา

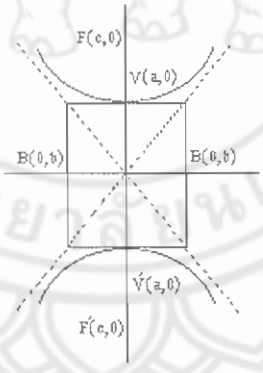
- เรียกจุด o ว่าจุดศูนย์กลาง
- เรียกจุด V' (-a, 0) และ V (a, 0) ว่า จุดยอด (Vertex)
- เรียกจุด F' (-c, 0) และ F (c, 0) ว่า จุดโฟกัส (Focus)
- เรียกจุด V' V ว่า แกนตามขวาง (Transverse Axis)
- เรียกจุด B' B ว่า แกนสังยุค (Conjugate Axis)
- เรียกจุด L_1 และ L_2 ว่า เส้นกำกับ (Asymtote)

ไฮเพอร์โบลาที่มีแกน x เป็นแกนสมมาตร จะมีรูปสมการดังนี้ $\frac{x^2}{a^2}-\frac{y^2}{b^2}=1$



รูปที่ 2.27 ไฮเพอร์โบลาที่มีแกนตามขวางอยู่บนแกน x

ไฮเพอร์โบลาที่มีแกน y เป็นแกนสมมาตร จะมีรูปสมการดังนี้ $\frac{y^2}{a^2}-\frac{x^2}{b^2}=1$

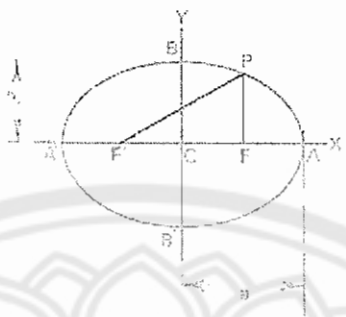


รูปที่ 2.28 ไฮเพอร์โบลาที่มีแกนตามขวางอยู่บนแกน y

การวาดรูปไฮเพอร์โบลาในระบบพิกัดขนานนั้น จะใช้สมการเหมือนกับการวาดบนพิกัดฉากเช่นกัน โดยที่จะต้องแปลงสมการให้มาอยู่ในรูปของ $y=\pm\sqrt{b^2\left(\frac{x^2}{a^2}-1\right)}$ เมื่อแกนตามขวางอยู่บนแกน x และอยู่ในรูปของ $x=\pm\sqrt{b^2\left(\frac{y^2}{a^2}-1\right)}$ เมื่อแกนตามขวางอยู่บนแกน y

2.2.5 วงรี

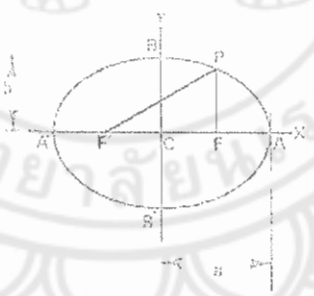
วงรี คือ เซตของจุดบนระนาบซึ่งผลบวกของระยะทางจากจุดใดๆ ในเซตนี้ไปยังจุดคงที่สองจุดมีค่าคงตัวเสมอ



รูปที่ **2.29** ส่วนประกอบต่างๆ ของวงรี

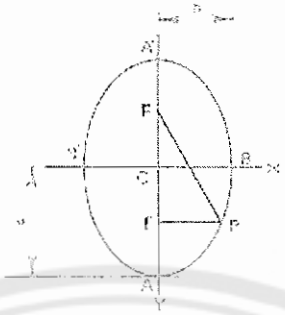
- เรียกจุด c ว่าจุดศูนย์กลาง
- เรียก A' และ A ว่า จุดยอด (Vertex)
- เรียกจุด F' และ F ว่าจุดโฟกัส (Focus)
- เรียก A' A ว่า แกนเอก (Major Axis)
- เรียก B' B ว่า แกนโท (Minor Axis)

วงรีที่มีจุดศูนย์กลางอยู่ที่ (0, 0) โดยที่มีแกนเอกอยู่บนแกน x จะมีรูปสมการดังนี้ $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$



รูปที่ **2.30** วงรีที่มีแกนเอกอยู่บนแกน x

วงรีที่มีแกนเอกอยู่บนแกน y จะมีรูปสมการดังนี้ $\frac{x^2}{b^2}+\frac{y^2}{a^2}=1$

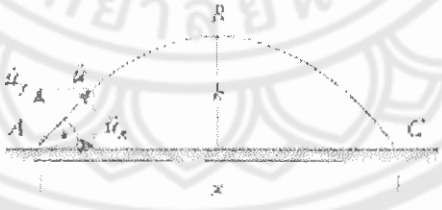


รูปที่ 2.31 วงรีที่มีแกนเอกอยู่บนแกน y

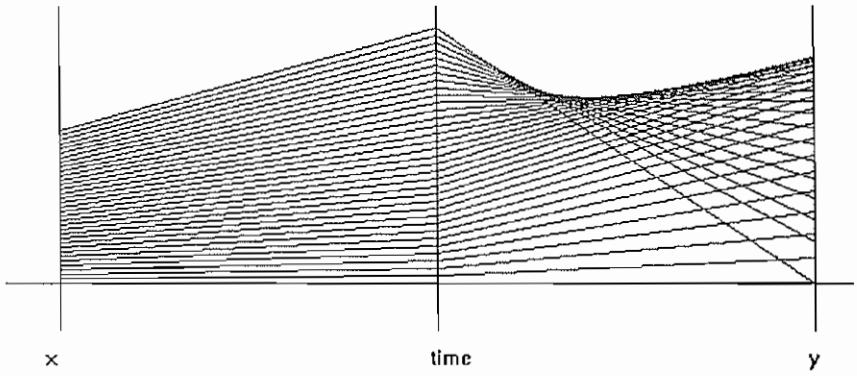
การวาดรูปวงรีในระบบพิกัดขนาน จะใช้สมการเหมือนกับการวาดวงรีบนระบบพิกัดฉาก โดยจะต้องแปลงสมการให้มาอยู่ในรูปของ $y=\pm\sqrt{b^2\left(1-\frac{x^2}{a^2}\right)}$ เมื่อแกนเอกอยู่บนแกน x และอยู่ในรูปของ $x=\pm\sqrt{b^2\left(1-\frac{y^2}{a^2}\right)}$ เมื่อแกนเอกอยู่บนแกน y

2.2.6 โปรเจคไทล์ (Projectile)

โปรเจคไทล์ คือ การเคลื่อนที่ของวัตถุภายใต้แรงโน้มถ่วงของโลก โดยที่ความเร็วในแนวราบมีค่าคงที่ (ไม่มีความเร่งในแนวราบ) ส่วนความเร็วในแนวดิ่งเปลี่ยนแปลงตลอดเวลา ซึ่ งเวลาที่ใช้ในการเคลื่อนที่ของทั้งแนวดิ่งและแนวราบนั้นจะใช้เวลาเท่ากัน แบ่งได้ 2 แบบคือ แบบจุดตกอยู่ต่ำกว่าจุดขว้าง และแบบจุดตกกับจุดขว้าง อยู่ระดับเดียวกัน ในที่นี้จะศึกษาเฉพาะแบบจุดตกกับจุดขว้าง อยู่ระดับเดียวกันดังนี้



รูปที่ 2.32 การเคลื่อนที่แบบโปรเจคไทล์ บนพิกัดฉาก



รูปที่ 2.33 การเคลื่อนที่แบบโปรเจคไทล์ บนพิกัดขนาน

สมการที่ใช้คำนวณมีดังนี้

$$t = \frac{2u\sin\theta}{g} \qquad S_x = \frac{u^2\sin 2\theta}{g} \qquad S_x = u\cos\theta \times t$$

$$\frac{S_y}{S_x} = \frac{1}{4}\tan\theta \qquad S_y = \frac{u^2\sin^2\theta}{2g} \qquad S_y = u\sin\theta \times t - gt^2$$

โดยที่ t คือ เวลาทั้งหมดที่ใช้ในการเคลื่อนที่

u คือ ความเร็วต้น

g คือ แรงดึงดูดของโลก

S_x คือ ระยะในการเคลื่อนที่ในแนวราบ

S_y คือ ระยะในการเคลื่อนที่ในแนวดิ่ง

ในส่วนของโปรเจคไทล์เมื่อนำมาวาดบนระบบพิกัดขนานนั้น จะใช้สมการเหมือนกับการวาดบนระบบพิกัดฉาก และมีวิธีการคิดเช่นเดียวกันดังนี้

1. ขั้นแรกหาเวลาทั้งหมดที่ใช้ในการเคลื่อนที่ก่อนจากสมการ
2. เมื่อได้เวลาแล้วจึงนำมาหา S_x และ S_y
3. เมื่อได้ค่าของ t, S_x และ S_y ครบแล้วจึงนำมา plot โดยใช้เวลาเป็นแกนหลัก

2.2.7 สมการการเคลื่อนที่ของเครื่องบิน

เครื่องบินเป็นวัตถุที่เคลื่อนที่อยู่บนระนาบแบบ 3 มิติ ซึ่งจะแตกต่างจากวัตถุทั่วไป ตัวอย่างเช่น รถจะเคลื่อนที่อยู่ในระนาบ 2 มิติ คือ ระนาบ x และ y เท่านั้นเนื่องจากการเคลื่อนที่ของเครื่องบินจะมีระนาบของความสูงจากพื้นดินเข้ามาเกี่ยวข้อง จึงทำให้เป็นการเคลื่อนที่บนระนาบ 3 มิตินั่นเอง และในขณะที่มีการเคลื่อนที่ก็จะมีปัจจัยที่เข้ามาเกี่ยวข้องอีก 2 ตัว คือ ความเร็ว และเวลา ดังนั้นการเคลื่อนที่ของเครื่องบินจึงเป็น 5 มิติ

แต่ในส่วนที่นำมาใช้พัฒนาโปรแกรมจำลองการเคลื่อนที่ชนกันของเครื่องบิน 2 ลำนั้น จะนำมาใช้เฉพาะในส่วนของความเร็ว เวลา ระนาบ x และระนาบ y โดยที่จะให้ระนาบ z ที่เป็นความสูงนั้นคงที่

สมการที่ใช้ในการเคลื่อนที่ของเครื่องบินคือ $s = vt$

โดยที่ s คือ ระยะทางที่เครื่องบินเคลื่อนที่ได้

v คือ ความเร็วของเครื่องบิน

t คือ เวลาในการเคลื่อนที่

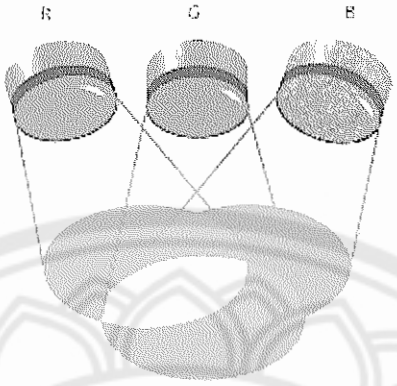
โดยเครื่องบินทั้ง 2 ลำ จะใช้เวลา t ร่วมกัน แล้วจึงนำไปเพื่อหาระยะทางในระนาบ x และระยะทางในระนาบ y เมื่อคำนวณเวลา ความเร็ว และระยะทางของเคลื่อนบินทั้งสองออกมา

ผลที่ได้สามารถแสดงได้บนระบบพิกัดขนาน ซึ่งจะไม่สามารถแสดงได้บนระบบพิกัดฉาก เนื่องจากว่าต้องประกอบไปด้วยตัวแปรถึง 9 ตัว ดังนี้คือ

- เวลาในการเคลื่อนที่
- ความเร็วของเคลื่อนบินลำที่ 1
- ความเร็วของเคลื่อนบินลำที่ 2
- ระยะทางบนระนาบ x ของเคลื่อนบินลำที่ 1
- ระยะทางบนระนาบ x ของเคลื่อนบินลำที่ 2
- ระยะทางบนระนาบ y ของเคลื่อนบินลำที่ 1
- ระยะทางบนระนาบ y ของเคลื่อนบินลำที่ 2
- ระยะทางบนระนาบ z ของเคลื่อนบินลำที่ 1
- ระยะทางบนระนาบ z ของเคลื่อนบินลำที่ 2

ซึ่งเมื่อนำมาวาดเป็นรูปบนพิกัดขนาน ก็จะพบว่าเครื่องบินทั้งสองลำจะชนกันหรือไม่ ถ้าชนจะชนกัน ณ เวลาใด

2.3 หลักการของสีแบบ RGB



รูปที่ 2.34 สีแบบ RGB [5]

การผสมสีบนคอมพิวเตอร์นั้นอาศัยแม่สี 3 สีคือ สีแดง สีเขียว และ น้ำเงิน หรือที่เรา เรียกว่า RGB Colors (Red-Green-Blue) ความเหมือนจริงของสีคอมพิวเตอร์นั้น ขึ้นอยู่กับในหนึ่งจุด (pixel) ของการแสดงผลนั้นใช้ระดับของสี หรือ color depths ว่ามีค่าเท่าไหร่ เช่นถ้าสี RGB มี color depths เป็น 8 planes นั่นคือ เราใช้ 8 บิตเก็บข้อมูลหนึ่งสี หมายความว่า เฉพาะแม่สี เช่น สีแดง สีเดียวก็จะมีความเป็นแดง อยู่ถึง $2^8 = 256$ ระดับ เมื่อเราผสมสี หนึ่งสี จาก แดง-เขียว-น้ำเงิน (RGB) เราต้องใช้สีแดง กี่ส่วนจาก 0 ถึง 255 ส่วน ใช้เขียว กี่ส่วนจาก 0 ถึง 255 และเช่นเดียวกัน สีน้ำเงินกี่ส่วน จาก 0 ถึง 255 สีที่เกิดขึ้นก็จะเกิดจากการผสมของสีทั้งสาม ในอัตราส่วนต่างๆ กัน ตัวอย่าง สีเหลืองธรรมดา เกิดจากการผสมสี แดง 255 ส่วน สีเขียว 255 ส่วน และสีน้ำเงิน 0 ส่วน ซึ่งระดับของสีแดง มีถึง 256 ระดับ สีเขียว 256 ระดับ สีน้ำเงิน 256 ระดับ ดังนั้น RGB ทั้งหมดใช้ 24 บิต (8+8+8) ในการแสดงสี RGB ของหนึ่งจุด (pixel) ซึ่งสามารถแสดงสีได้มากถึง 256x256x256 = 16.7 ล้านสี ตารางข้างล่างแสดงถึง จำนวนสีที่เป็นไปได้ ซึ่งขึ้นกับจำนวนบิต ที่ใช้กำหนดระดับของสี

ตารางที่ 2.1 แสดงจำนวนสีที่เป็นไปได้ซึ่งขึ้นกับจำนวนบิต

จำนวนบิตที่ใช้เก็บสีต่อหนึ่งจุด	Color Mode Name	จำนวนสีที่แสดงได้
1	Black and White	2
4	16-Color (EGA)	16
8	Pseudo Color	256
16	Hi-Color	65,536
24	True Color	16,777,216

2.4 หลักการของสีแบบ HSI

โหมดสีแบบ HSI เป็นโหมดสีอีกแบบหนึ่งที่ใช้ผสมสีบนคอมพิวเตอร์ โดยที่

H (Hue) คือ ค่าสี

S (Saturation) คือ ค่าความบริสุทธิ์ของสี

I (Intensity) คือ ความเข้มแสง

โดยที่แต่ละค่านั้นจะหาได้โดยต้องคำนวณมาจากโหมดสีแบบ RGB ซึ่งเป็นการแปลงค่าจากโหมดสีแบบ RGB มาสู่โหมดสีแบบ HSI โดยมีวิธีการคำนวณดังนี้

$$H = \begin{cases} \theta & if \quad B \geq G \\ 360 - \theta & if \quad B < G \end{cases}$$

โดยที่ $\theta = \cos^{-1}\left\{ \dfrac{\frac{1}{2}[(R-G)+(R-B)]}{[(R-B)^2+(R-G)(G-B)]^{\frac{1}{2}}} \right\}$

ซึ่ง θ อ้างอิงจากแกน R

$$S = 1 - \frac{3}{R+G+B}[\min(R,G,B)]$$

$$I = \frac{1}{3}(R+G+B)$$

ซึ่งค่าของ RGB ที่นำมาใช้จะต้องแปลงให้อยู่ในช่วง [0, 1] ก่อนจึงจะนำมาใช้คำนวณได้

บทที่ 3

วิธีการดำเนินการ

ในบทนี้จะเป็นส่วนที่อธิบายถึงวิธีการดำเนินงาน โดยที่จะเริ่มจากการศึกษาข้อมูลที่นำมาใช้ในการเขียนโปรแกรม อธิบายหลักการและขั้นตอนการทำงานของโปรแกรม และการออกแบบโปรแกรมทั้ง 2 โปรแกรม ซึ่งเขียนด้วยภาษา C++ โดยใช้ Microsoft Visual Studio 6 และ Microsoft Visual Studio 2005

3.1 การศึกษาข้อมูล

ข้อมูลที่ทำการศึกษาแบ่งเป็น 2 ส่วนดังนี้

3.1.1 การใช้ MFC Programming

ในส่วนของการศึกษาการใช้ MFC เราต้องเรียนรู้ตามองค์ประกอบของโปรแกรมที่ได้ทำการออกแบบไว้ ซึ่งได้ทำการออกแบบไว้ให้มีส่วนประกอบดังนี้ หน้าต่างหลักของโปรแกรมกำหนดให้มีปุ่มกดเพื่อเลือกหัวข้อที่ต้องการให้โปรแกรมวาดรูป เมื่อผู้ใช้กดปุ่มแล้วให้โปรแกรมแสดงหน้าต่างย่อยขึ้นมาเพื่อกรอกข้อมูลของสมการต่างๆ แล้วกดตกลง จากนั้นจะให้โปรแกรมทำการคำนวณ และแสดงรูปออกมา จะเห็นได้ว่าส่วนแรกที่ต้องทำการศึกษาคือ การสร้าง Dialog based เพื่อทำเป็นหน้าต่างหลัก จากนั้นศึกษาการสร้างปุ่มกด การกำหนดตัวแปร การผูกติดตัวแปรแบบ DDX เพื่อนำค่าตัวแปรไปใช้สร้างเงื่อนไขการตอบสนองปุ่มกดต่างๆ ขั้นตอนต่อไปจะเป็นส่วนที่สำคัญมากส่วนหนึ่งคือ การสร้าง Header File เพื่อใช้ในการคำนวณค่าของสมการแต่ละอย่าง ที่จำเป็นต้องสร้าง Header File ก็เพื่อสะดวกในการปรับปรุงและแก้ไขในส่วนของการคำนวณรายละเอียดต่าง โดยไม่ต้องเข้าไปวุ่นวายในส่วนของ MFC และเมื่อทำการสร้าง Header File เสร็จก็ใช้เรียกใน MFC โดยการ Include เมื่อต้องการใช้ให้สร้าง Object ขึ้นมาจากนั้นส่งค่าเพื่อทำการคำนวณและอาจสร้างตัวแปรเพื่อรับค่ามาใช้ ต่อไปก็นำค่าที่ได้จากการคำนวณมาวาดโดยใช้เทคนิค GDI วาดภาพในส่วนของ Onpaint ที่ MFC ได้เตรียมไว้ให้เรียบร้อยแล้ว จากนั้นทำการวาดด้วยคำสั่ง MoveTo และ LineTo

3.1.2 การใช้ Class CImage เบื้องต้น

การใช้ Class CImage เป็นส่วนสำคัญมากอีกส่วนหนึ่งเนื่องจากต้องใช้เขียนโปรแกรมที่ 2 ที่ใช้แสดง Parallel Coordinate ของรูปภาพ เนื่องจากต้องมีการเข้าถึงแต่ละ pixel ของภาพแล้วเก็บค่าของสีต่างๆ แล้วนำมาแสดงเป็นรูปบนแกนคู่ขนาน

อันดับแรกต้องทำการสร้าง object ของ class CImage ขึ้นมาก่อนเพื่อนำรูปที่เลือกมาเก็บไว้โดยใช้คำสั่ง CImage image; จากนั้นเมื่อได้ตำแหน่งที่เก็บภาพก็จะทำการเรียกภาพนั้นเข้ามาเก็บไว้ใน object ที่สร้างไว้ โดยใช้คำสั่ง image.Load("ตำแหน่งที่เก็บรูปภาพ");

อันดับต่อมาเป็นหลักการเข้าถึงค่าสี RGB ของแต่ละ pixel เป็นดังนี้ ขั้นแรกสร้าง pointer ชื่อ pSrc ซึ่งเป็นแบบ BYTE โดยใช้คำสั่ง BYTE *pSrc; จากนั้นใช้คำสั่ง pSrc = (BYTE*)image.GetPixelAddress(x, y); ซึ่ง x, y คือตำแหน่งของ pixel ที่จะนำค่า RGB ออกมา จากนั้นเมื่อเราให้ pSrc เข้าไปชี้ตำแหน่งที่ต้องการแล้วเราจะนำค่า RGB ออกมาซึ่งโครงสร้างของการเก็บสีเป็นดังนี้ สีแดงจะอยู่ที่ตำแหน่ง pSrc[2] สีเขียวจะอยู่ที่ตำแหน่ง pSrc[1] สีน้ำเงินจะอยู่ที่ตำแหน่ง pSrc[0] จากนั้นก็สร้าง array ไว้เก็บค่าสีต่างๆ แล้วนำไปวาดในส่วนของ MFC Programming

3.2 หลักการและขั้นตอนการทำงานของโปรแกรม

หลักการของโปรแกรมนี้คือ แยกส่วนของการคำนวณหาค่าจากสมการต่างๆ ออกจากันเพื่อลดปัญหาเรื่องการเกิด Error โดยจะทำการสร้างเป็น Header File หรือ แยกเป็น Function

1. ขั้นตอนการรับค่าจากปุ่มกดของหน้าต่างหลัก
2. ขั้นตอนแสดงหน้าต่างรองเพื่อรับค่า
3. ขั้นตอนการส่งค่าไปให้ Function และรับค่าคืน
4. ขั้นตอนการวาดรูป

3.2.1 ขั้นตอนการรับค่าจากปุ่มกดของหน้าต่างหลัก

เริ่มจากการแสดงหน้าต่างหลักของโปรแกรมซึ่งจะมีปุ่มให้ผู้ใช้เลือกหัวข้อที่ต้องการให้โปรแกรมวาดรูป โดยโปรแกรมจะรับค่าที่ผู้ใช้ใส่จากนั้นนำไปตรวจสอบเพื่อเรียกหน้าต่างรองให้แสดงขึ้นมาให้ผู้ใช้ใส่ค่าต่างๆ ของสมการที่ต้องการให้วาดรูปลงไป

3.2.2 ขั้นตอนแสดงหน้าต่างรองเพื่อรับค่า

หลังจากที่ได้รับค่าการกดปุ่มจากผู้ใช้แล้วก็นำค่าที่ได้ไปตรวจสอบเงื่อนไขว่าจะต้องแสดงหน้าต่างรองใดออกมาให้ผู้ใช้ ในส่วนของหน้าต่างรองนี้จะแสดงรายละเอียดไม่เหมือนกัน ซึ่งขึ้นอยู่กับหัวข้อที่ผู้ใช้เลือกซึ่งจะมี Line, Circle, Parabola, Hyperbola, Ellipse, Projectile, Cube and Sphere และ Plane

3.2.3 ขั้นตอนการส่งค่าไปให้ Fuuction และรับค่าคืน

หลังจากที่ผู้ใช้ได้ทำการป้อนค่าต่างๆ ตามสมการที่ต้องการให้โปรแกรมทำการวาดรูปแล้วนั้น โปรแกรมจะทำการรับค้าที่ได้มาแล้วส่งไปให้ส่วนของ header คำนวณ เมื่อ header คำนวณเสร็จก็จะส่ง

ค่าคืนมาให้โดยที่จะต้องสร้างตัวแปลไว้รอรับค่าด้วย แต่บางสมการไม่ได้สร้าง header ไว้ ก็ส่งค่าให้ฟังก์ชั่นคำนวณได้ทันที

3.2.4 ขั้นตอนการวาดรูป

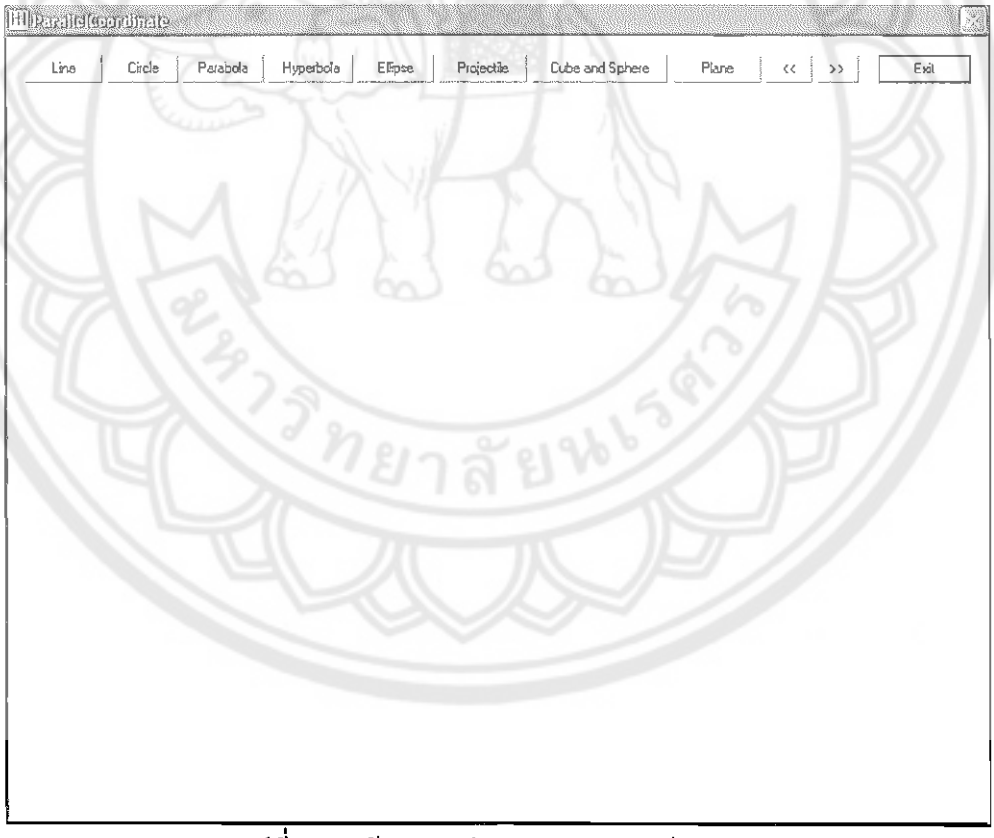
เมื่อได้รับค่าจากส่วนของ header แล้ว หรือได้รับค่าที่คำนวณจากฟังก์ชั่นแล้วก็นำค่าที่ได้ไปวาดรูปโดยใช้การวนลูปแล้ววาดโดยใช้คำสั่ง MoveTo และ LineTo

3.3 การออกแบบส่วนประกอบของโปรแกรมที่ 1

ต่อไปเป็นส่วนของการออกแบบส่วนประกอบของโปรแกรมโดยจะมี 2 ส่วนใหญ่ๆ ดังนี้

1. ส่วนของหน้าต่างหลัก
2. ส่วนของหน้าต่างรองเพื่อรับค่าต่างๆ ของสมการ

3.3.1 ส่วนของหน้าต่างหลัก



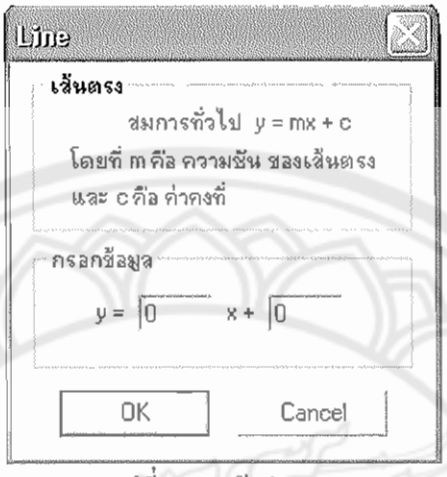
รูปที่ 3.1 หน้าตาของโปรแกรมส่วนหน้าต่างหลัก

ในส่วนนี้จะรับค่าที่ผู้ใช้เลือกแล้วนำค่าปุ่มกดที่ได้ไปตรวจสอบเงื่อนไขว่าตรงกับปุ่มกดใดแล้วจะต้องแสดงหน้าต่างรองนั้นออกมา

3.3.2 ส่วนของหน้าต่างรองเพื่อรับค่าต่างๆ ของสมการ

ในส่วนนี้จะมีทั้งหมด 8 หน้าต่าง เพื่อรองรับการป้อนสมการที่แตกต่างกัน ดังนี้

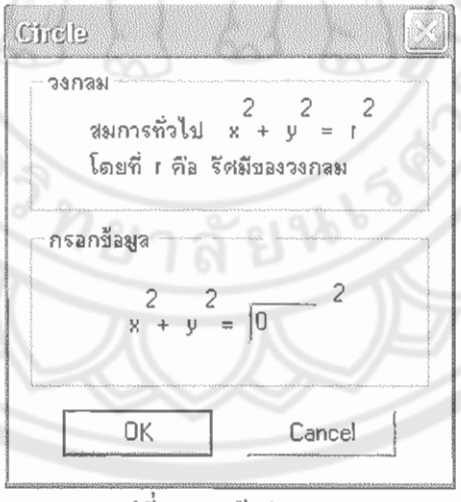
1. Line



รูปที่ 3.2 หน้าต่าง Line

ช่องแรกจะรับค่า ความชัน ช่องที่ 2 รับค่า ค่าคงที่ตามสมการ

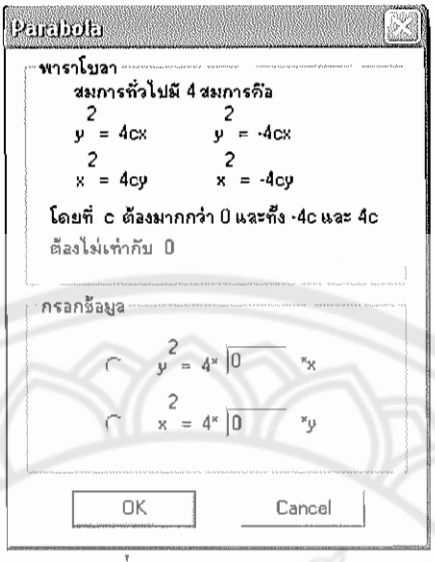
2. Circle



รูปที่ 3.3 หน้าต่าง Circle

ช่องนี้จะรับค่ารัศมีของวงกลม

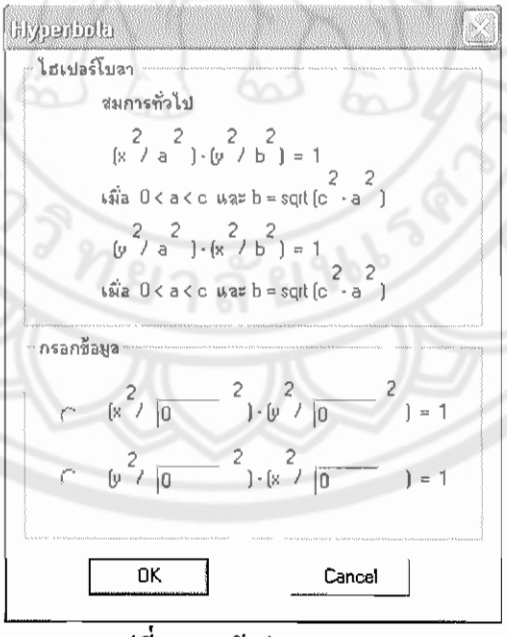
3. Parabola



รูปที่ 3.4 หน้าต่าง Parabola

จะมีให้เลือก 2 หัวข้อคือแบบอยู่บนแกน x และ y ค่าที่ใส่คือค่า c

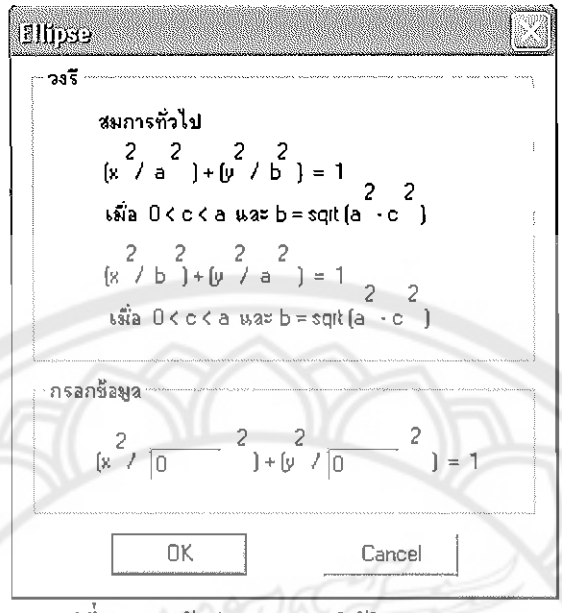
4. Hyperbola



รูปที่ 3.5 หน้าต่าง Hyperbola

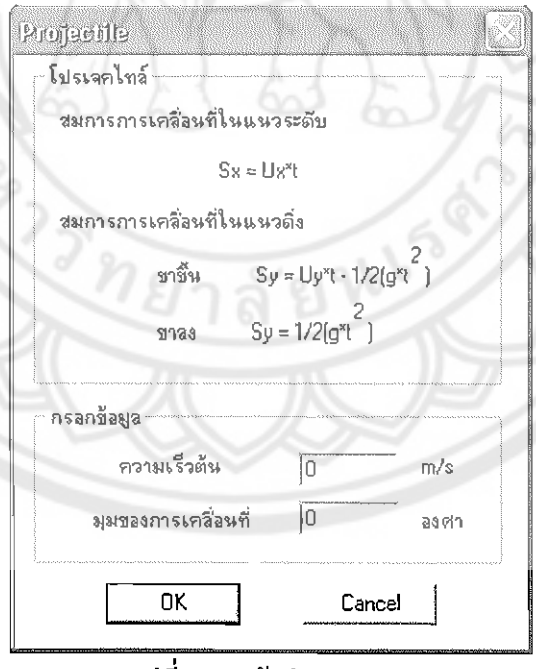
มีให้เลือก 2 หัวข้อและให้ใส่ค่า a และ b

5. Ellipse



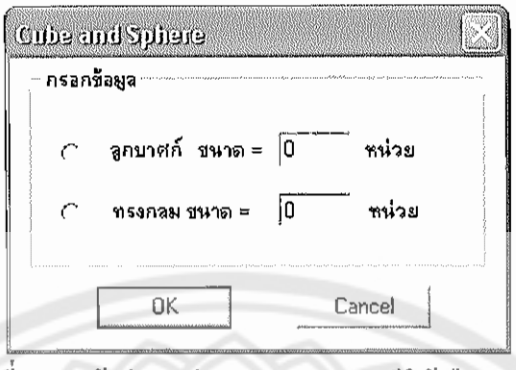
รูปที่ 3.6 หน้าต่าง Ellipse ให้ใส่ค่า a และ b

6. Projectile



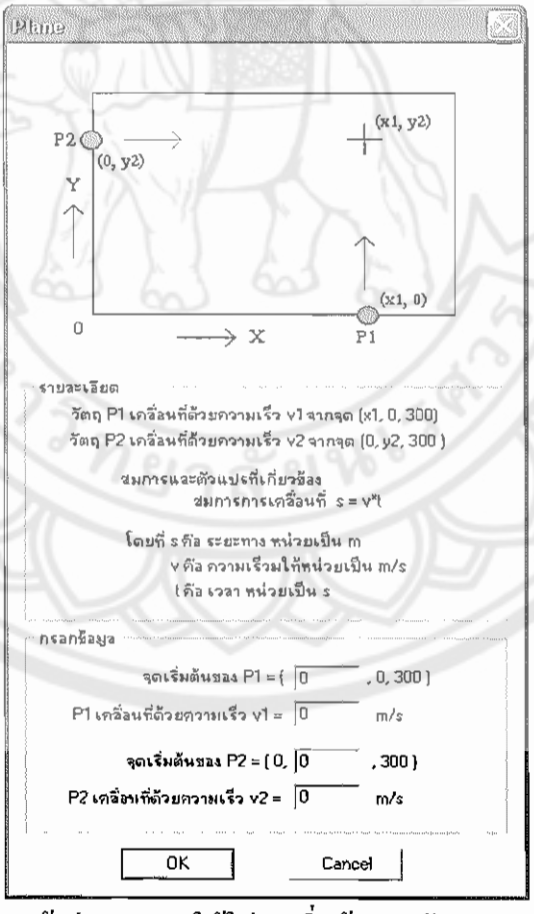
รูปที่ 3.7 หน้าต่าง Projectile
ให้ใส่ค่าความเร็วและมุมของการเคลื่อนที่

7. Cube and Sphere



รูปที่ 3.8 หน้าต่าง กล่องและทรงกลม มีให้เลือก 2 ข้อ ให้ใส่ค่าขนาดของกล่องหรือรัศมีทรงกลม

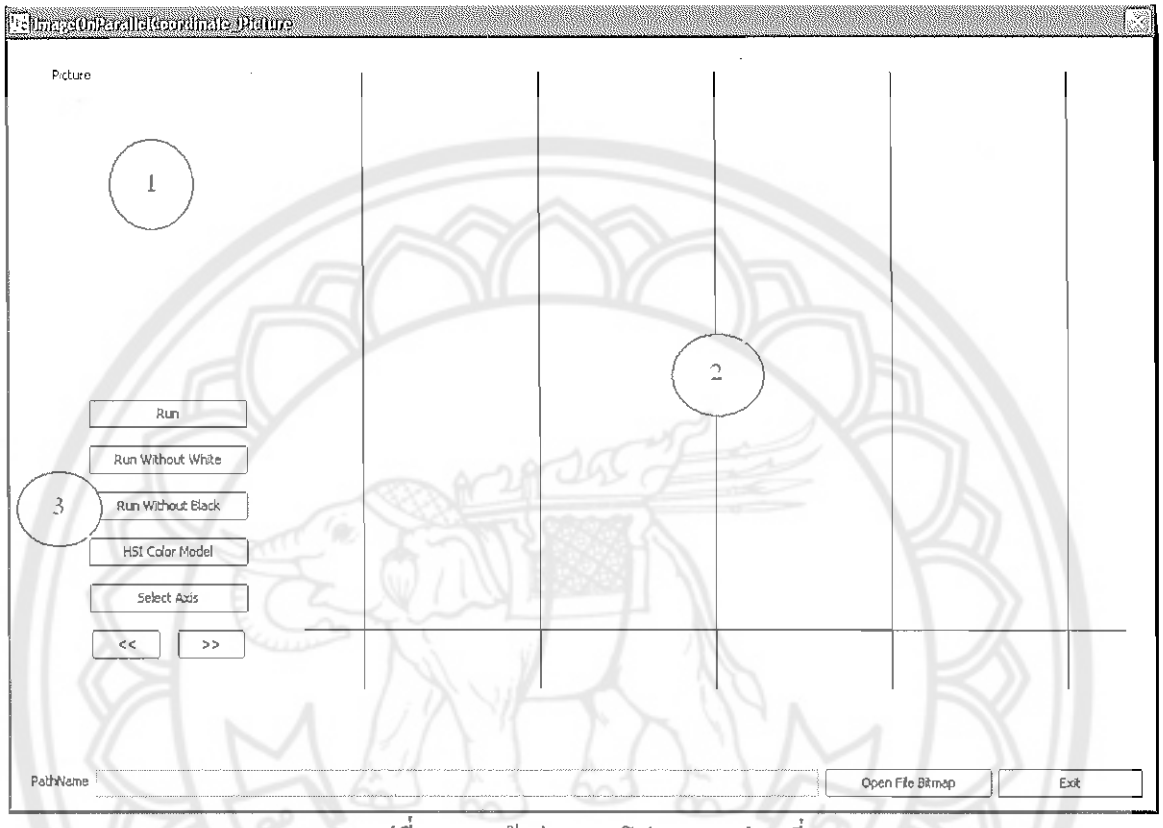
8. Plane



รูปที่ 3.9 หน้าต่าง Plane ให้ใส่ จุดเริ่มต้นของวัตถุ 1 และความเร็ว ให้ใส่จุดเริ่มต้นของวัตถุ 2 และความเร็ว

3.4 การออกแบบส่วนประกอบของโปรแกรมที่ 2

ส่วนที่ 2 เป็นส่วนของโปรแกรมที่ใช้ระบบพิกัดขนาน ในการประมวลผลรูปภาพ หน้าตาของโปรแกรมมีดังนี้



รูปที่ 3.10 หน้าต่างของโปรแกรมส่วนที่ 2

หมายเลข 1 คือส่วนที่ใช้แสดงรูปภาพที่เราต้องการ

หมายเลข 2 คือส่วนที่ใช้แสดงรูปวาดบนแกนพิกัดคู่ขนาน

หมายเลข 3 คือส่วนควบคุมการแสดงผลของโปรแกรมซึ่งมีรายระเอียดดังนี้

ปุ่ม Open File Bitmap ใช้เปิดหารูปที่ต้องการ

ส่วนนี้เป็นส่วนแสดงของตำแหน่งรูปที่เปิดขึ้นมา

PathName

ปุ่ม Run เมื่อต้องการวาดรูปบนแกนพิกัดคู่ขนานในโหมดสี RGB

ปุ่ม Run Without White เมื่อต้องการวาดรูปบนแกนพิกัดคู่ขนาน แบบไม่วาดสีขาว

ปุ่ม Run Without Black เมื่อต้องการวาดรูปบนแกนพิกัดคู่ขนาน แบบไม่วาดสีดำ

ปุ่ม HSI Color Model เมื่อต้องการแยกสีในโหมดสีแบบ HSI

ปุ่ม Select Axis และ << >> เมื่อต้องการเลื่อนแกนต่างๆ

ปุ่ม Exit เพื่อออกจากโปรแกรม

3.5 การเขียนโปรแกรม

โครงงานนี้ใช้ภาษา C++ ของ Microsoft Visual Studio 6 และ Microsoft Visual Studio 2005 ซึ่งในส่วนของโปรแกรมที่ 1 จะใช้ Microsoft Visual Studio 6 และในส่วนของโปรแกรมที่ 2 นั้นจะใช้ Microsoft Visual Studio 2005

3.5.1 โปรแกรมที่ 1 ส่วนของหน้าต่างหลัก

ใช้ MFC Programming ของ Microsoft Visual Studio 6 โดยสร้างเป็น Dialog based จากนั้นสร้างปุ่มกดต่างๆ แล้วเขียนการตอบรับปุ่มกดแต่ละปุ่มเพื่อให้แต่ละปุ่มมีความสามารถเรียกหน้าต่างรองของตัวเองได้

3.5.2 เทคนิคการรับค่าและส่งค่า

สร้าง Struct ชื่อว่า GRAPHDATA ไว้ในส่วนของ ParallelCoordinateDlg.h เพื่อเป็นตัวแปลกลางให้รับส่งค่าจากหน้าต่างรองมาส่งให้ฟังก์ชั่นต่างๆ ตามที่ได้เลือกไป โดยที่แต่ละปุ่มกดจะมีตัวแปลเป็นของตัวเองที่ชื่อ isdatavalid ตั้งแต่ 1 ถึง 8 ตัวแปลนี้เป็นแบบ bool ซึ่งเมื่อกดแต่ละปุ่มค่าก็จะกลายเป็น true ทันทีจากนั้นก็จะเรียก Invalidate(); เพื่อให้โปรแกรม refresh หน้าต่างหลักจากนั้นก็จะแสดงรูปออกมา

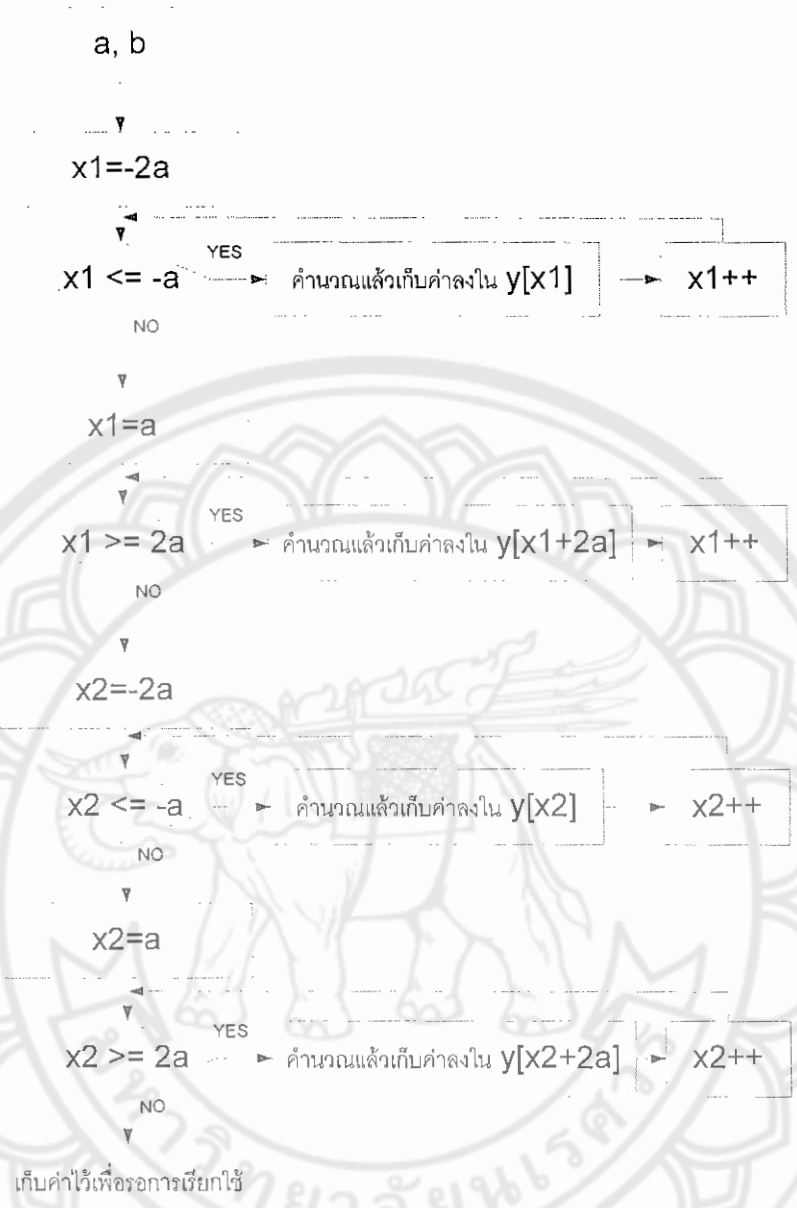
3.5.3 โปรแกรมที่ 2 การใช้ระบบพิกัดขนาน กับ รูปภาพ

ใช้ MFC Programming ของ Microsoft Visual Studio 2005 โดยสร้างเป็น Dialog based เช่นกัน โปรแกรมนี้ไม่มีหน้าต่างรองทุกอย่างจะอยู่บนหน้าต่างหลักทั้งหมด เริ่มจากการสร้างปุ่มกดเพื่อหารูปภาพที่ต้องการนำมาวาด โดยต้องสร้าง Object ของ Class CFileDialog แล้วใช้คำสั่งดังนี้ CFileDialog dlg(true,NULL,NULL,OFN_HIDEREADONLY | OFN_OVERWRITEPROMPT,file); ซึ่ง file คือ Object ของ Class CString ที่ประกาศไว้ดังนี้ CString file; file="Image(*.bmp)|*.bmp||"; ซึ่งเมื่อกดปุ่มก็จะได้หน้าต่างค้นหาที่เลือกได้เฉพาะไฟล์รูปภาพนามสกุล .bmp เท่านั้น

การทำงานของโปรแกรมที่ 1 จะมี Class ย่อย อยู่ 2 Class คือ calprojectile.h และ calhyperbola.h ซึ่งทั้งสอง มีการทำงานดังรูปที่ 3.11 และ 3.12 ตามลำดับ

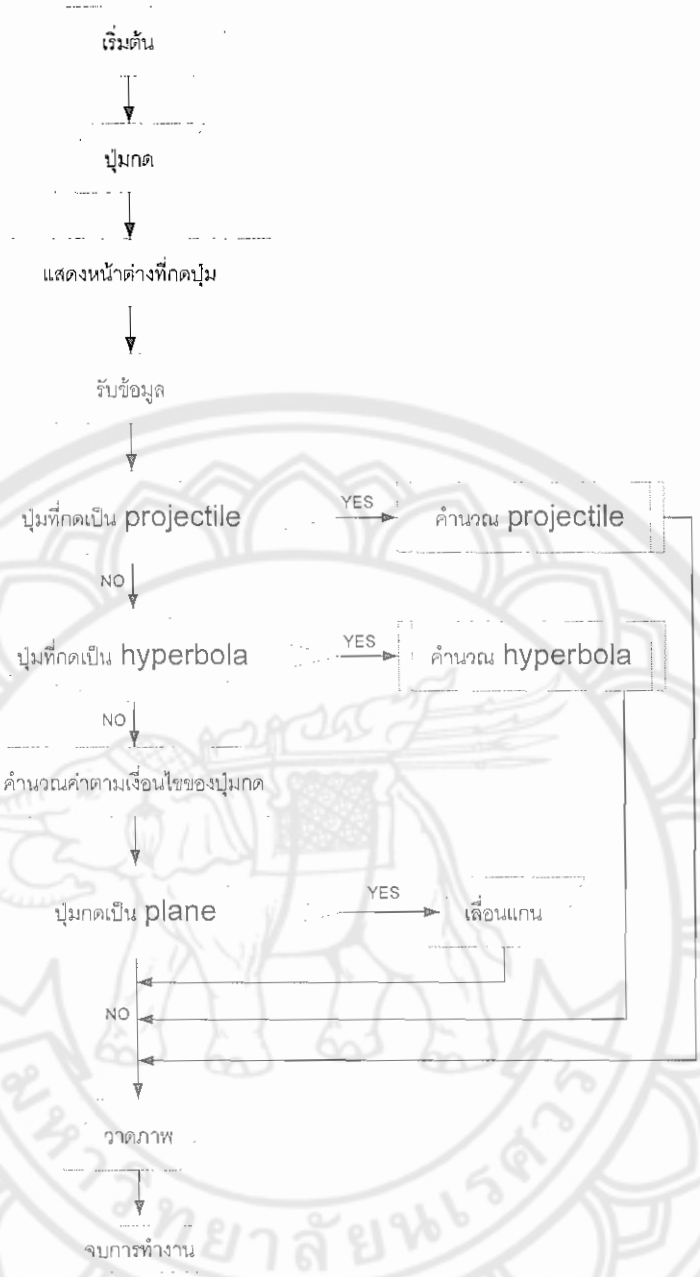
รูปที่ 3.11 การทำงานของ Class calprojectile.h

รับค่า u (ความเร็วต้น) และ degree (มุมในการเคลื่อนที่) จากนั้นคำนวณหาค่า t (เวลาในการเคลื่อนที่เป็นจำนวนเต็ม) และ tt (เวลาในการเคลื่อนที่จริง) จากนั้นหาค่าระยะสูงสุด หาค่าระยะไกลสุด จากนั้นหาค่าระยะทางตามแนวแกน y ทุกวินาที โดยวนลูป x ให้ x = 0 แล้วเพิ่ม x ไปเรื่อยๆ จน x = t จึงหยุด จากนั้นเก็บค่าลงใน y[x] หลังจากนั้นหาค่าระยะทางตามแนวแกน x ทุกวินาที โดยวนลูป x ให้ x = 0 แล้วเพิ่ม x ไปเรื่อยๆ จน x = t จึงหยุด จากนั้นเก็บค่าลงใน x[x] จากนั้นรอการเรียกใช้ค่าตัวแปร



รูปที่ 3.12 การทำงานของ Class calhyperbola.h

รับค่า a (ค่า a ของรูปที่ต้องการ) และ b (ค่า b ของรูปที่ต้องการ) เข้ามาจากโปรแกรมหลัก แล้วให้ x1 = -2a ทำการวนลูปเพื่อคำนวณหาค่า ในช่วงของ -2a < x1 < -a แล้วเก็บค่าไว้ใน y[x1] เสร็จแล้วก็กำหนด x1 ใหม่ ให้เท่ากับ a แล้วการวนลูปเพื่อคำนวณหาค่า ในช่วงของ a < x1 < 2a แล้วเก็บค่าไว้ใน y[x1+2a] หลังจากนั้น กำหนดค่า x2 = -2a ทำการวนลูปเพื่อคำนวณหาค่าเช่นกัน ในช่วงของ -2a < x2 < -a แล้วเก็บค่าไว้ใน y[x2] เสร็จแล้วก็กำหนด x2 = a แล้วการวนลูปเพื่อคำนวณหาค่า ในช่วงของ a < x2 < 2a แล้วเก็บค่าไว้ใน y[x2+2a] เพื่อรอการเรียกใช้



รูปที่ 3.13 Flowchart โปรแกรมที่ 1

เริ่มต้นโดยรับค่าปุ่มกดเข้ามา จากนั้นแสดงหน้าต่างรองตามค่าของปุ่มกด เพื่อรับข้อมูล จากนั้นตรวจสอบว่าใช่ปุ่มกดโปรเจคไทล์หรือไม่ ถ้าใช่จะส่งค่าเข้าไปใน Class calprojectile.h เพื่อคำนวณค่าแล้วนำไปวาด ถ้าไม่ใช่ จะตรวจสอบว่าใช่ปุ่มกดไฮเปอร์โบลาหรือไม่ ถ้าใช่จะส่งค่าเข้าไปใน Class calhyperbolae.h เพื่อคำนวณค่า แล้วนำไปวาด ถ้าไม่ใช่ทั้ง 2 อย่าง จะคำนวณค่าตามเงื่อนไขของปุ่มกด จากนั้นตรวจสอบว่าเป็น plane หรือไม่ ถ้าใช้ ต้องเลือกแกนก่อน จึงจะวาดรูป

ในส่วนของโปรแกรมที่ 2 จะมี class ย่อย อยู่ 1 class คือ hsi.h ซึ่งมีการทำงานดังรูปที่ 3.1

รูปที่ 3.14 การทำงานของ Class hsi.h

รับค่า r, g, b (ค่าสีของแต่ละ pixel ของรูป) เข้ามา คำนวณค่า H (Hue) เก็บค่าไว้ใน ตัวแปร H แล้วคำนวณค่า S (Saturation) เก็บค่าไว้ใน ตัวแปร S และคำนวณค่า I (Intensity) เก็บค่าไว้ใน ตัวแปร I จากนั้นรอการเรียกใช้ค่า

รูปที่ 3.15 Flowchart โปรแกรมที่ 2

เริ่มต้น โหลดภาพขึ้นมา แล้วรับค่าปุ่มกด เก็บค่าสี RGB ไวในตัวแปร r, g, b แล้ววาดภาพ ตรวจสอบการเลือกแกน ถ้ามีให้วาดภาพใหม่ ถ้าไม่มีตรวจสอบการเลือกแกน ถ้ามีวาดรูปใหม่ ถ้าไม่มีสิ้นสุดโปรแกรม

บทที่ 4

ผลการทดลอง

ในบทนี้จะอธิบายการใช้โปรแกรมในแต่ละส่วนและผลการทดลองในแบบต่างๆ ซึ่งโปรแกรมมีอยู่สองโปรแกรมด้วยกันคือ โปรแกรมที่ 1 ใช้วาดรูปจากสมการต่างๆ และโปรแกรมที่ 2 ใช้ในการทำ Color Image Processing โดยสามารถแยกโหมดสีแบบ RGB และแบบ HSI

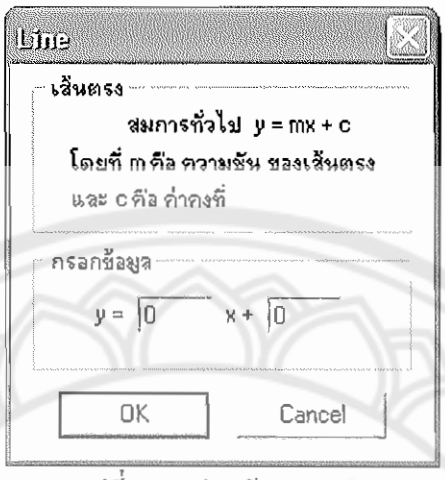
4.1 วิธีการใช้งานโปรแกรมที่ 1

โปรแกรมแรก คือ โปรแกรมวาดรูปจากสมการต่างๆ ซึ่งการใช้งานโปรแกรมมีดังนี้ เริ่มต้นโดยเลือกชนิดของรูปที่ต้องการจะวาด โดยจะมีให้เลือกทั้งหมด 9 ชนิดด้วยกัน ดังรูปที่ 4.1

รูปที่ 4.1 การเลือกชนิดของรูปที่ต้องการจะวาด

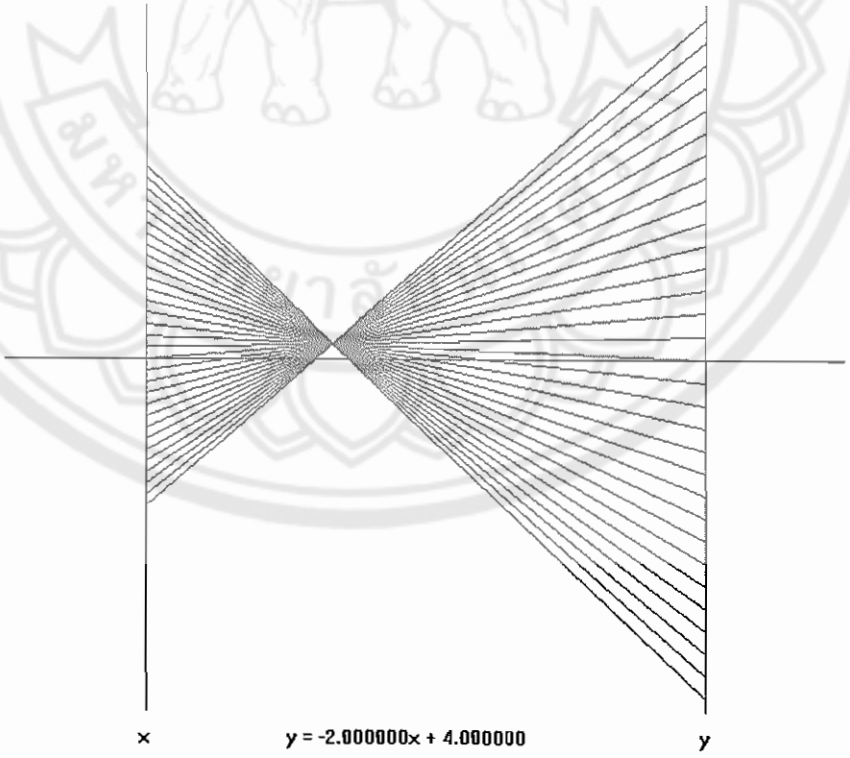
โดยที่การใช้งานของแต่ละส่วน มีวิธีการใช้งานดังนี้

4.1.1. เส้นตรง (Line) เลือกที่ Line จะมีกล่องข้อความ Line ขึ้นมาดังรูปที่ 4.2



รูปที่ 4.2 กล่องข้อความ Line

กรอกค่า m และ ค่า c ของสมการเส้นตรงที่ต้องการวาดลงไป แล้วกด OK จะได้รูปของสมการเส้นตรงบนพิกัดขนานออกมา ดังตัวอย่างรูปที่ 4.3

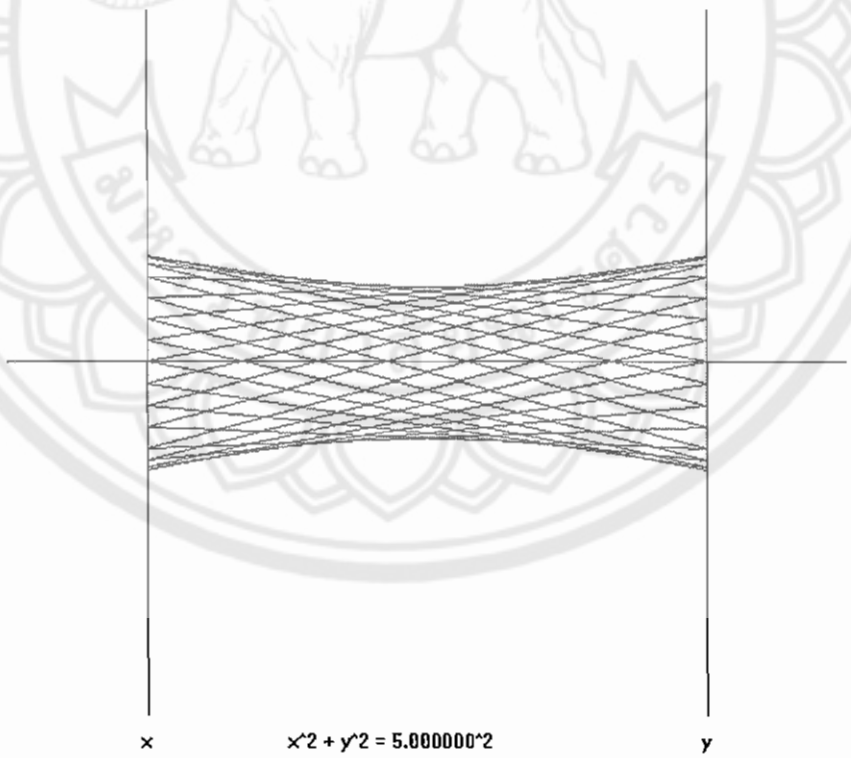


รูปที่ 4.3 รูปเส้นตรงบนพิกัดขาน โดย m = -2 และ c = 4

4.1.2. วงกลม (Circle) เลือกที่ Circle จะมีกล่องข้อความ Circle ขึ้นมา ดังรูปที่ 4.4

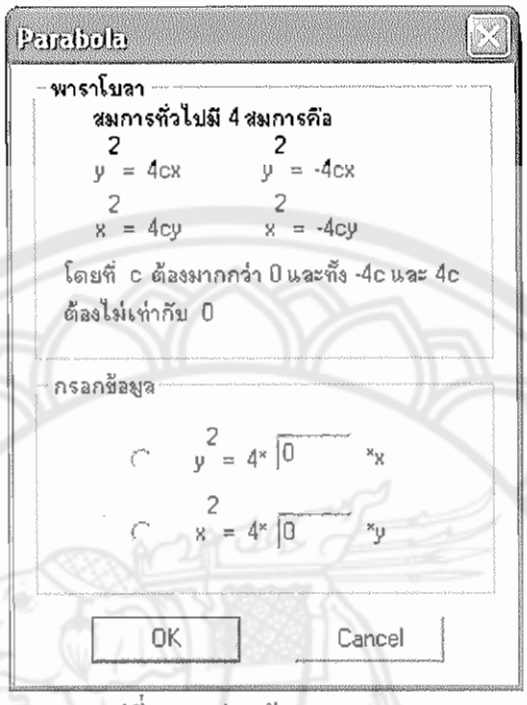
รูปที่ 4.4 กล่องข้อความ Circle

กรอกค่า r ของสมการวงกลมที่ต้องการวาดลงไป แล้วกด OK จะได้รูปของสมการวงกลมบนพิกัดขนานออกมา ดังตัวอย่างรูปที่ 4.5



รูปที่ 4.5 รูปวงกลมบนพิกัดขนาน โดยที่ r = 5

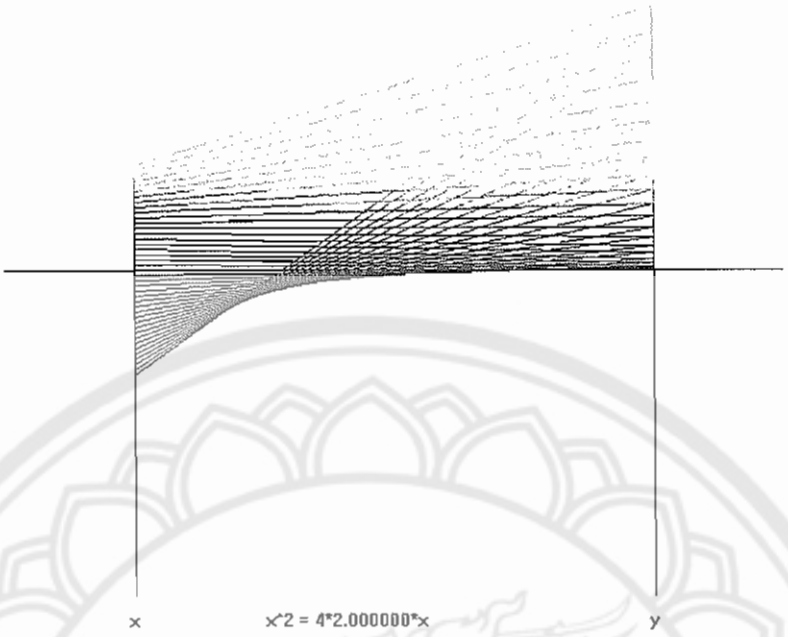
4.1.3. พาราโบลา (Parabola) เลือกที่ Parabola จะมีกล่องข้อความ Parabola ขึ้นมาดังรูปที่ 4.6



รูปที่ 4.6 กล่องข้อความ Parabola

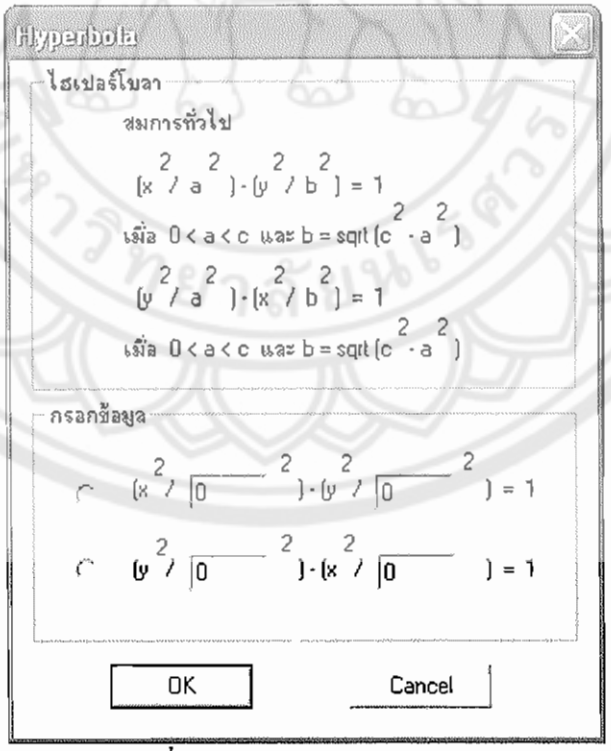
ขั้นแรก จะต้องเลือกสมการ ระหว่างสมการ $y^2 = 4cx$ เป็นพาราลาโบลาบนแกน x หรือ สมการ $x^2 = 4cy$ เป็นพาราโบลาบนแกน y

ขั้นที่สอง กรอกค่า c ของสมการพาราโบลาที่ต้องการลงไป แล้วกด OK จะได้รูปของสมการพาราโบลาบนพิกัดขนานออกมา ดังตัวอย่างรูปที่ 4.7



รูปที่ 4.7 รูปพาราโบลาบนพิกัดขนาน โดยเลือกสมการ $x^2 = 4cy$ และ $c = 2$

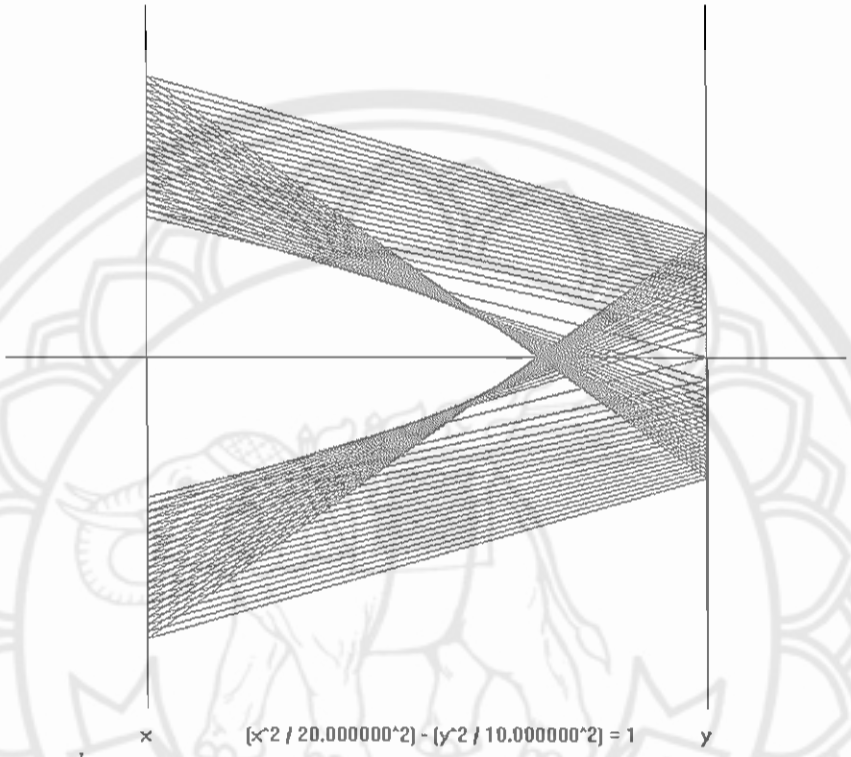
4.1.4. ไฮเปอร์โบลา (Hyperbola) เลือกที่ Hyperbola จะมีกล่องข้อความ Hyperbola ขึ้นมา ดังรูปที่ 4.8



รูปที่ 4.8 กล่องข้อความ Hyperbola

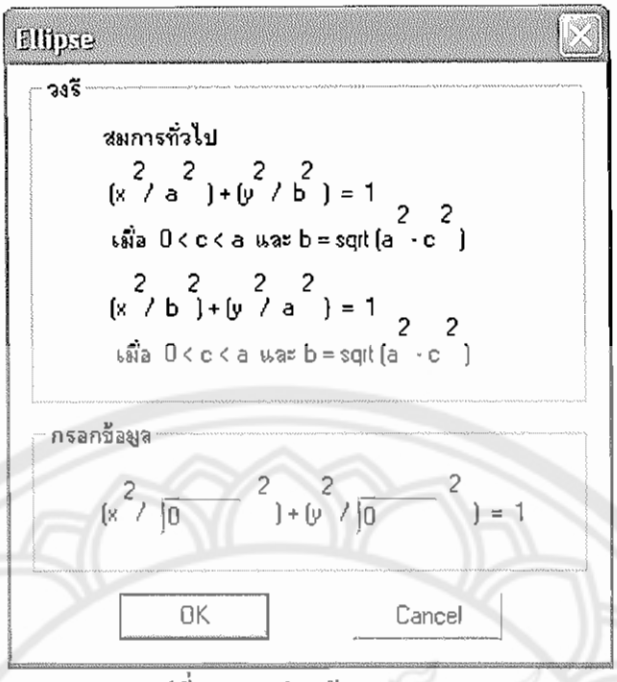
ขั้นแรก เลือกสมการ ระหว่างสมการ $x^2/a^2 - y^2/b^2 = 1$ เป็นไฮเปอร์โบลาบนแกน x หรือ สมการ $y^2/a^2 - x^2/b^2 = 1$ เป็นไฮเปอร์โบลาบนแกน y

ขั้นที่สอง กรอกค่า a และ b ของสมการไฮเปอร์โบลาที่ต้องการลงไป แล้วกด OK จะได้รูปของสมการไฮเปอร์โบลาบนพิกัดขนานออกมา ดังตัวอย่างรูปที่ 4.9



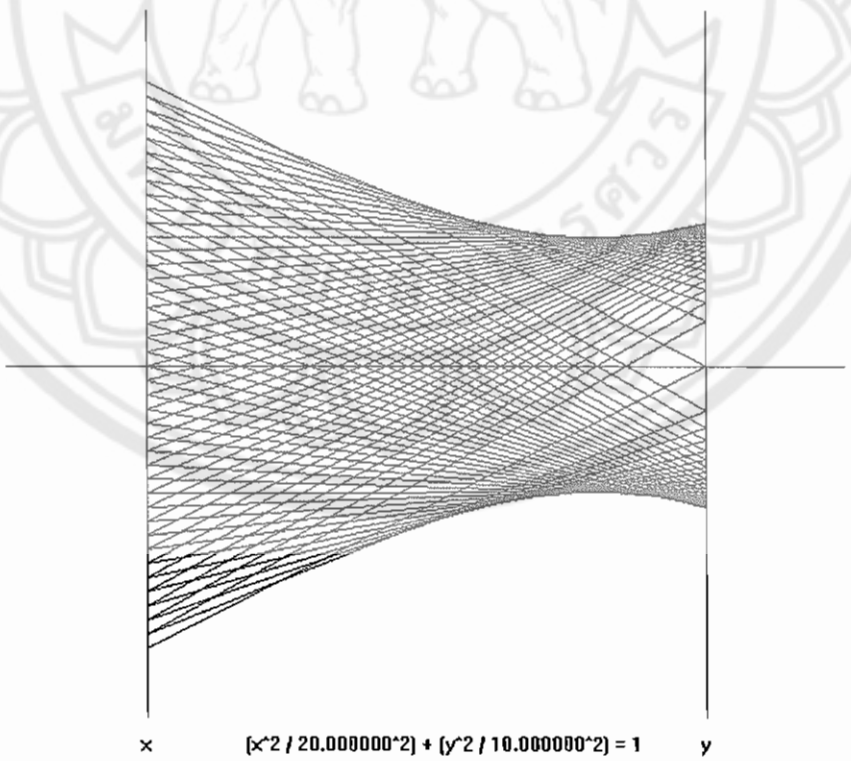
รูปที่ 4.9 รูปไฮเปอร์โบลาบนพิกัดขนาน โดยเลือกสมการ $x^2/a^2 - y^2/b^2 = 1$ และให้ค่า a = 20 ค่า b = 10

4.1.5. วงรี (Ellipse) เลือกที่ Ellipse จะมีกล่องข้อความ Ellipse ขึ้นมา ดังรูปที่ 4.10



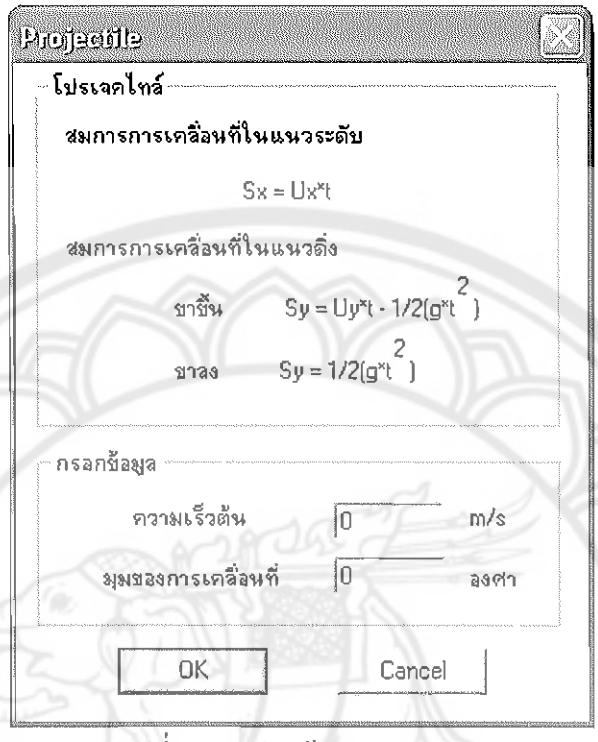
รูปที่ 4.10 กล่องข้อความ Ellipse

กรอกค่า a และ b ของสมการวงรีที่ต้องการลงไป ถ้า a > b แกนเอกจะอยู่บนแกน x แต่ถ้า a < b แกนเอกจะอยู่บนแกน y แล้วกด OK จะได้รูปของสมการวงรีบนพิกัดขนานออกมา ดังรูปที่ 4.11



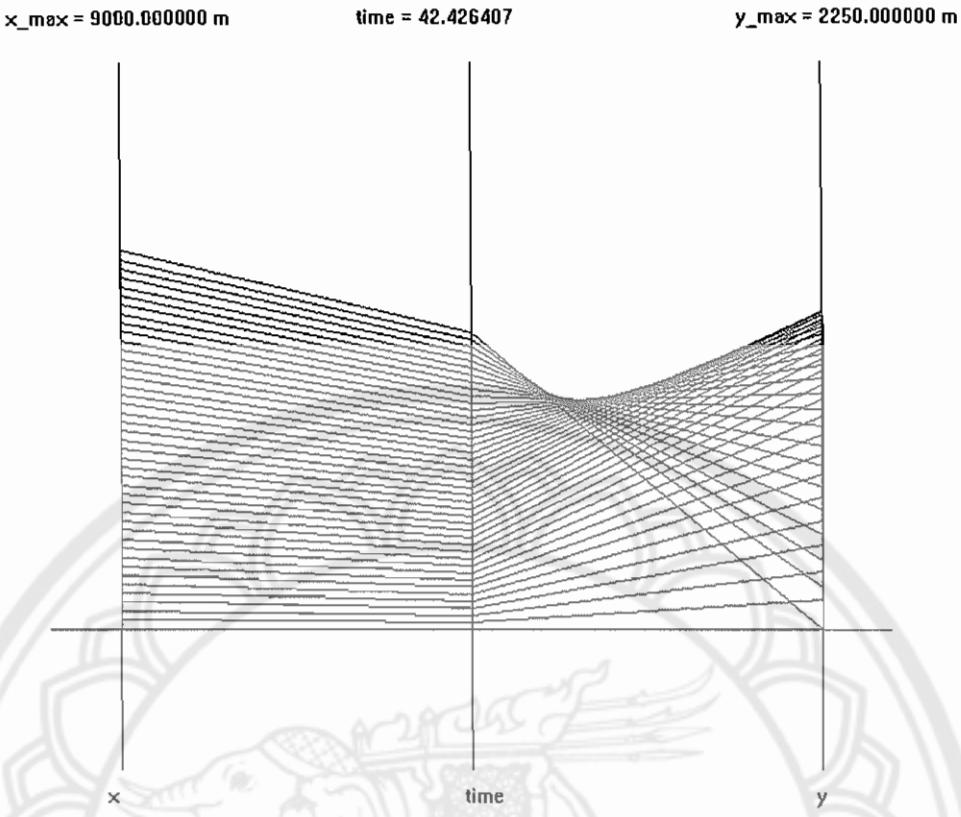
รูปที่ 4.11 รูปวงรีบนพิกัดขนาน โดยที่ให้ค่า a = 20 และ b = 10

4.1.6. โปรเจคไทล์ (Projectile) เลือกที่ Projectile จะมีกล่องข้อความ Projectile ขึ้นมา ดังรูปที่ 4.12



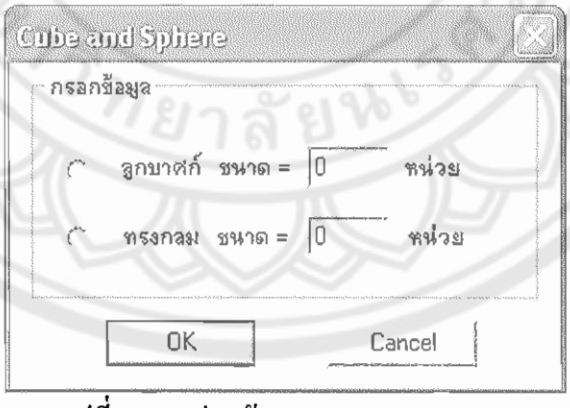
รูปที่ 4.12 กล่องข้อความ Projectile

กรอกค่าความเร็วต้นและมุมของการเคลื่อนที่ ของสมการโปรเจคไทล์ที่ต้องการลงไป แล้วกด OK จะได้รูปของสมการโปรเจคไทล์บนพิกัดขนานออกมา ดังรูปที่ 4.13



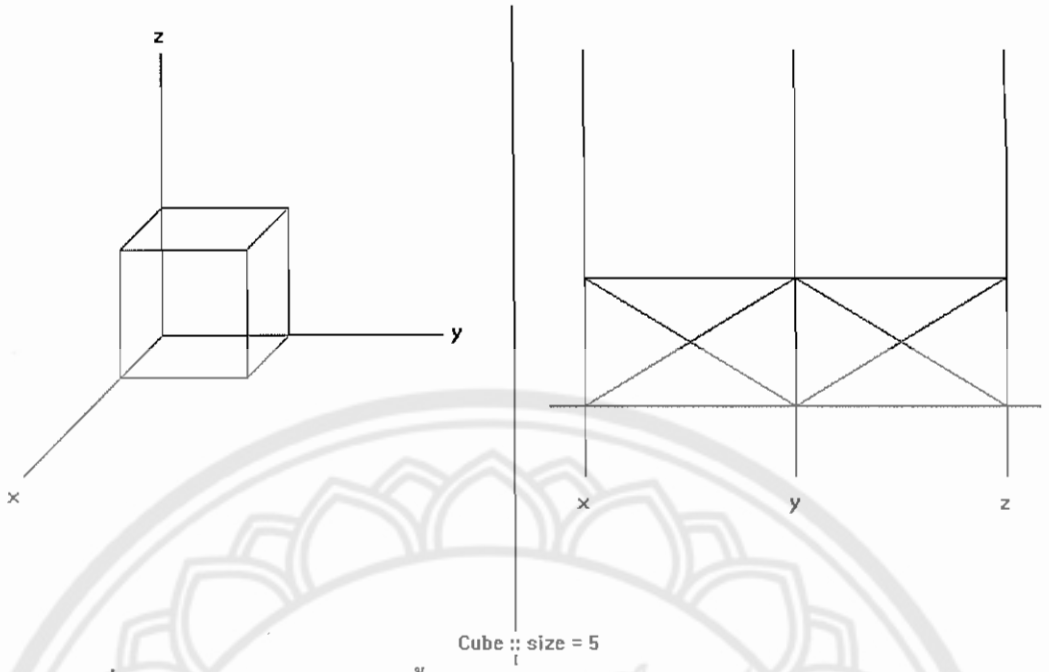
รูปที่ 4.13 รูปของโปรเจคไทล์ โดยให้ความเร็วเริ่มต้น = 300 m/s และมุม = 45 องศา

4.1.7 ลูกบาศก์และทรงกลม (Cube and Sphere) เลือกที่ Cube and Sphere จะมีกล่องข้อความ Cube and Sphere ขึ้นมา ดังรูปที่ 4.14

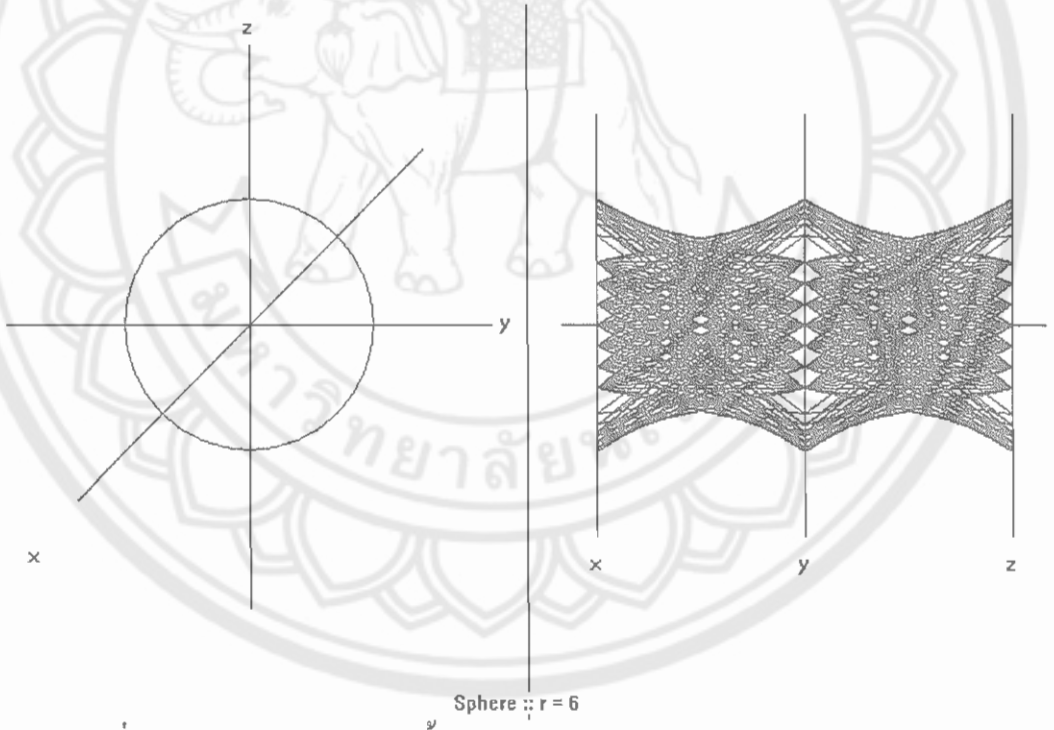


รูปที่ 4.14 กล่องข้อความ Cube and Sphere

เลือกรูปที่ต้องการจะวาดระหว่างลูกบาศก์หรือวงกลม แล้วกรอกขนาดของรูปที่ต้องการจะวาดลงไป (ขนาดของลูกบาศก์ คือ ความยาวของแต่ละด้าน, ขนาดทรงกลม คือ รัศมีของทรงกลม) จะได้ออกมาดังรูปที่ 4.15 และ 4.16

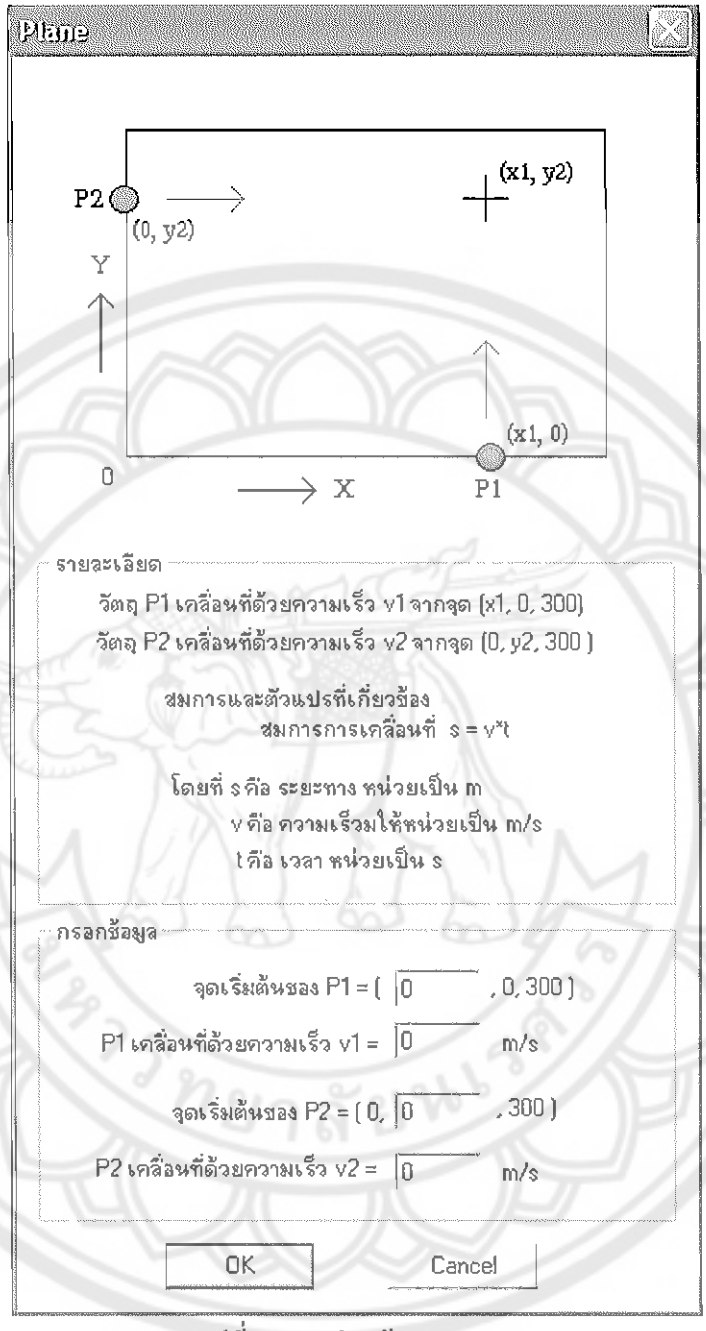


รูปที่ 4.15 รูปลูกบาศก์บนแกนตั้งฉากและบนแกนขนาน โดยมีขนาดเท่ากับ 5 หน่วย



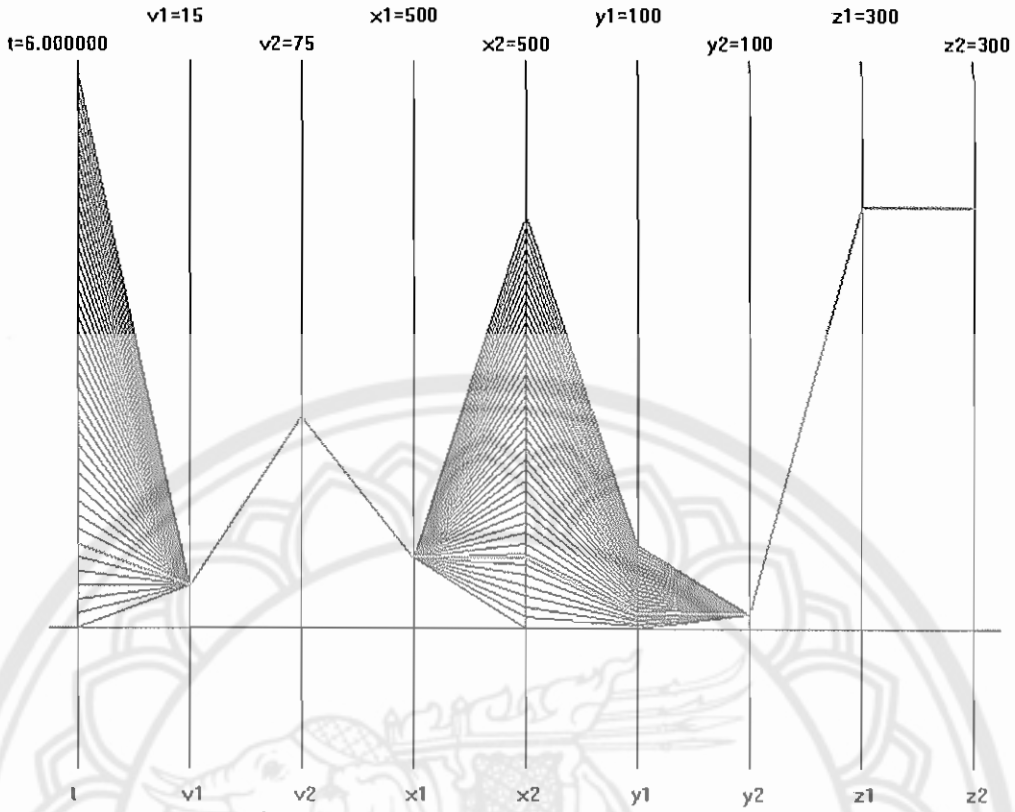
รูปที่ 4.16 รูปทรงกลมบนแกนตั้งฉากและบนแกนขนาน โดยมีรัศมีเท่ากับ 6 หน่วย

4.1.8 เครื่องบิน (Plane) เลือกที่ Plane จะมีกล่องข้อความ Plane ขึ้นมา ดังรูปที่ 4.17



รูปที่ 4.17 กล่องข้อความ Plane

กรอกข้อมูลของตำแหน่งเริ่มต้นและความเร็วของวัตถุ(สมมุติให้แทนเครื่องบิน)ทั้งสองลงไป โดยที่ วัตถุ P1 จะให้ใส่ตำแหน่งของ x ส่วนค่า y และ z เป็นค่าคงที่ เท่ากับ 0 และ 300 ตามลำดับ ส่วนวัตถุ P2 ให้ใส่ตำแหน่งของ y ลงไป โดยที่ค่า x และ z เป็นค่าคงที่ เท่ากับ 0 และ 300 ตามลำดับเช่นกัน แล้วกด OK จะได้รูปออกมาดังตัวอย่าง รูปที่ 4.18



รูปที่ 4.18 รูปของการจำลองการเคลื่อนที่ของเครื่องบิน

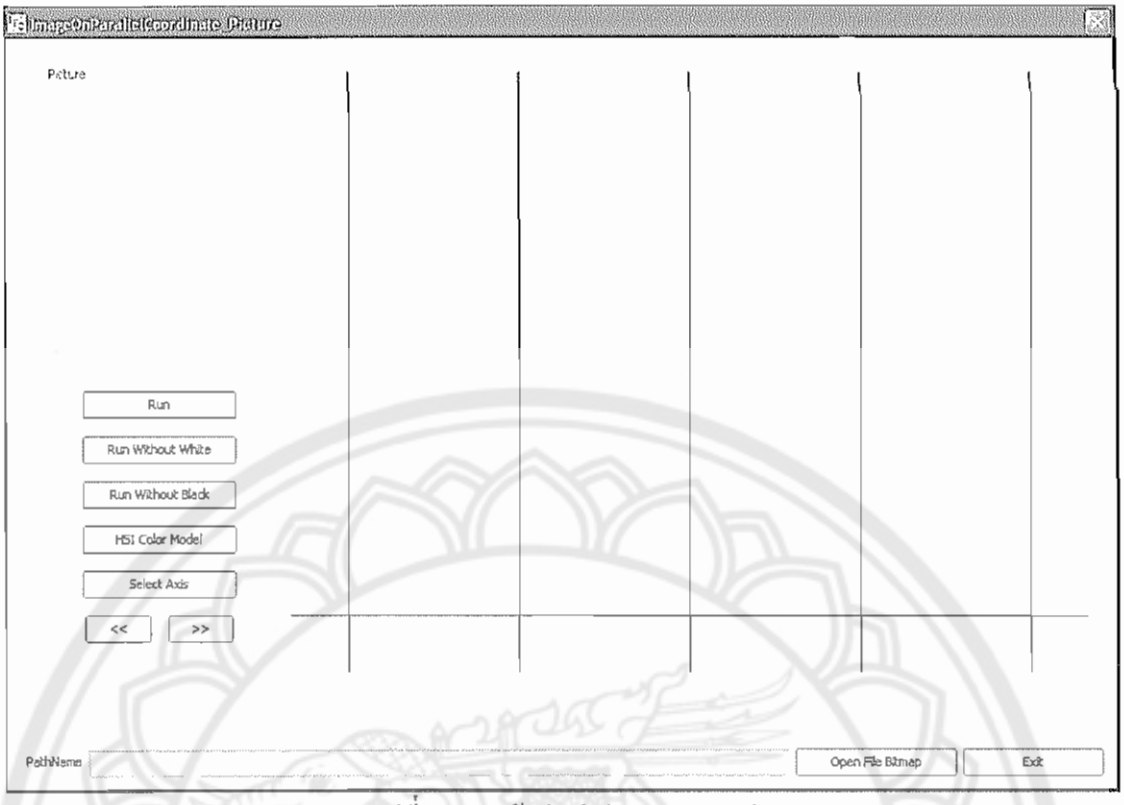
จากรูปที่ 4.18 กำหนดให้ตำแหน่งเริ่มต้นของ P1 อยู่ที่ (500, 0, 300) มีความเร็วเท่ากับ 15 m/s และตำแหน่งเริ่มต้นของ P2 อยู่ที่ (0,100,300) มีความเร็วเท่ากับ 75 m/s

โดยที่จะมีปุ่มไว้สำหรับเลื่อนแกนเวลาดังนี้ คือ << ใช้สำหรับเลื่อนแกนเวลาไปทางซ้าย และ >> ใช้สำหรับเลื่อนแกนเวลาไปทางขวา

4.2 วิธีการใช้งานโปรแกรมที่ 2

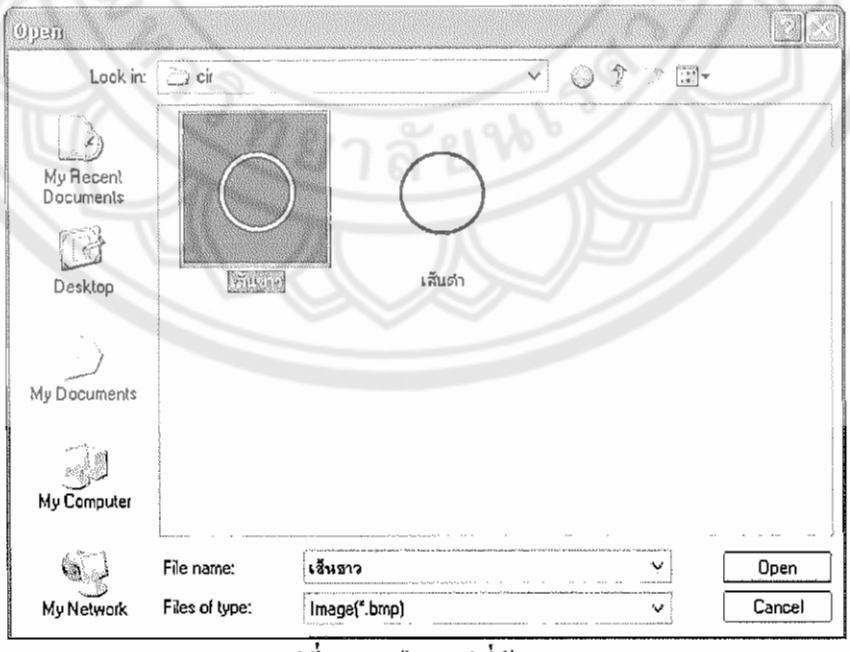
โปรแกรมนี้ คือ โปรแกรมที่ใช้แยกสีออกเป็น RGB, HSI, Gray Scale และ สีขาวและสีดำ ซึ่งการใช้งานโปรแกรมมีดังต่อไปนี้

เปิดโปรแกรมขึ้นมา โดยที่โปรแกรมจะมีลักษณะดังรูปที่ 4.19



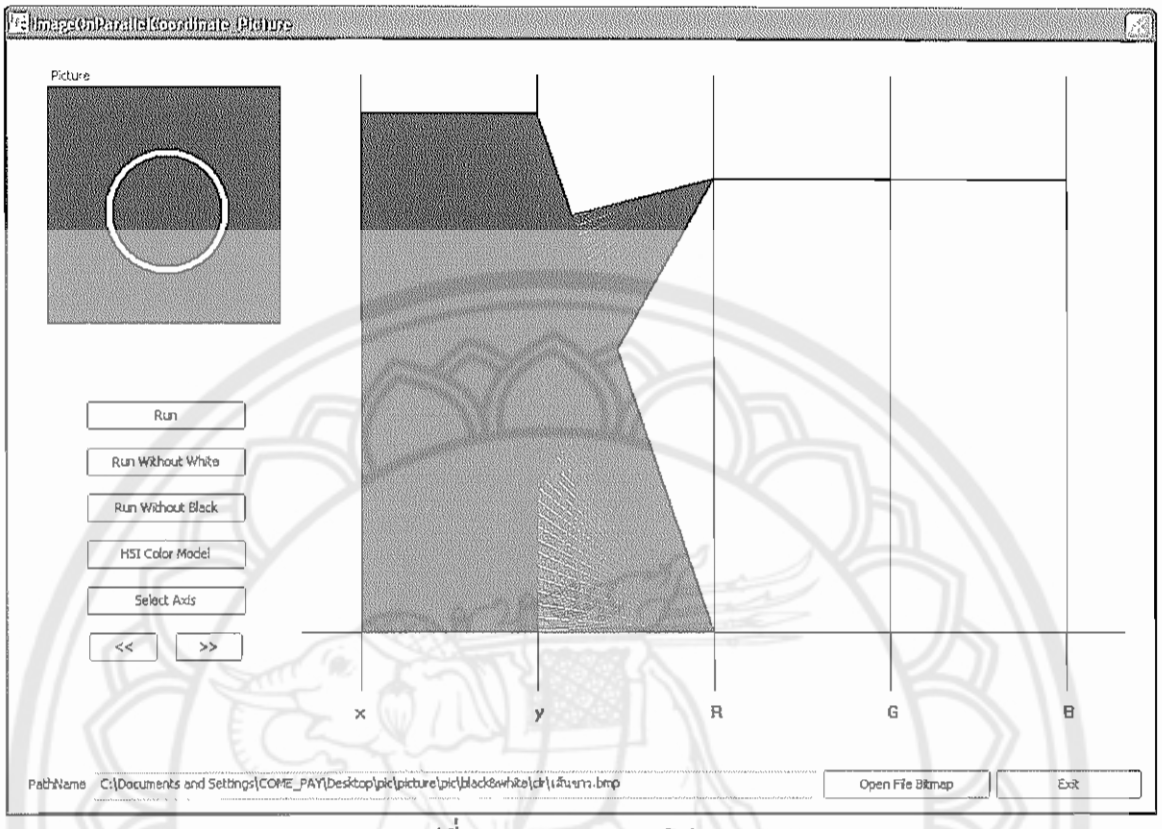
รูปที่ 4.19 หน้าต่างโปรแกรมแยกสี

เปิดไฟล์รูปภาพขึ้นมาโดยเลือกที่ Open File Bitmap แล้วทำการเลือกรูปภาพที่ต้องการจะนำมาใช้ ซึ่งรูปภาพที่ใช้จะต้องเป็นภาพ Bitmap เท่านั้น ดังรูปที่ 4.20



รูปที่ 4.20 เลือกรูปที่ต้องการ

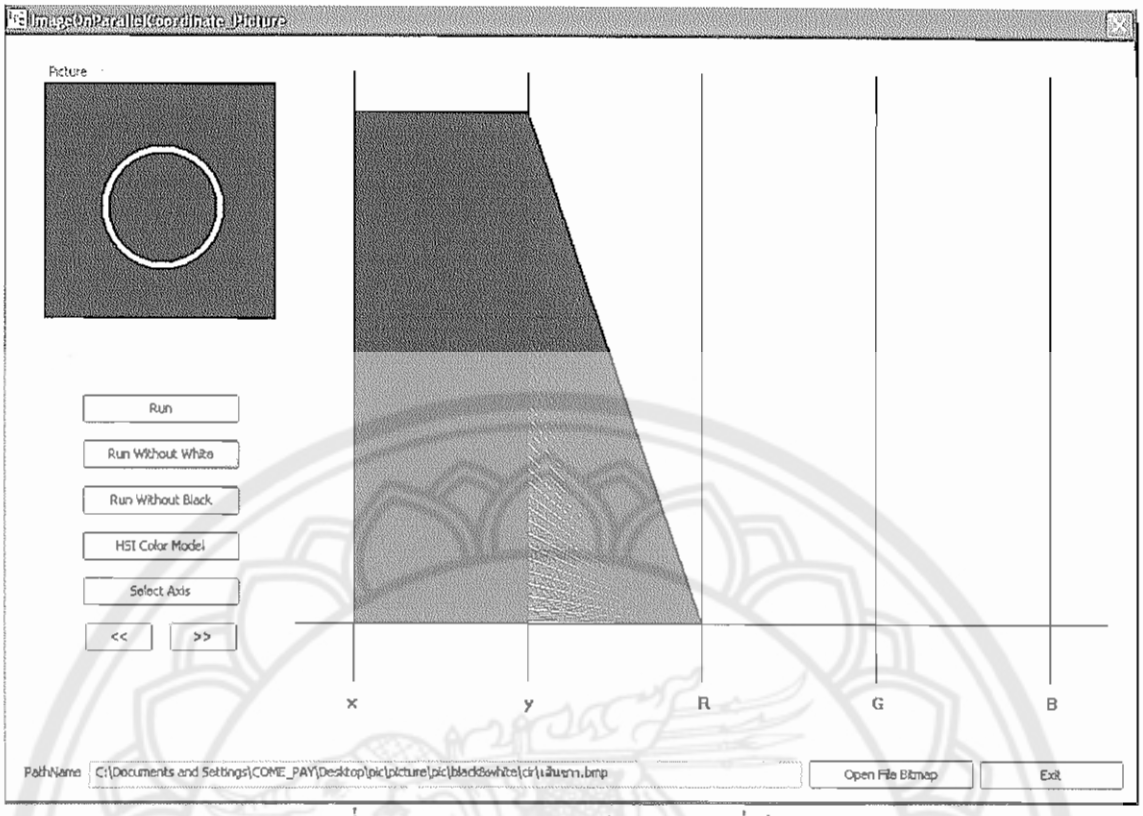
เมื่อเลือกรูปมาแล้วก็จะมาปรากฏที่โปรแกรม แล้วกด Run ได้ผลออกมาดังรูปที่ 4.21



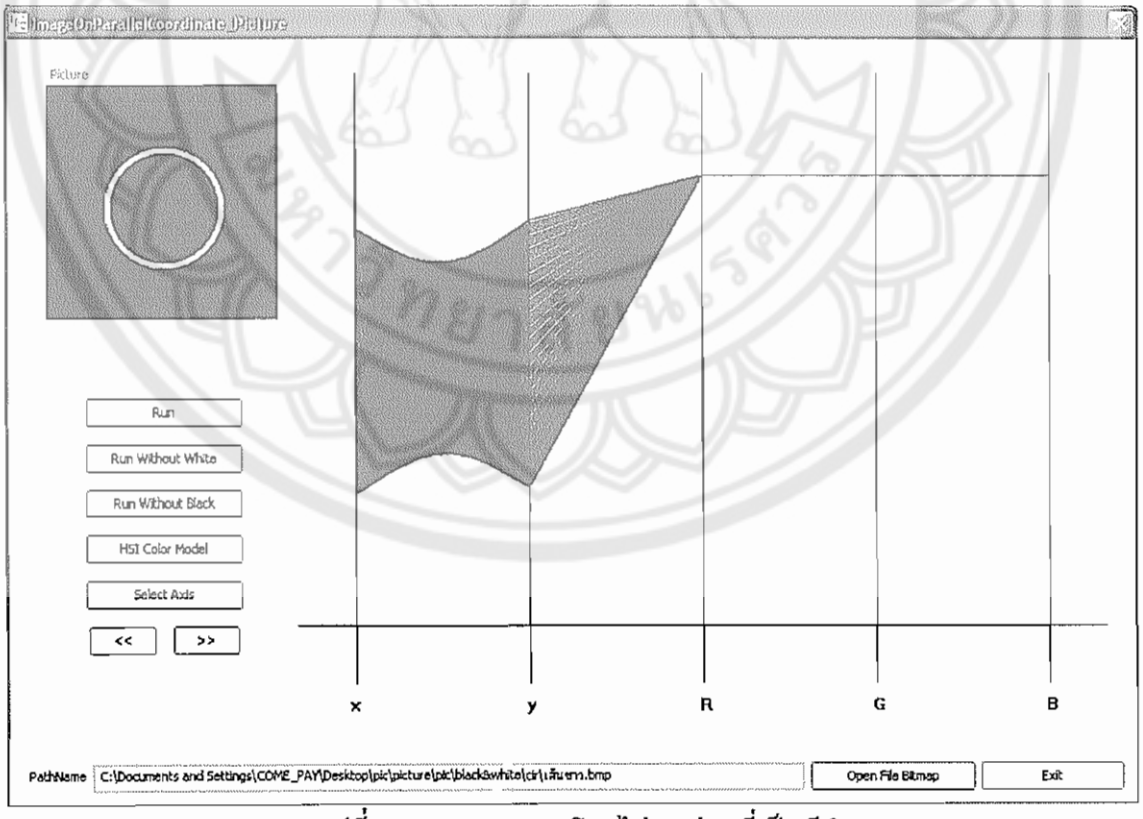
รูปที่ 4.21 ผลการ Run โปรแกรม

จากรูปที่ 4.21 ผลที่ออกมาจะมีแกนอยู่ด้วยกัน 5 แกน ซึ่งมีรายละเอียดดังนี้ คือ แกน x และแกน y เป็นตำแหน่งของแต่ละ pixel ในรูปนั้นๆ ส่วนแกน R G และ B นั้นเป็นแกนที่บอกค่าสีของแต่ละ pixel โดย R คือ สีแดง G คือ สีเขียว และ B คือ สีน้ำเงิน

ผลจากรูปนี้จะเห็นได้ว่าทั้งหมดแยกออกไปสองจุด คือ จุดที่มีค่าสีสูงที่สุด (255, 255, 255) ซึ่งก็คือสีขาว อีกจุดหนึ่งคือจุดที่มีค่าสีต่ำที่สุด (0, 0, 0) คือสีดำ โดยโปรแกรมนี้สามารถเลือก Run ได้โดยที่ถ้าเราไม่ต้องการสีขาว ก็ให้เลือกที่ Run Without White จะได้ผลออกมาดังรูปที่ 4.22 แต่ถ้าไม่ต้องการเอาสีดำ ให้เลือกที่ Run Without Black จะได้ผลออกมาดังรูปที่ 4.23



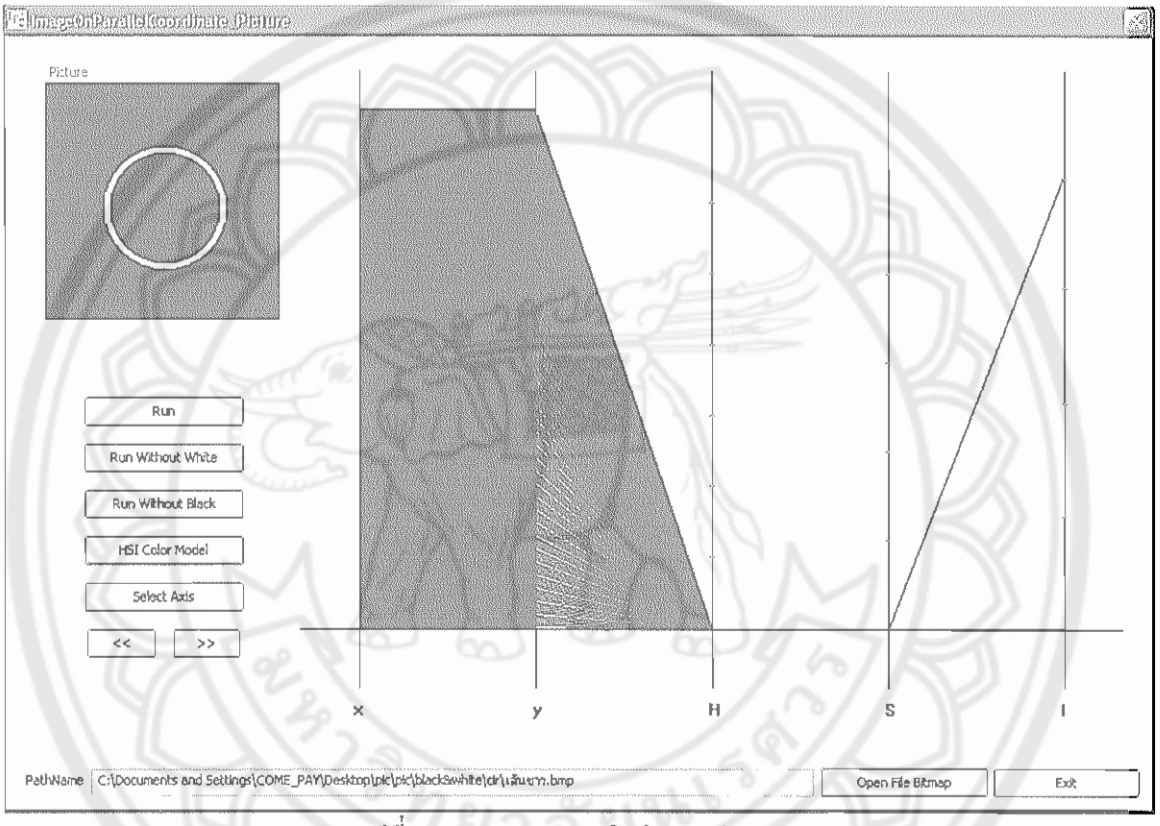
รูปที่ 4.22 ผลการ Run โดยไม่เอาส่วนที่เป็นสีขาว



รูปที่ 4.23 ผลการ Run โดยไม่เอาส่วนที่เป็นสีดำ

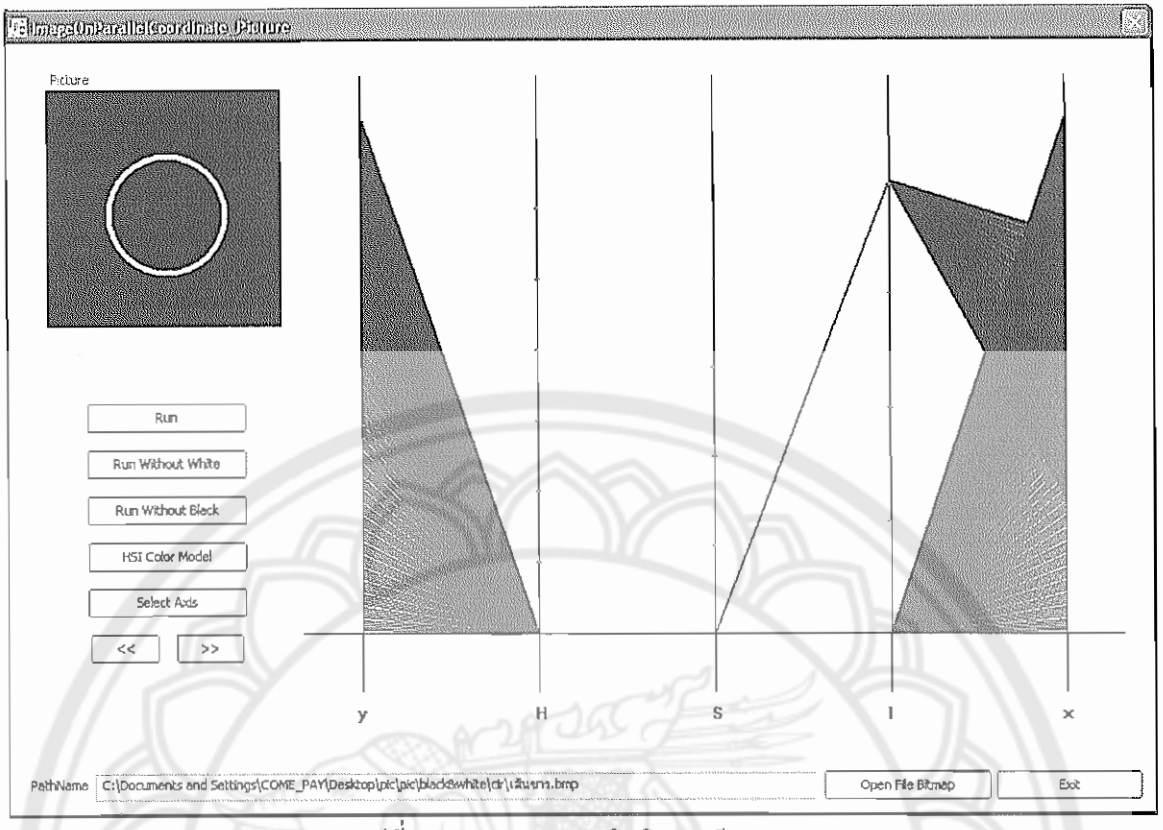
จากรูปที่ 4.22 จะเห็นว่าส่วนที่เป็นสีขาวนั้นไม่ได้แสดงออกมาด้วย จะมีเฉพาะในส่วนที่ pixel เป็น สีดำเท่านั้น รูปที่ได้จึงลงมารวมกันที่จุดที่มีค่าสีต่ำที่สุดเพียงจุดเดียว ส่วนรูปที่ 4.23 นั้นจะมีขึ้นมาเฉพาะส่วนที่เป็นวงกลมสีขาวเท่านั้น ส่วนที่เป็นสีดำจะไม่แสดงออกมา รูปที่ได้จึงรวมกันที่จุดๆ เดียวเช่นกัน ซึ่งคือจุดที่มีค่าสีสูงที่สุด

สำหรับโหมดสีแบบ HSI สามารถRun โปรแกรมได้โดยเลือกที่ HSI Color Model แล้วจะได้ผลออกมาดังรูปที่ 4.24



รูปที่ 4.24 ผลการ Run ในโหมดสีแบบ HSI

ซึ่งจากรูปจะเห็นได้ว่าค่า H และ S ที่ได้เป็น 0 ทั้งหมด และจะมีค่า I เท่านั้นที่มีค่าเป็น 1 โดยที่เป็นส่วนของที่เป็นสีขาว ซึ่งเราสามารถดูได้โดยที่เลื่อนแกนใดแกนหนึ่งระหว่าง x กับ y ให้ไปอยู่ข้างแกน I โดยเลือกที่ Select Axis แล้วเลือกแกน x และเลือก << จะได้ผลออกมาดังรูปที่ 4.25



รูปที่ 4.25 ผลการ Run ในโหมดสีแบบ HSI

ซึ่งจะเห็นได้ว่ากลุ่ม pixel ที่เป็นสีขาวจะมีค่า I = 1 ส่วนกลุ่ม pixel ที่เป็นสีดำ I = 0

4.3 ผลการทดลอง

ในการทดลอง ได้ทำการทดลองต่างๆ ดังนี้

การทดลองที่ 1 ทดลองใส่ค่าต่างๆ ของการวาดรูปแต่ละแบบลงไป โดยทดลองใส่เป็นจำนวนเต็มบวก จำนวนเต็มลบ ศูนย์ จำนวนทศนิยม และหาค่าต่ำสุดและสูงสุดที่สามารถใส่ลงไปแล้วทำให้ได้รูปที่เหมาะสม

การทดลองที่ 2 วัดค่าที่ได้จากโปรแกรมวาดรูปการเคลื่อนที่แบบโปรเจคไทล์ ซึ่งได้ค่าผลลัพธ์จากการคำนวณออกมา ซึ่งเป็นค่าของเวลาที่ใช้ในการเคลื่อนที่ทั้งหมด ระยะสูงสุดในแนวราบ และระยะทางสูงสุดในแนวดิ่ง โดยทดลองที่ความเร็วต้นเท่ากับ 50, 100, 150, 200, 250 และ 300 โดยที่ให้มุมการเคลื่อนที่เท่ากับ 0, 15, 30, 37, 45, 53, 60, 75 และ 90

การทดลองที่ 3 ทดลองแบบจำลองการชนกันของเครื่องบิน โดยจะกำหนดให้มีตำแหน่งเริ่มต้นของเครื่องบินแต่ละลำที่แตกต่างกัน โดยในลำที่ 1 ใส่ตำแหน่งบนระนาบ x ในลำที่ 2 บนระนาบ โดย z คงที่ และมีความเร็วที่ต่างๆกันออกไป โดยจะดูว่าเครื่องบินทั้ง 2 จะชนกันหรือไม่

การทดลองที่ 4 ทำการทดลองแยกสีของโปแกรมโดยใช้รูปขาวดำและรูปสี แล้วดูว่าโปรแกรมสามารถแยกออกมาได้อย่างไรบ้าง โดยทำการทดลองทั้งการ Run แยกสีในโหมดสี RGB, Run without white, Run without black และทดลองแยกสีในโหมดของ HSI

ตารางที่ 4.1 ผลการทดลองที่ 1

ชนิดของรูป	ตัวแปร	ศูนย์	จำนวนเต็ม		ทศนิยม	ค่าต่ำสุด	ค่าสูงสุด
			บวก	ลบ			
เส้นตรง	m	√	√	√	√	-10	10
	c	√	√	√	√	-200	200
วงกลม	r	×	√	×	√	> 0	< 51
พาราโบลา	c	×	√	√	√	> -31	< 31
ไฮเปอร์โบลา	a	×	√	×	×	1	28
	b	×	√	×	×	1	28
วงรี	a	×	√	×	√	> 0	50
	b	×	√	×	√	> 0	50
โปรเจคไทล์	ความเร็วต้น	×	√	×	√	1	*
	มุม	×	√	×	√	1	90
ลูกบาศก์	ขนาด(กว้าง)	×	√	×	×	1	30
ทรงกลม	ขนาด(รัศมี)	×	√	×	×	1	30
เครื่องบิน	x ของ P1	√	√	√**	×	0	***
	y ของ P2	√	√	√**	×	0	***
	ความเร็ว P1	×	√	√**	×	1	***
	ความเร็ว P2	×	√	√**	×	1	***

* ความเร็วต้นสูงสุดขึ้นอยู่กับมุมในการเคลื่อนที่ ซึ่งเมื่อคำนวณออกมาแล้วจะต้องได้ y_{max} ไม่เกิน 4000 เมตร เนื่องจากถ้าเกินกว่านี้รูปที่ได้จะเลยแกนออกไป

** สามารถใส่ค่าเป็นลบได้ แต่รูปที่แสดงออกมาจะไม่สวยงามเนื่องจากเวลาที่คิดออกมาหรือระยะทางที่ได้ออกมาจะเป็นค่าลบ จะไปอยู่ใต้ แกน 0 จึงไม่แนะนำให้ใส่ค่าที่เป็นลบ

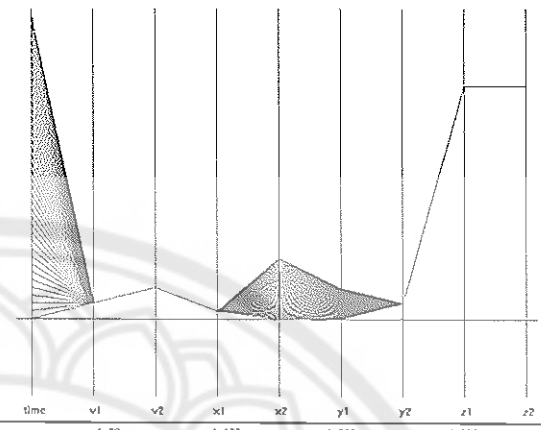
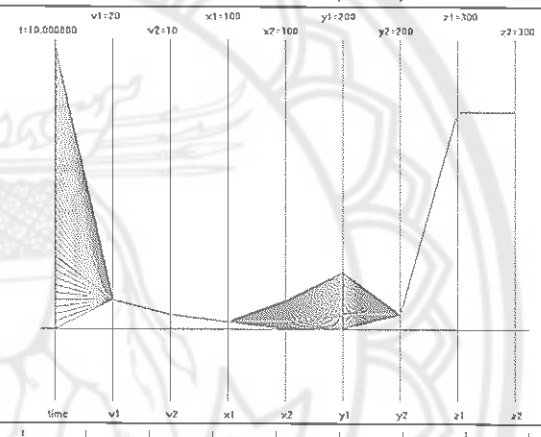
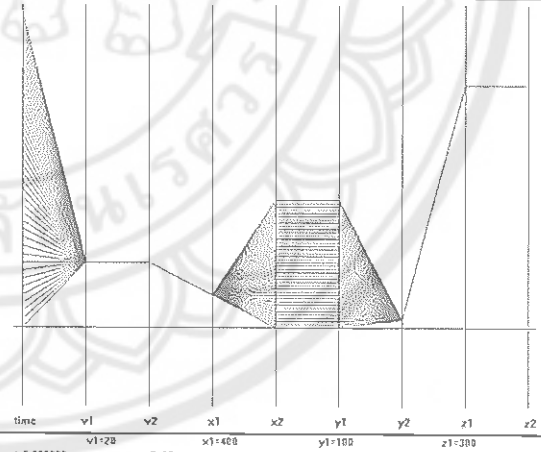
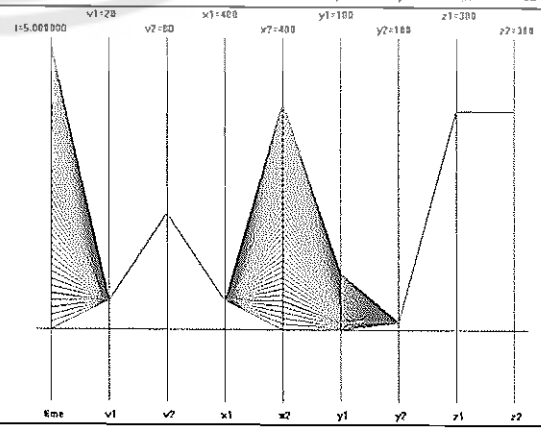
*** ทั้งหมดขึ้นอยู่ซึ่งกันและกัน โดยรูปจะออกมาสวยงามและเหมาะสมที่สุด (ไม่เกินกรอบรูปของโปรแกรม) ก็ต่อเมื่อ เวลาที่ชนกันนั้นต้องไม่เกิน 40 วินาที

ตารางที่ 4.2 ผลการทดลองที่ 2

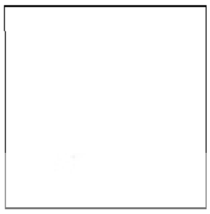
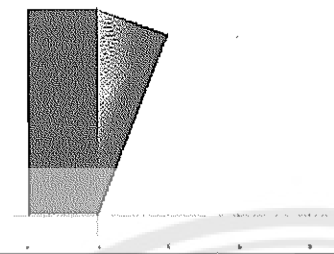
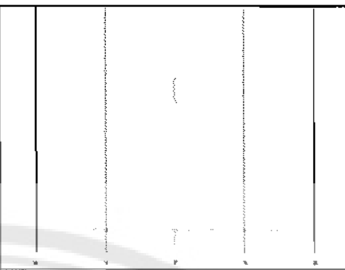
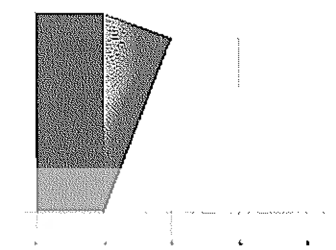
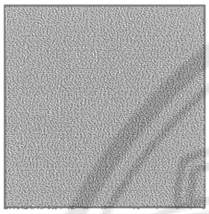
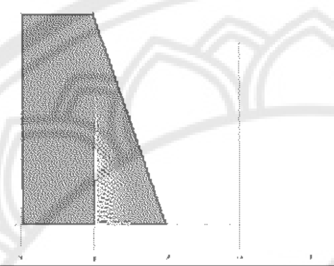
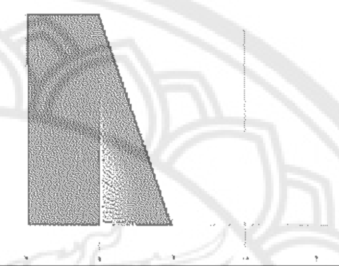
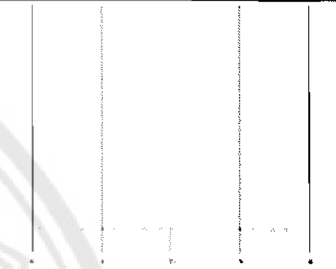
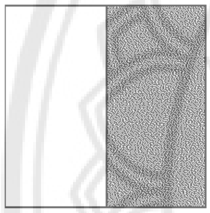
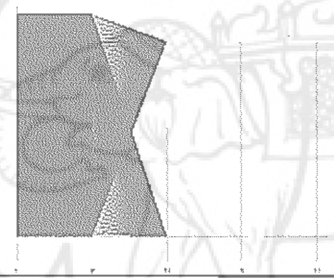
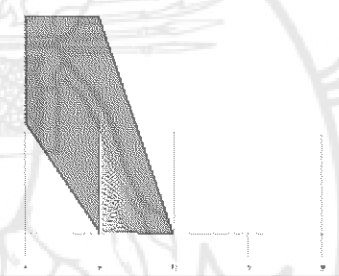
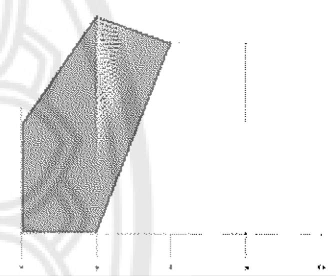
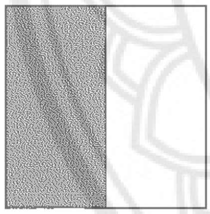
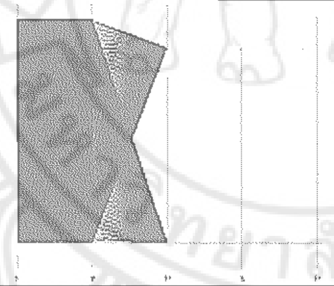
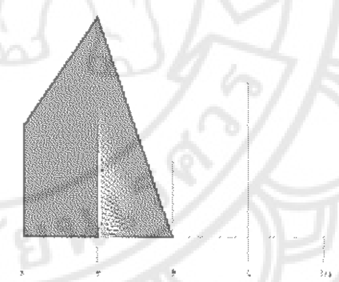
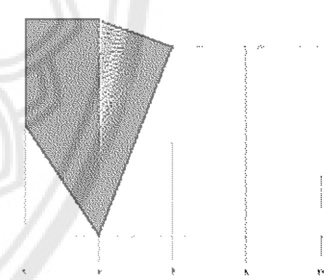
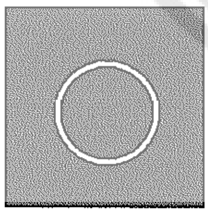
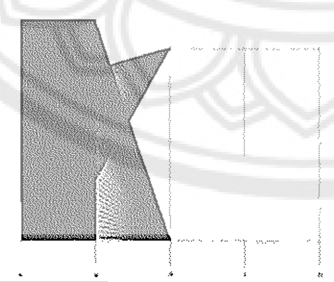
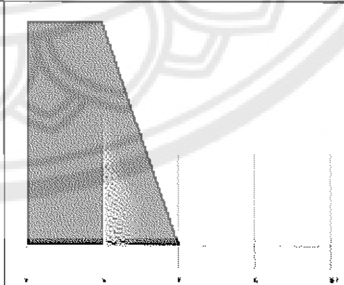
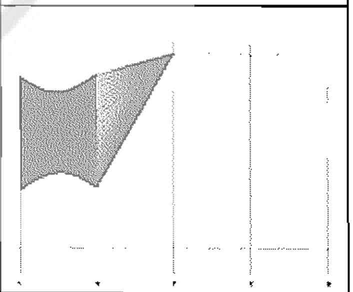
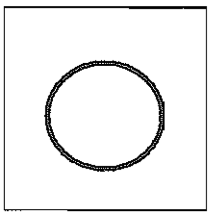
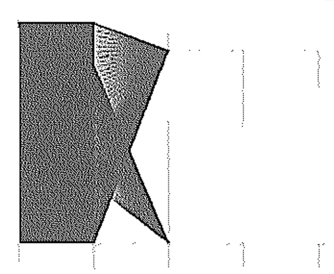
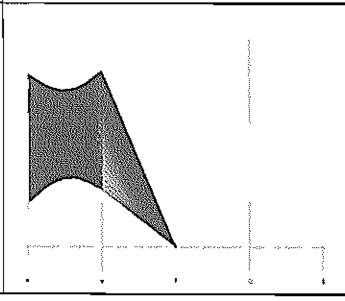
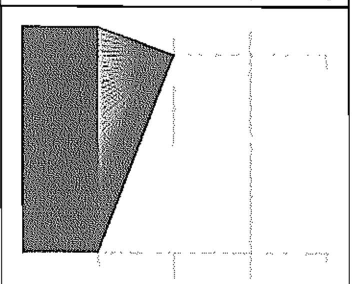
มุม \ ความเร็วต้น		50 m/s	100 m/s	150 m/s	200 m/s	250 m/s	300 m/s
0	t_{max}	-	-	-	-	-	-
	x_{max}	-	-	-	-	-	-
	y_{max}	-	-	-	-	-	-
15	t_{max}	2.588	5.176	7.765	10.353	12.941	15.529
	x_{max}	125.000	500.000	1125.000	2000.000	3125.000	4500.000
	y_{max}	8.373	33.493	75.361	133.945	209.335	301.443
30	t_{max}	5.000	10.000	15.000	20.000	25.000	30.000
	x_{max}	216.506	866.025	1948.557	3464.102	5412.659	7794.229
	y_{max}	31.250	125.000	281.250	500.000	781.250	1125.000
37	t_{max}	6.018	12.036	18.054	24.073	30.091	36.109
	x_{max}	240.315	961.261	2162.839	3845.047	6007.886	8651.355
	y_{max}	45.273	181.091	407.454	724.363	1131.817	1629.816
45	t_{max}	7.071	14.142	21.213	28.284	35.355	42.426
	x_{max}	250.000	1000.000	2250.000	4000.000	6250.000	9000.000
	y_{max}	62.500	250.000	562.500	1000.000	1562.500	2250.000
53	t_{max}	7.986	15.973	23.959	31.945	39.932	47.918
	x_{max}	79.727	961.262	2162.839	3845.047	6007.886	8651.355
	y_{max}	204.315	318.909	717.546	1275.637	1993.183	2870.184
60	t_{max}	8.662	17.321	25.981	34.641	43.301	51.962
	x_{max}	216.506	866.025	1948.557	3464.102	5412.659	7794.229
	y_{max}	93.750	375.000	843.750	1500.000	2343.750	3375.000
75	t_{max}	9.659	19.318	28.978	38.637	48.296	*
	x_{max}	125.000	500.000	1125.000	2000.000	3125.000	*
	y_{max}	116.627	466.506	1049.639	1866.025	2915.665	*
90	t_{max}	10.000	20.000	30.000	40.000	50.000	*
	x_{max}	0.000	0.000	0.000	0.000	0.000	*
	y_{max}	125.000	500.000	1125.000	2000.000	3125.000	*

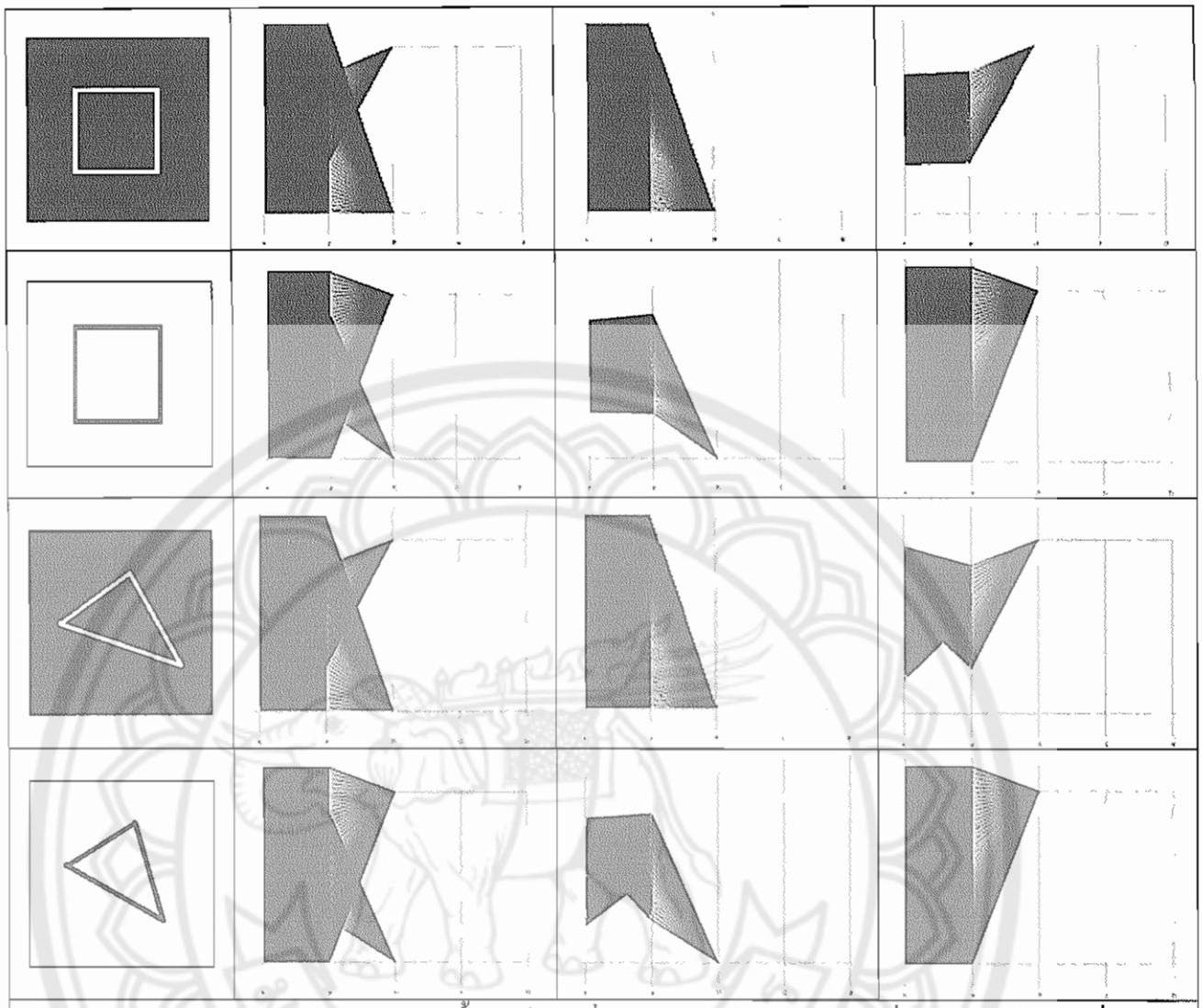
*ค่าเกินกว่าที่โปรแกรมจะวาดได้

ตารางที่ 4.3 ผลการทดลองที่ 3

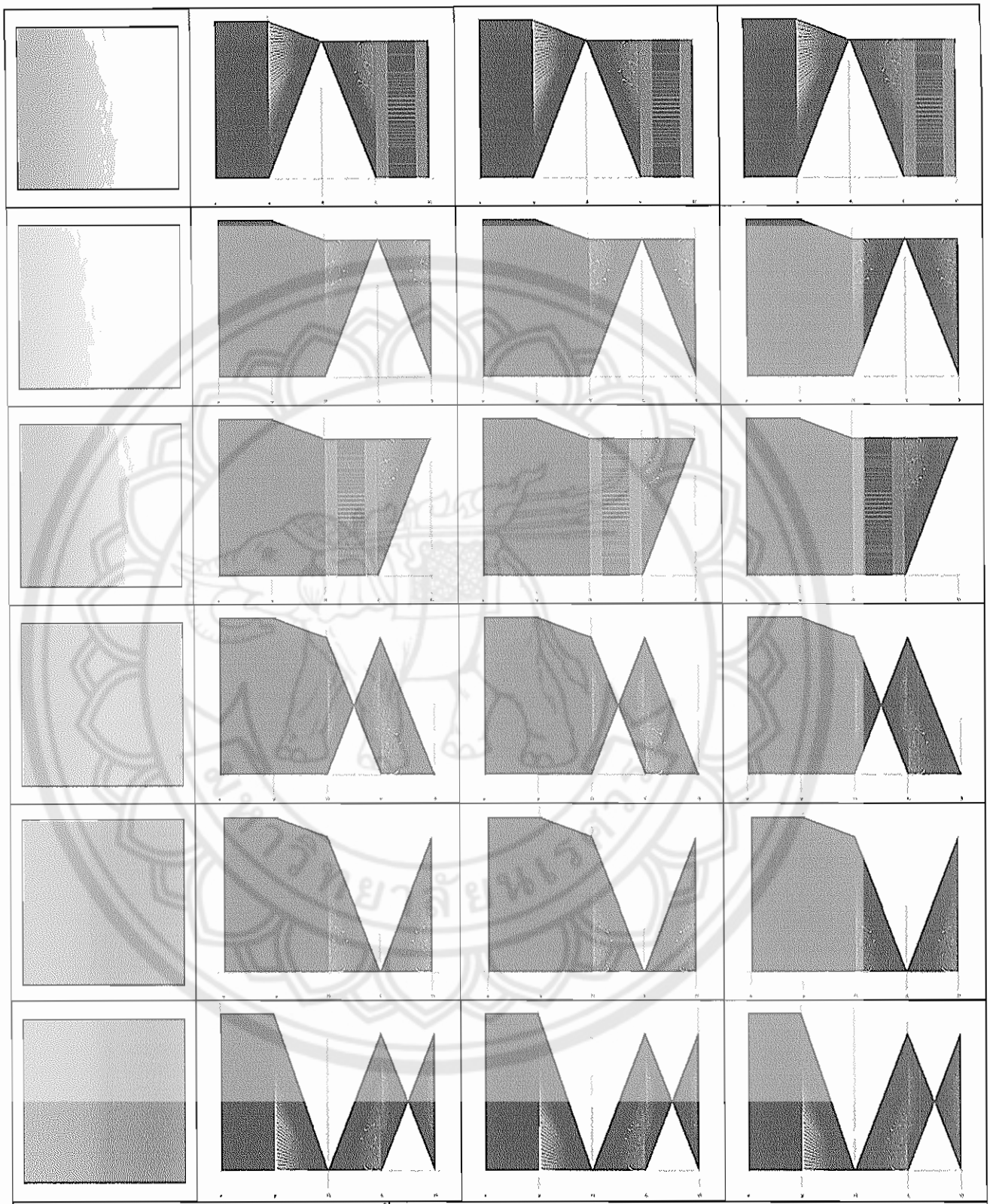
ตำแหน่งเริ่มต้น		ความเร็ว		ผลการทดลอง (เส้นสีแดงคือตำแหน่งและเวลาที่ชนกัน)
x ลำที่ 1	y ลำที่ 2	ลำที่ 1	ลำที่ 2	
100	200	10	20	
100	200	20	10	
400	100	40	40	
400	100	20	80	

ตารางที่ 4.4 ผลการทดลองที่ 4 โหมดสี RGB

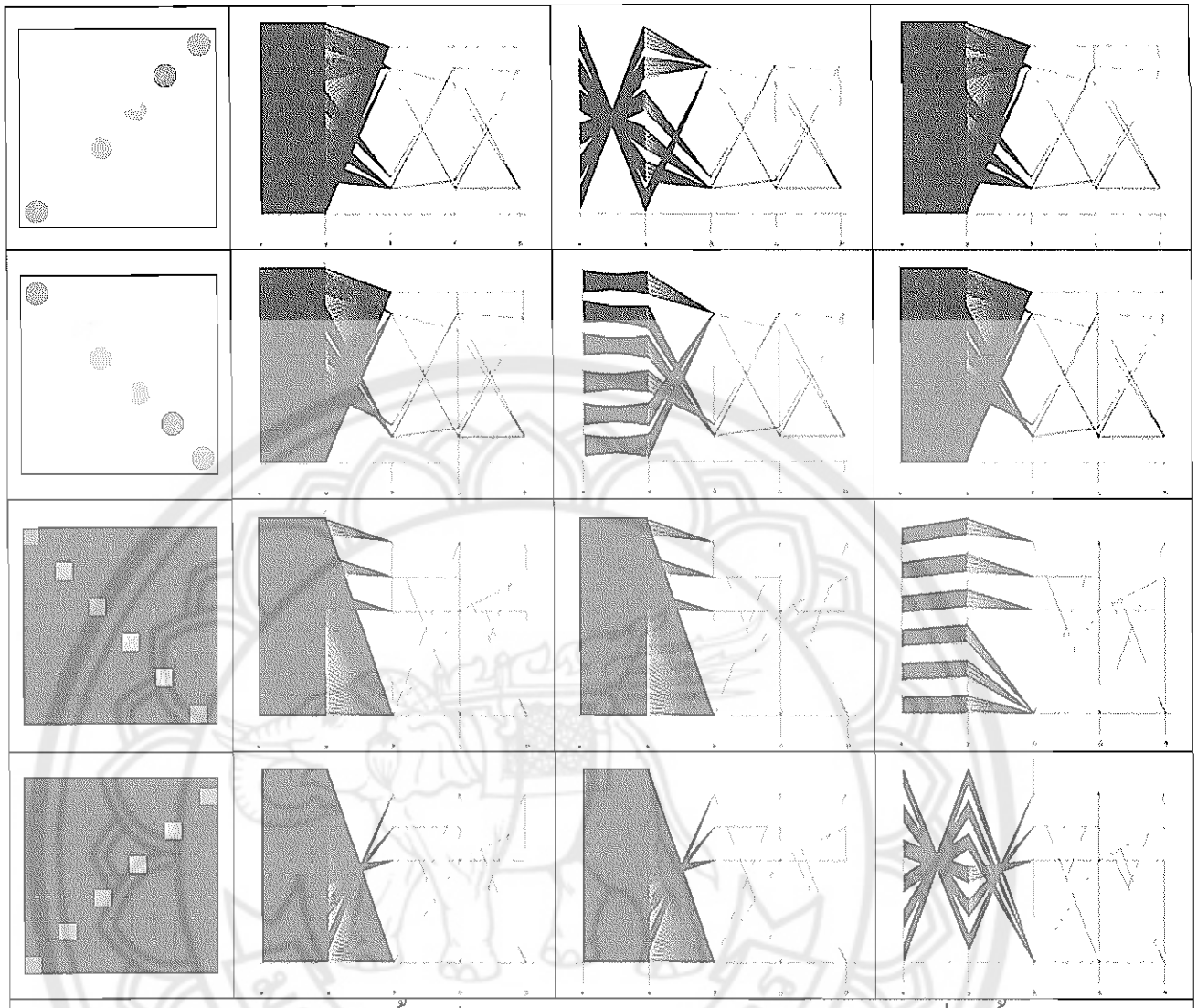
รูปที่ใช้	ผลการ Run		
	Run	Run without white	Run without Black
			
			
			
			
			
			



จากผลการทดลองของรูปภาพชุดนี้ จะเห็นว่าเมื่อทำการ Run ธรรมดา ผลที่ออกมาจะเป็นภาพที่เกิดจากทั้งส่วนที่เป็นสีขาว และสีดำ แต่เมื่อ Run without White หรือ Run without Black ผลที่ได้ออกมาจะแสดงสเปกตรัมของสี RGB อย่างชัดเจน

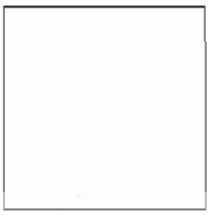
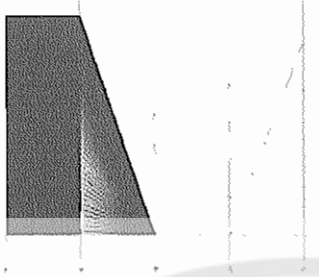
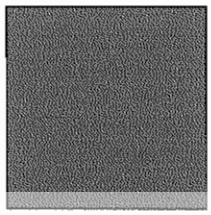
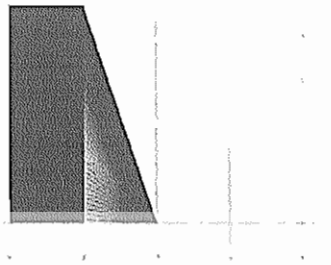
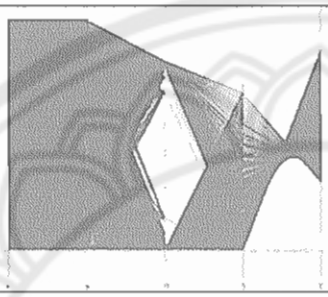
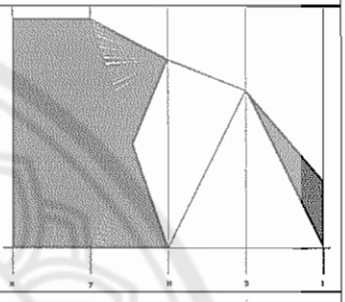
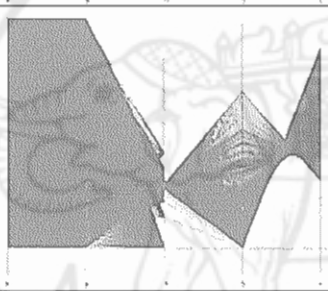
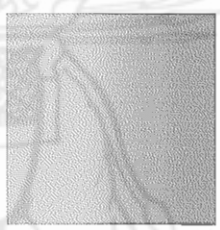
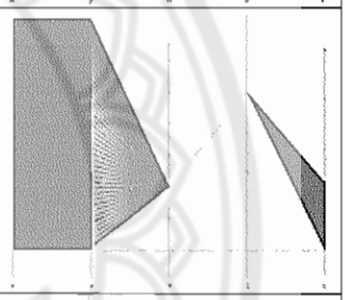
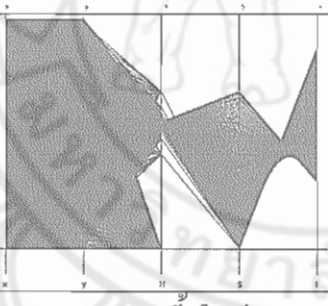
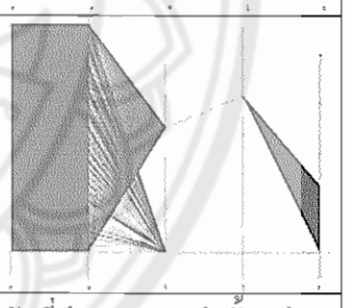
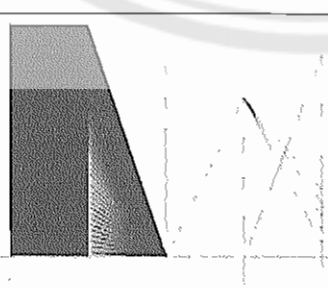
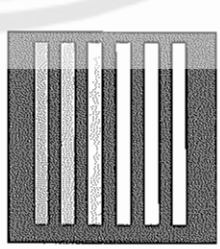
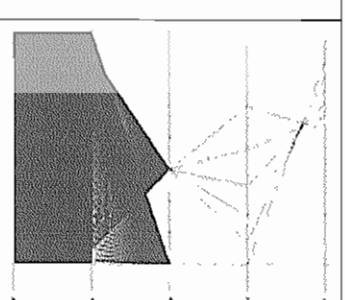


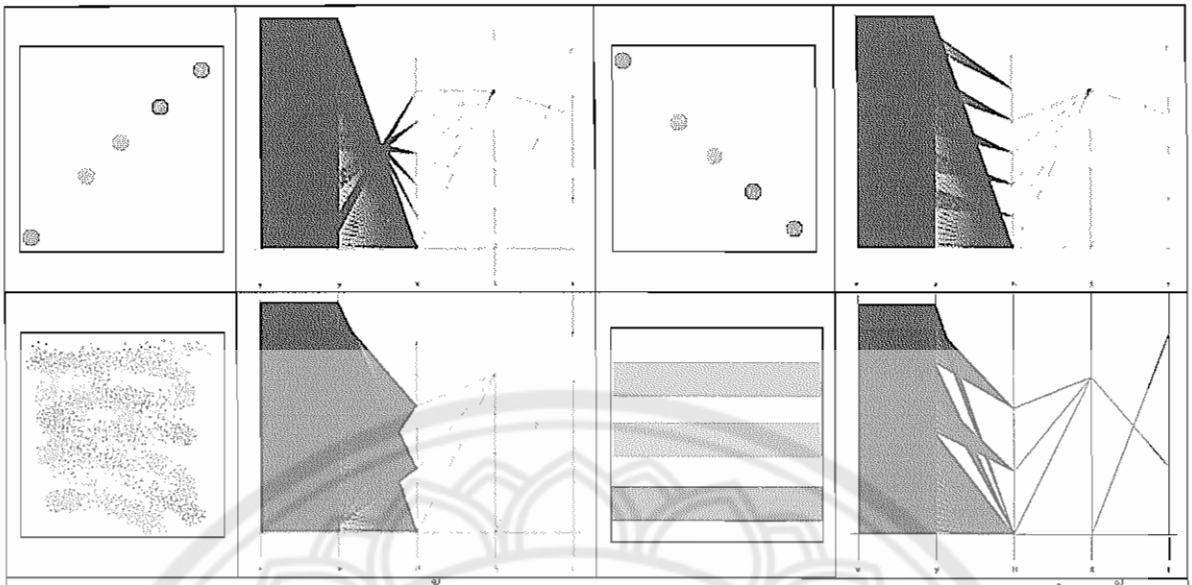
จากผลการทดลองของรูปภาพชุดนี้ จะเป็นการแสดงสเปกตรัมของสี ที่เกิดการเปลี่ยนแปลงของสีต่างๆ ในแต่ละภาพ โดยที่ผลการ Run จากทั้งสามแบบ จะเหมือนกันหรือต่างกันน้อยมาก เนื่องจากว่า มีส่วนที่เป็นสีขาว และสีดำน้อยมาก ทำให้เห็นการเปลี่ยนแปลงไม่ชัดเจน



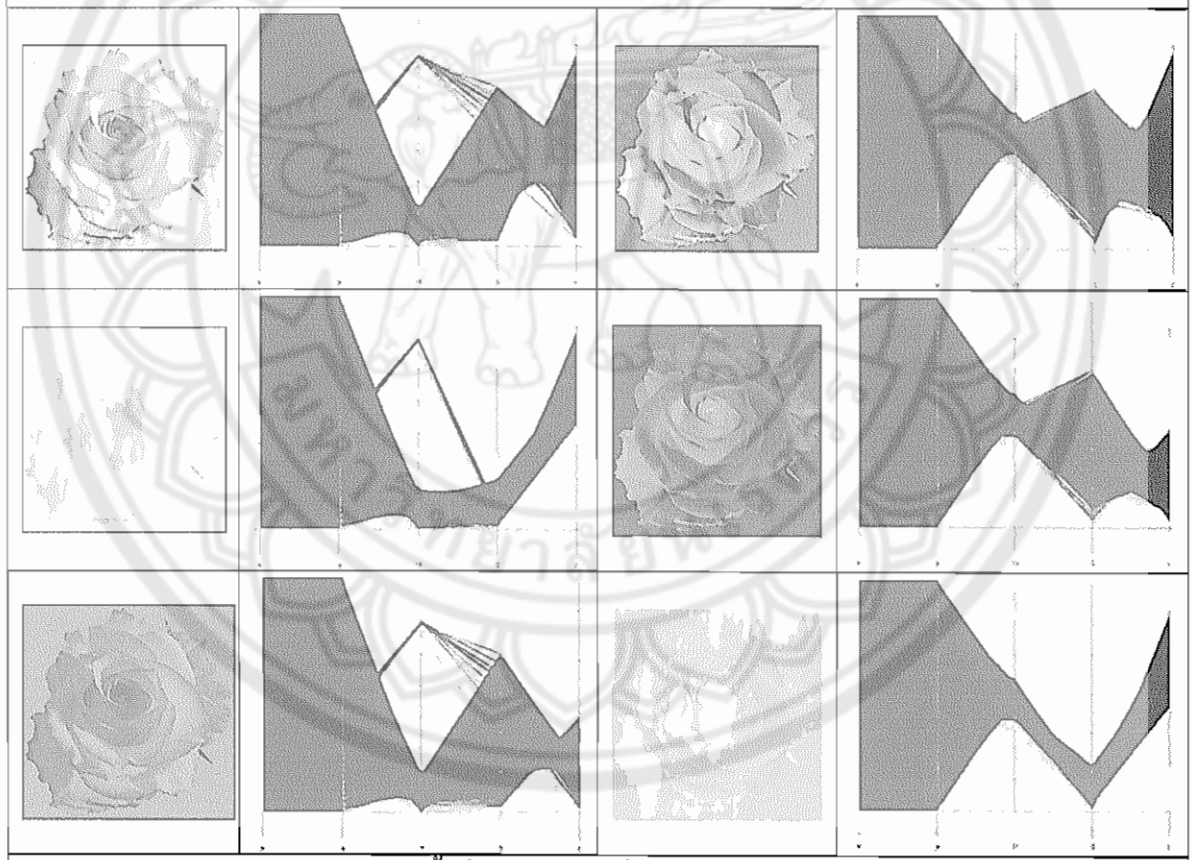
จากการทดลองของภาพชุดนี้ จะเห็นว่าภาพแตกต่างกัน แต่สเปกตรัมของสี RGB ที่เกิดขึ้นในแต่ละภาพจะเหมือนกัน ซึ่งส่วนที่แตกต่างกันนั้น คือ ค่าของสีที่อยู่ในแต่ละ Pixel

ตารางที่ 4.5 ผลการทดลองที่ 4 โหมดสี HSI

รูปที่ใช้	ผลการ Run	รูปที่ใช้	ผลการ Run
			
			
			
			
จากการทดลองของภาพชุดนี้ ในส่วนของสีขาวและดำจะแตกต่างกันที่ค่า ของ I เท่านั้น ส่วนในภาพที่เปลี่ยนจากสีจากแดง เขียว น้ำเงินเป็นขาวหรือดำ จะมีการเปลี่ยนแปลงค่าของ H เพียงอย่างเดียวเท่านั้น			
			
จากหารทดลองของภาพชุดนี้ ในภาพซ้าย ค่าของ H และ S จะมีค่าเท่าเดิมทั้งหมด เปลี่ยนเฉพาะค่า I เท่านั้นเนื่องจากรูปจะมืดลงเรื่อยๆ ส่วนในภาพขวา ค่าของ H เท่าเดิม จะเปลี่ยนในส่วนของ S และ I เนื่องจากว่าภาพจะสว่างขึ้นเรื่อยๆ			



จากการทดลองของภาพชุดนี้ จะเห็นว่าภาพแตกต่างกัน แต่สเปกตรัมของสี RGB ที่เกิดขึ้นใน แต่ละภาพจะเหมือนกัน ซึ่งส่วนที่แตกต่างกันนั้น คือ ค่าของสีที่อยู่ในแต่ละ Pixel



จากการทดลองของภาพชุดนี้ เมื่อมีการปรับเพิ่มลดค่าความสว่าง จะเห็นว่าเกิดการเปลี่ยนแปลง สเปกตรัมของ S และ I อย่างชัดเจน โดยที่ค่า H เท่าเดิม

บทที่ 5

สรุปผล

โครงงานนี้ศึกษาถึงหลักการของพิกัดขนาน (Parallel Coordinate) แล้วนำมาเขียนขึ้นเป็นโปรแกรมที่ใช้สำหรับวาดรูปต่างๆ ขึ้นมาบนพิกัดขนาน โดยที่สามารถวาดรูปต่างๆ ได้ดังนี้คือ เส้นตรง, วงกลม, พาราโบลา, ไฮเปอร์โบลา, วงรี, ลูกบาศก์, ทรงกลม, การเคลื่อนที่แบบโปรเจคไทล์ และแบบจำลองเครื่องบินเพื่อหาว่าเครื่องบินสองเครื่องจะชนกันหรือไม่ ถ้าชนจะชนกันที่เวลาใด นอกจากนี้ได้เขียนโปรแกรมแยกสีของรูปภาพโดยใช้หลักการของ Image Processing เพื่อแยกสีของรูปออกมาว่ามีค่าความเข้มเท่าไหร่ในโหมดสีแบบ RGB และค่าต่างๆของสีในโหมดสีแบบ HSI การออกแบบการทดลองในโครงงานนี้ในส่วนของโปรแกรมที่ 1 ได้ทดลองวาดรูปโดยใช้ค่าต่างๆ เพื่อหาขอบเขตของโปรแกรม และทดลองวัดค่าที่ได้จากโปรแกรมวาดรูปการเคลื่อนที่แบบโปรเจคไทล์ ในส่วนของโปรแกรมที่ 2 ได้ทดลองแยกสีกับรูปต่างๆ เพื่อทดสอบประสิทธิภาพและหาข้อบกพร่องของโปรแกรมเพื่อให้เหมาะสมและมีประสิทธิภาพมากที่สุด

5.1 ผลการทดลอง

จากการทดลองที่ 1 จะพบว่าขอบเขตของค่าที่สามารถใส่โปรแกรมเพื่อทำการวาดภาพนั้น จะจำกัดตามทฤษฎีของรูปแต่ละแบบซึ่งแตกต่างกันไป และจำกัดตามขอบเขตที่กำหนดไว้ด้วยโปรแกรมเองเนื่องจากว่าค่าที่ใส่ไปนั้นจะทำให้ได้รูปที่มีขนาดใหญ่เกินไป จึงต้องจำกัดค่าเอาไว้ซึ่งแต่ละรูปก็จะมีค่าที่แตกต่างกันไป

จากการทดลองที่ 2 ทดลองวาดโปรเจคไทล์ แล้วนำค่าที่โปรแกรมคำนวณได้มาได้เปรียบเทียบกัน ซึ่งได้ผลออกมาถูกต้องตามทฤษฎีของโปรเจคไทล์ว่า วัตถุสามารถไปไกลที่สุดในแนวราบได้โดยต้องทำมุม 45 องศากับพื้นราบ

จากการทดลองที่ 3 จากการทดลองได้ใส่ค่าตำแหน่งเริ่มต้นและความเร็วของเครื่องบินทั้ง 2 ที่มีค่าแตกต่างกัน ให้กับโปรแกรม แล้วผลที่ได้ออกมานั้น สามารถบ่งบอกได้ถึง เวลาที่เครื่องบินทั้งสองจะชนกันและชนกันที่ตำแหน่งใด ซึ่งได้ผลตรงตามสมมุติฐานทุกๆ ในทุกๆค่า

จากการทดลองที่ 4 โปรแกรมสามารถแยกสีออกมาได้อย่างถูกต้อง ตามหลักการของโหมดสีทั้ง 2 แบบ ซึ่งจะมีเพียงบางรูปที่ค่าออกมาไม่ตรงตามสมมุติฐานเนื่องมาจากรูปที่นำมาใช้มีบาง pixel ที่มีค่าสีที่ผิดแปลกไปจากที่ควรจะเป็นอันเนื่องมาจากตอนที่สร้างรูปขึ้นมา

จากผลการทดลองพบว่าความสามารถของโปรแกรมทั้งสองเป็นไปตามวัตถุประสงค์ แต่จะเกิดข้อผิดพลาดบางส่วนจากการเขียนโปรแกรม และวิธีการคิดบางขั้นตอนเท่านั้น

ตารางที่ 5.1 ตัวอย่างสมการที่ใช้บนระบบพิกัดขนาน ซึ่งแปลงมาจากสมการบนพิกัดฉาก

รูป	สมการบนพิกัดฉาก	สมการบนพิกัดขนาน
เส้นตรง	$y = mx + c$	$y = mx + c$
วงกลม	$x^2 + y^2 = r^2$	$y = \pm\sqrt{r^2 - x^2}$
พาราโบลา (แกน x เป็นแกนสมมาตร)	$y^2 = 4cx$	$x = \frac{y^2}{4c}$
พาราโบลา (แกน y เป็นแกนสมมาตร)	$x^2 = 4cy$	$y = \frac{x^2}{4c}$
ไฮเปอร์โบลา (แกนตามขวางอยู่บนแกน x)	$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$	$y = \pm\sqrt{b^2\left(\frac{x^2}{a^2} - 1\right)}$
ไฮเปอร์โบลา (แกนตามขวางอยู่บนแกน y)	$\frac{y^2}{a^2} - \frac{x^2}{b^2} = 1$	$x = \pm\sqrt{b^2\left(\frac{y^2}{a^2} - 1\right)}$
วงรี (แกนเอกอยู่บนแกน x)	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$	$y = \pm\sqrt{b^2\left(1 - \frac{x^2}{a^2}\right)}$
วงรี (แกนเอกอยู่บนแกน y)	$\frac{x^2}{b^2} + \frac{y^2}{a^2} = 1$	$x = \pm\sqrt{b^2\left(1 - \frac{y^2}{a^2}\right)}$
ทรงกลม	$x^2 + y^2 + z^2 = r^2$	$y = \pm\sqrt{r^2 - x^2 - z^2}$
โปรเจคไทล์	$t = \frac{2u\sin\theta}{g}$ $S_x = u\cos\theta \times t$ $S_y = u\sin\theta \times t - gt^2$	$t = \frac{2u\sin\theta}{g}$ $S_x = u\cos\theta \times t$ $S_y = u\sin\theta \times t - gt^2$

5.2 ปัญหาและแนวทางแก้ไข

ในส่วนของโปรแกรมที่เป็นแบบจำลองของเครื่องบินนั้น สามารถคำนวณได้เฉพาะบนแกน x และ y เท่านั้น โดยที่อีกหนึ่งแกน (z) เป็นค่าคงที่ ซึ่งยังไม่สมจริงเท่าที่ควร เนื่องจากว่าต้องใช้ความรู้ทางด้านการบินที่สูงนอกเหนือไปจากการศึกษาโครงงานนี้ มาเพื่อใช้คำนวณหาค่าเพื่อที่จะนำมาสร้างรูปให้สมจริงยิ่งขึ้น

ในส่วนของโปรแกรมแยกสี RGB นั้น ค่าสีที่ได้ออกมาแสดงออกมาได้เป็นรูปบนพิกัดขนานเท่านั้น ผู้ใช้โปรแกรมไม่สามารถเห็นค่าจริงของสีนั้นได้ เนื่องจากว่า ค่าสีที่ได้ในบางภาพที่ได้ออกมามีหลายค่า ซึ่งไม่สามารถแสดงออกมาได้เพียงพอในพื้นที่หน้าต่างที่จำกัด จะต้องออกแบบการแสดงผลใหม่ให้เหมาะสมในส่วนนี้ สำหรับผู้ที่สนใจ

5.3 สรุปผลการทดลอง

จากการทดลองการวาดรูปบนพิกัดขนานในแบบต่างๆและโปรแกรมแยกสี RGB จากรูปต่างๆ แสดงถึงประสิทธิภาพของโปรแกรมที่สามารถวาดรูปออกมาบนพิกัดขนานและแยกสี RGB จากรูปต่างๆได้ดี และสามารถนำไปใช้ให้เกิดประโยชน์แก่บุคคลที่สนใจศึกษาในเรื่องของ Parallel Coordinate ต่อไปได้ แต่ยังมีข้อจำกัดในเรื่องของการบินและการแสดงผลค่าสีที่ได้ออกมาจากโปรแกรมแยกสี RGB จึงต้องมีการพัฒนาให้ดีขึ้นต่อไป

5.4 ข้อเสนอแนะ

1. ใช้ความรู้ทางด้านการบินในขั้นสูง มาเพื่อพัฒนาโปรแกรมให้สามารถใช้เป็นแบบจำลองที่เสมือนจริงมากยิ่งขึ้น

2. ใช้เทคนิคในการเขียนโปรแกรมขั้นสูงขึ้นไป เพื่อพัฒนาให้โปรแกรมมีประสิทธิภาพในการแสดงผลมากยิ่งขึ้นทั้งในโปรแกรมวาดรูปและโปรแกรมแยกสี

บรรณานุกรม

[1] Christopher V. Jones. "Parallel Coordinate." [Online]. Available: http://catt.bus.okstate.edu/jones98/parallel.html. 2005.

[2] Alfred Inselberg. "Parallel Coordinate: VISUAL Multidimensional Geometry and its Applications; Multidimensional Lines." [Online]. Available: www.math.tau.ac.il/~aiisreal/index_files/lect-pdg/lect-lines.pdf. 2004.

[3] Alfred Inselberg. "Parallel Coordinate: VISUAL Multidimensional Geometry and its Applications; The plane R^2||-coords." [Online]. Available: www.math.tau.ac.il/~aiisreal/index_files/lect-pdg/lect-planes.pdf. 2004.

[4] Wikipedia. "Sphere - Wikipedia, the free encyclopedia." [Online]. Available: http://en.wikipedia.org/wiki/Image:Sphere-wireframe.png. 2006.

[5] Anonymous. [Online]. Available: www.prc.ac.th/newart/webboard/colour08.html

[6] เกษรา จันทรฑีโร. "คณิตศาสตร์ -- สรุปสูตร – ภาคตัดกรวย." [Online]. Available: http://web.ku.ac.th/schoolnet/snet2/knowledge_math/formul1/fomul1.htm. 2005.

[7] นิรุธ อำนวยศิลป์. Visual C++ and MFC Programming เขียนโปรแกรมด์วินโดวส์ด้วย Visual C++ และ MFC. กรุงเทพมหานคร : บริษัท ด่านสุทธาการพิมพ์ จำกัด. 2548.

[8] นิรุธ อำนวยศิลป์. C++ Desing and its applications การออกแบบชุดคำสั่งและการประยุกต์. กรุงเทพมหานคร : บริษัท จ. เจริญการพิมพ์ จำกัด. 2548.

[9] ยุทธนา ลีลาศวัฒนกุล. คู่มือการเขียนโปรแกรมวินโดว์ขั้นสูงด้วย Visual C++ .NET Episode One. กรุงเทพมหานคร : หจก. ไทยเจริญการพิมพ์. 2546.

[10] ยุทธนา ลีลาศวัฒนกุล. คู่มือการเขียนโปรแกรมและใช้งาน Visual C++ .NET ฉบับสมบูรณ์. กรุงเทพมหานคร : บริษัท ด่านสุทธาการพิมพ์ จำกัด. 2546.