การพัฒนาเกมจำลองยานยิง 3 มิติโดยใช้ไดเร็กเอ็กซ์เอนจิน

3D Simulation Shooting Game Development using DirectX Engine

นายเฉลิมพัฒน์ กมลรัตนา

นายธนาวุฒิ เนติสุนทร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2545

การพัฒนาเกมจำลองยานยิง 3 มิติโดยใช้ไดเร็กเอ็กซ์เอนจิน

3D Simulation Shooting Game Development using DirectX Engine

โดย

นายเฉลิมพัฒน์ กมลรัตนา

นายธนาวุฒิ เนติสุนทร

อาจารย์ที่ปรึกษา

ดร. วรวัฒน์ ลิ้มโภคา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2545

ปริญญานิพนธ์ปีการศึกษา 2545

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การพัฒนาเกมจำลองยานยิง 3 มิติโดยใช้ไดเร็กเอ็กซ์เอนจิน

3D Simulation Shooting Game Development using DirectX Engine

ผู้จัดทำ

1. นาย เฉลิมพัฒน์ กมลรัตนา รหัสประจำตัว 42010068

2. นาย ธนาวุฒิ เนติสุนทร รหัสประจำตัว 42010140

อาจารย์ที่ปรึกษา

(ดร. วรวัฒน์ ลิ้มโภคา)

การพัฒนาเกมจำลองยานยิง 3 มิติโดยใช้ไดเร็กเอ็กซ์เอนจิน

นาย เฉลิมพัฒน์ กมลรัตนา 42010068
นาย ธนาวุฒิ เนติสุนทร 42010140
ดร. วรวัฒน์ ลิ้มโภคา อาจารย์ที่ปรึกษา
ปีการศึกษา 2545

บทคัดย่อ

การพัฒนาเกม 3 มิติ เป็นโครงงานที่กล่าวถึง การพัฒนาโปรแกรมคอมพิวเตอร์ประเภทเกม 3 มิติ โดยใช้ไดเร็กเอ็กซ์มาเป็นส่วนประกอบของโปรแกรมในด้านมัลติมีเดีย ซึ่งเริ่มต้นตั้งแต่การพัฒนาเกมเอนจินซึ่งเป็นหัวใจสำคัญสำหรับการสร้างเกม สำหรับการเขียนโปรแกรมเพื่อพัฒนาเกม 3 มิติทั้งหมดนี้อาศัยการโปรแกรมในเชิงวัตถุซึ่งภาษาที่ใช้ในการพัฒนาคือ "ซีพลัสพลัส"

ขอบเขตการทำงานของโครงงานนี้: ในขั้นแรกจะทำการศึกษาถึงสถาปัตยกรรมของไดเร็กเอ็กซ์เพื่อนำมาช่วยในการพัฒนาโปรแกรมเกมในด้านต่าง ๆ อาทิ การแสดงผลทางด้านภาพ, เสียง และการติดต่อกับอุปกรณ์อินพุต รวมถึงศึกษาทฤษฎีพื้นฐานที่จะนำไปสู่การพัฒนาเกมเอนจิน จากนั้นจึงทำการออกแบบวางแนวความคิดโครงสร้างและส่วนประกอบของเกมเอนจิน และเริ่มพัฒนาส่วนประกอบของเอนจินควบคู่ไปด้วยเพื่อศึกษาถึงผลลัพธ์ของวิธีการต่างๆแล้วเลือกใช้อย่างเหมาะสม โดยเริ่มจากด้านกราฟิก, เสียง, อินพุตของเกม และส่วนประกอบอื่นๆที่จำเป็นในตอนเริ่มต้นของการพัฒนาเกมเอนจิน รวมถึงเริ่มทำการสร้างและการจัดการงานด้านโมเดล 3 มิติและ เอนจินในส่วนนี้จะนำมาใช้เป็นพื้นฐานของเกมที่จะพัฒนาขึ้นมา ดังนั้นจะมีสร้าง เอนจินขึ้นมาอีกตัวเพื่อควบคุมเอนจินตัวแรกที่สร้างขึ้น เรียกว่าเอนจินระดับที่ 2

ปริญญานิพนธ์ฉบับนี้ เป็นส่วนหนึ่งของปริญญานิพนธ์ในโครงงานเดียวกันทั้งหมด 4 ฉบับ ในฉบับนี้ได้แสดงในส่วนของการพัฒนาส่วนสำหรับการจัดการกล้อง และสร้างเอนจินในการควบคุมการเคลื่อนไหวของกล้อง โดยเอนจิน มีการเคลื่อนไหวแบบต่างๆให้เลือกใช้งาน

ในขั้นที่สอง นำเอนจินที่พัฒนาจากขั้นแรกมาสร้างเกมแบบจำลองการยิง 3 มิติแบบต่อเนื่อง และทำการพัฒนาเกมเอนจินในส่วนอื่นๆที่เหลือ เพื่อช่วยในการสร้างเกม แล้วจึงเริ่มสร้างโมเดลสามมิติโดยใช้ โปรแกรมสร้างรูปสามมิติ และนำโมเดลสามมิติที่สร้างนั้น ไปใส่ในเอนจินที่พัฒนาขึ้นเพื่อดูผลที่ได้ว่าเป็นยังไง

3D Simulation Shooting Game Development using DirectX Engine

Chalermpat Kamonrattana 42010068
Thanawut Naetisoonthon 42010140
Dr. Worawat Limpoka Advisor

ABSTRACT

3D Games Development is the project that describes the use of DirectX Application Programming Interface (API) as the multimedia component in Developing 3D Game which is starting with game engine development that is the core of game creating. The object oriented programming language in "C++" is used for developing this project.

This project covers: first of all, studying architecture of DirectX and the way to implement them for game development in many ways such as a graphics system, sound effects and interfacing with input devices and also studying the basic theorem road map to game engine development. Then get the concept to design the game engine components and beginning to develop that component for learning the result of many methods to decision the appropriate way at the same time. The programming will begin from graphics, sound, input interfacing and the other components that are necessary in the beginning step of game engine development and also 3D modeling field. And this Engine will use in basic game programming ,so This project will development another engine to use and control first engine (call engine level 2)

The 3D Game project consist of four part, this thesis is one part of all that related to camera and create engine to control the way to move camera .This engine will have many type to move camera to select

After that , use the engine that create to develop Real-time 3D Simulation shooting game and Develop the left engine components for helping develop game then build the 3d model by using 3D Program and put that 3D Model to the create engine to see how it done

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และความร่วมมือจากหลาย ๆ ฝ่ายขอขอบพระคุณอาจารย์วรวัฒน์ ลิ้มโภคา ซึ่งอาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความใส่ใจดูแลเอาใจใส่ ให้คำปรึกษาอย่างใกล้ชิดด้วยดีตลอดมา ทำให้ผู้จัดทำรู้สึกภูมิใจเป็นอย่างมากที่ได้มีโอกาสได้ทำปริญญานิพนธ์นี้

ขอขอบพระคุณบุคคลสำคัญที่สุดที่ทำให้ข้าพเจ้ามีวันนี้ ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ซึ่งได้เลี้ยงดู ให้โอกาสในการศึกษา และคอยให้กำลังใจเสมอมาในทุก ๆ ด้านอันหาที่เปรียบมิได้ นอกจากนี้ขอขอบพระคุณคณาจารย์ทุกท่านที่ประสิทธิประสาทวิชาความรู้ให้แก่ข้าพเจ้าตลอดมาตั้งแต่ได้มาศึกษาเล่าเรียนในสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังจนมีวันนี้

ขอขอบคุณทุกคนในห้องวิจัย OLALA ที่คอยซักถามความคืบหน้า คอยกระตุ้น และคอยเป็นกำลังใจในการทำงานจนสำเร็จได้ และขอขอบคุณห้องวิจัย Multimedia ที่เอื้อเฟื้ออาหารและขนม

สุดท้ายขอขอบคุณเพื่อน ๆ ในแต่ละกลุ่มที่ร่วมพัฒนาเกมด้วยกันในภาควิชาตลอดระยะเวลาเกือบ 1 ปี ที่ให้ความช่วยเหลือสนับสนุนในเรื่องของการโปรแกรมและการทำโมเดล 3 มิติซึ่งทำให้ปริญญานิพนธ์นี้เสร็จสมบูรณ์ได้ด้วยดี

นายเฉลิมพัฒน์ กมลรัตนา

นายธนาวุฒิ เนติสุนทร

สารบัญ

หน้าที่

สารบัญ (ต่อ)

หน้าที่

สารบัญ (ต่อ)

หน้าที่

สารบัญ (ต่อ)

หน้าที่

สารบัญภาพ

หน้าที่

สารบัญภาพ (ต่อ)

หน้าที่

สารบัญตาราง

หน้าที่

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

การพัฒนาทางด้านเกมได้พัฒนาไปอย่างรวดเร็วมาก โดยเฉพาะในด้านเกมคอมพิวเตอร์ จากที่สมัยก่อนทำงานบนดอส (Dos) ไม่เป็นกราฟิกเป็นเท็กซ์โหมด ต่อมาค่อยเป็นกราฟิก และพัฒนาสู่วินโดว์ ซึ่งที่ผ่านมามีผู้ผลิตทำการผลิตเกมออกมามากมาย มีการพัฒนาด้านฮาร์ดแวร์ให้ระบบการแสดงผลสามารถประมวลผลสามมิติได้เพื่อแบ่งเบาภาระของหน่วยประมวลผลกลางของคอมพิวเตอร์ (CPU) ทำให้การแสดงผลมีความเร็วราบรื่นและมีความสวยงาม ไม่ว่าจะเป็นการ์ดแสดงผลสามมิติที่จัดการด้านการแสดงผลสามมิติโดยตรง หรือหน่วยประมวลผลกลางที่พัฒนาคำสั่งต่างๆ มารองรับการประมวลผลข้อมูลสามมิติ ทางด้านซอฟแวร์ก็มีโปรแกรมต่าง ๆ เข้ามาช่วยสร้างสรรค์ภาพสามมิติ ทำให้มีการสร้างเกมในลักษณะสามมิติออกมามากมายเรื่อย ๆ แต่กว่าจะมาเป็นเกมให้ได้เล่นกันจะต้องมีการพัฒนาสร้างเกมซึ่งเครื่องมือในการสร้างเหล่านั้นส่วนหนึ่งก็คือ ไดเร็กเอ็กซ์ (DirectX) และเกมเอนจินต์ (Game Engine)

ไดเร็กเอ็กซ์ เริ่มมาจากที่บริษัทไมโครซอฟต์ได้คิดที่จะพัฒนาไลบรารี่กลางสำหรับติดต่อกับฮาร์ดแวร์และระบบมัลติมีเดีย เพื่อนำมาใช้ในการดึงความสามารถทางฮาร์ดแวร์ต่าง ๆ ได้อย่างเต็มประสิทธิภาพแล้ว จึงได้พัฒนาไดเร็กเอ็กซ์ขึ้นมา โดยทำงานในระบบปฏิบัติการไมโครซอฟต์วินโดว์ (Microsoft Windows) และได้พัฒนาเวอร์ชั่นใหม่ ๆ ออกมาเรื่อย ๆ และสามารถใช้ความสามารถของการ์ดเร่งความเร็วกราฟิกแบบ 3 มิติที่ติดตั้งอยู่บนเครื่องคอมพิวเตอร์และชุดคำสั่งพิเศษของไมโครโปรเซสเซอร์ได้ อีกทั้งยังช่วยให้ผู้พัฒนาเกมไม่ต้องยุ่งเกี่ยวกับการจัดการฮาร์ดแวร์ที่มีอยู่มากมายหลายยี่ห้อที่มีมาตรฐานต่างกันอีกด้วย สาเหตุก็เนื่องจากมาตรฐานของไดเร็กเอ็กซ์สนับสนุนฮาร์ดแวร์เหล่านี้

การพัฒนาของการเขียนเกมด้วยไดเร็กเอ็กซ์ ยังสามารถพัฒนาได้อีกขั้น คือการพัฒนาให้เป็นเกมเอนจิน ซึ่งเป็นการรวบรวมคำสั่งต่าง ๆ ของไดเร็กเอ็กซ์เข้าด้วยกันเป็นฟังก์ชันโดยรวบรวมแบบเฉพาะทาง ดังนั้นผู้พัฒนาเกมส์จะสามารถเรียกใช้งานฟังก์ชันต่างๆได้สะดวก รวดเร็ว และง่ายยิ่งขึ้นกว่าเดิม

เกมเอนจินต์ที่ดีควรจะใช้งานที่ง่าย และผู้ใช้เอนจินควรจะมีความรู้พื้นฐานในเรื่องการทำเกมสามมิติซึ่งเป็นสิ่งสำคัญในการพัฒนาเกมสามมิติให้มีประสิทธิภาพ ไม่ว่าจะเป็นเทคนิคในการคำนวณทางคณิตศาสตร์ ไปจนถึงอัลกอริทึมในการเขียนโปรแกรมกับระบบสามมิติ ดังนั้น ในการพัฒนาเกมสามมิติให้ได้ดียิ่ง การศึกษาให้ได้ลึกและใกล้ชิดอย่างหนึ่งก็คือ เริ่มตั้งแต่การทำเกมเอนจินต์ขึ้นมาเอง และใช้เกมเอนจินต์นั้นในการพัฒนาเกมสามมิติต่อไป

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อศึกษาการทำงานของไดเร็กเอ็กซ์ และการนำไปใช้ในการพัฒนาเกมแอพพลิเคชัน

1.2.2 เพื่อพัฒนาชุดคำสั่งเกมเอนจินสำหรับนำไปพัฒนาเกม

1.2.3 ศึกษาและทำการพัฒนาเอนจินสำหรับการควบคุมมุมกล้องได้อย่างอิสระ

1.2.4 สร้างเกมสามมิติ จากเกมเอนจินต์ที่ได้ออกแบบมาได้

1.3 ขอบเขตของโครงงาน

โครงงานที่ได้จัดทำขึ้นนี้เป็นการรวบรวมชุดคำสั่งของไดเร็กเอ็กซ์ที่มีลักษณะในแนวเดียวกันมารวมเข้าไว้ด้วยกันเป็นเกมเอนจิน โดยเราจะทำการแบ่งฟังก์ชันหลักภายในเกมเอนจินออกเป็นส่วนย่อย เนื่องจากแต่ละส่วนนั้นมีความชัดเจนในการทำงานต่างกันออกไป และแต่ละกลุ่มจะทำการพัฒนาส่วนย่อยต่างๆ นี้แล้วนำมารวมเข้าไว้ด้วยกัน เป็นชุดคำสั่งไลบรารี เพื่อการทำงานในการพัฒนาเกมที่สมบูรณ์

เมื่อพัฒนาเกมเอนจินในระดับแรกเสร็จ แต่ละกลุ่มก็จะนำไปพัฒนาขั้นต่อไปตามแต่ละกลุ่มตามที่วางขอบเขตไว้ ในส่วนของโครงงานชิ้นนี้จะเป็นการพัฒนาในเรื่องของชุดคำสั่งของการควบคุมกล้อง ซึ่งสามารถกำหนดค่าต่างๆ พัฒนาให้เกมมีการควบคุมมุมกล้องได้อย่างอิสระ สามารถกำหนดค่าต่างๆ ของกล้องได้ตามต้องการ

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1. ทักษะและความสามารถในการพัฒนาเกม 3 มิติ

1.4.2. ความรู้ทางคณิตศาสตร์และอัลกอลิทึมในการสร้างเกม เช่น การคำนวณในการหมุนวัตถุ การคำนวณสำหรับการเคลื่อนที่วัตถุ

1.4.3. การออกแบบโปรแกรมเชิงคอมโพเน็นต์

1.4.4. การใช้ไดเร็กเอ็กซ์ในการสร้างแอพพลิเคชันใช้งาน และติดต่อกับอุปกรณ์มัลติมีเดีย

1.4.5. ความรู้ทางภาษา C++ ที่ใช้ในการพัฒนาเกมแอพพลิเคชัน

1.5 วิธีในการดำเนินงาน

1.5.1. ทำการศึกษาและค้นคว้าความรู้ทางด้านการเขียนเกม 3 มิติ และไดเร็กเอ็กซ์ จากหนังสือและอินเตอร์เน็ต

1.5.2. ศึกษาแนวทางการพัฒนาและความรู้ที่ต้องใช้เพิ่มเติมจากความรู้พื้นฐาน

1.5.3. เลือกและศึกษาเครื่องมือที่ใช้ในการพัฒนา

1.5.4. ออกแบบโครงสร้างและแบ่งงานตามกลุ่มตามงานที่ได้รับมอบหมาย

1.5.5. สร้างเกมเอนจินในแต่ละส่วนตามโครงสร้างที่ออกแบบไว้

1.5.6. ทำการทดสอบเกมเอนจินที่ได้สร้างขึ้น

1.5.7. ทำการรวบรวมส่วนต่างๆ ของเกมเอนจินจากแต่ละกลุ่มให้กลายเป็นเกมเอนจินที่สามารถนำไปใช้งานได้

1.5.8. วางแนวทางในการพัฒนาขั้นต่อไปของเอนจินที่ควบคุมเกี่ยวกับกล้องและแนวเกมที่จะพัฒนาให้สอดคล้องกับเอนจิน

1.5.9. ศึกษาและสร้างโมเดล 3 มิติและสร้างภาพกราฟิกที่ต้องใช้งานในเกม

1.5.10. สร้างเกมตามโครงสร้างที่ได้ออกแบบไว้

1.5.11. ทดสอบการทำงานของเกม และหาข้อผิดพลาดของเกมที่พัฒนาขึ้น

1.5.12. ทำการรวบรวมผลทดสอบ วิเคราะห์ปัญหาที่เกิดขึ้น และหาแนวทางในการแก้ปัญหาเพื่อนำมาสรุปเป็นข้อมูลในการพัฒนาขั้นต่อไป

บทที่ 2
แนะนำไดเร็กเอ็กซ์

2.1 ไดเร็กเอ็กซ์คืออะไร

ไดเร็กเอ็กซ์คือชุดของคำสั่ง API (Application Programming Interface) ที่ช่วยให้ผู้พัฒนาสามารถสร้างแอพพลิเคชันที่ทำงานทางด้านมัลติมีเดียได้อย่างสะดวกและรวดเร็ว โดยไม่จำเป็นต้องติดต่อกับฮาร์ดแวร์โดยตรง และสามารถทำงานได้กับอุปกรณ์ทุกตัวที่สนับสนุนการทำงานของไดเร็กเอ็กซ์ และอยู่บนระบบปฏิบัติการ Windows โดยผู้พัฒนาไม่จำเป็นต้องคำนึงถึงความเข้ากันได้กับแอพพลิเคชัน

2.2 จุดมุ่งหมายของไดเร็กเอ็กซ์

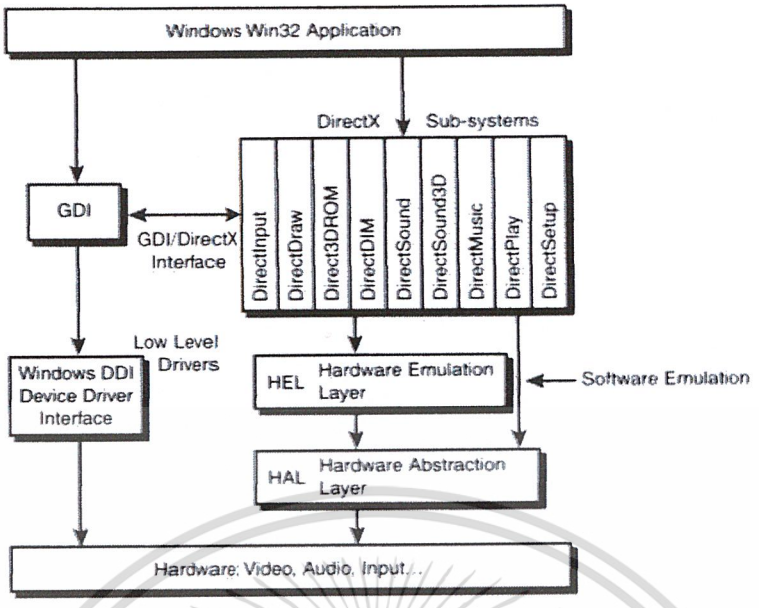
เป้าหมายที่สำคัญที่สุดของไดเร็กเอ็กซ์ คือการจัดหาชุดคำสั่งและเครื่องมือให้แก่นักพัฒนาเพื่อให้นักพัฒนาสามารถพัฒนาแอพพลิเคชันที่ทำงานเกี่ยวกับมัลติมีเดียและเกม ซึ่งใช้เครื่องคอมพิวเตอร์ที่มีระบบปฏิบัติการ Windows เป็นมาตรฐาน โดยไม่จำเป็นต้องตระหนักถึงฮาร์ดแวร์ที่รองรับชุดคำสั่งของไดเร็กเอ็กซ์ ซึ่งจะรวมถึงการสนับสนุนหน้าที่การทำงานทางมัลติมีเดียอย่างหลากหลาย

ไมโครซอฟท์ได้พัฒนาไดเร็กเอ็กซ์ โดยมีเป้าหมายพื้นฐานอยู่ 2 อย่างคือ

- เพื่อให้นักพัฒนาแน่ใจได้ว่าแอพพลิเคชั่นทางมัลติมีเดียนั้นสามารถที่จะทำงานบนเครื่องคอมพิวเตอร์ใดๆ ก็ตามที่มี Windows เป็นระบบปฏิบัติการ โดยไม่จำเป็นต้องใส่ใจถึงฮาร์ดแวร์ที่ติดตั้งอยู่ในเครื่องคอมพิวเตอร์
- เพื่อให้แน่ใจว่าผลิตภัณฑ์ที่ผลิตออกมานั้น จะมีข้อดีของการใช้ความสามารถของฮาร์ดแวร์ที่ติดตั้งอยู่ในเครื่องคอมพิวเตอร์ได้อย่างมีประสิทธิภาพสูงสุด เพื่อบรรลุเป้าหมายของการใช้งานอย่างมีประสิทธิภาพและคุณภาพสูงสุด

2.3 สถาปัตยกรรมของไดเร็กเอ็กซ์

ไดเร็กเอ็กซ์ได้สร้างขึ้นมาบนพื้นฐานของคอมโพเน็นต์ โดยใช้โครงสร้างของ HAL (Hardware Abstraction Layer) ซึ่งสามารถซ่อนลักษณะเฉพาะของดีไวซ์ (Device) ที่เกี่ยวพันอยู่กับฮาร์ดแวร์ และเนื่องจากไดเร็กเอ็กซ์ได้รับการออกแบบให้มีความสามารถในการพัฒนาได้เพิ่มมากขึ้นในอนาคต ดังนั้นจึงสามารถรับรองความสามารถของฮาร์ดแวร์เร่งความเร็วชนิดใหม่ที่เข้ามา โดยใช้การทำงานผ่าน HEL (Hardware Emulation Layer)



รูปที่ 2-1 สถาปัตยกรรมของไดเร็กเอ็กซ์

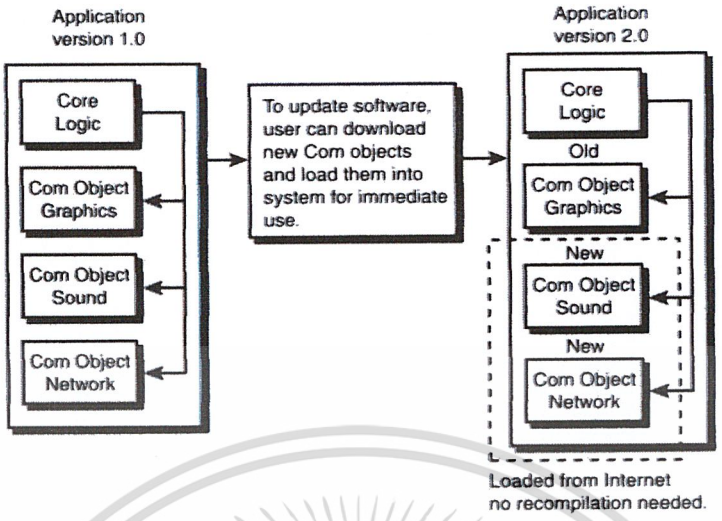
2.3.1 บทบาทของ COM

COM (Component Object Model) ได้กล่าวถึงรายละเอียดเกี่ยวกับวิธีการติดต่อสื่อสารกันระหว่างอ็อบเจ็กต์ โดยข้ามกันระหว่างโพรเซสและข้ามผ่านเน็ตเวิร์ค ระบบที่มีการเรียกใช้ COM ใช้งานจะถูกรับรองเป็นอ็อบเจ็กต์อินสแตนซ์ (Object Instance)

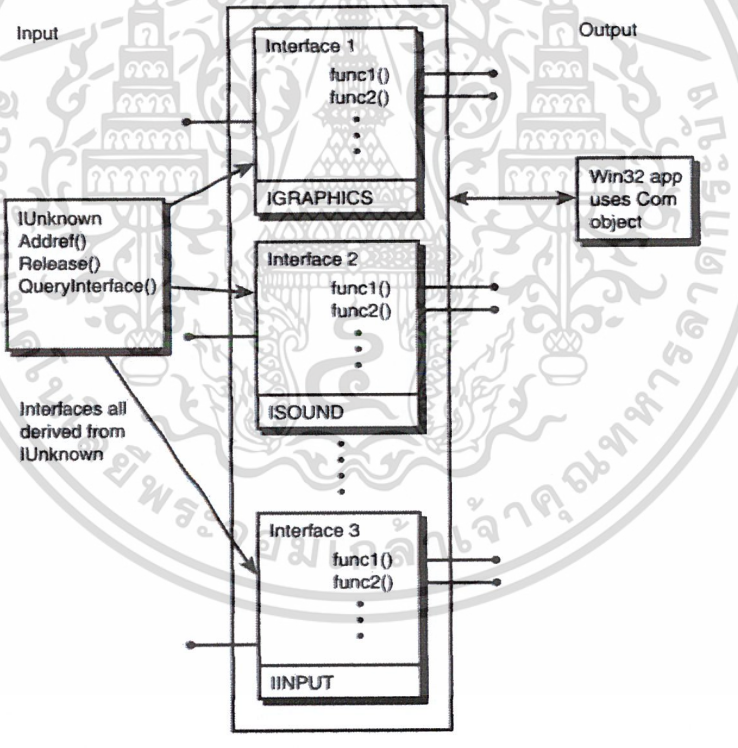
ทุกๆ COM อ็อบเจ็กต์จะต้องยึดติดกันจนเป็นโครงสร้างแบบไบนารี ซึ่งมีความคล้ายคลึงกับโครงสร้างของ Virtual Table ของ C++ ที่ถูกคอมไพล์แล้ว ดังนั้นจะทำให้ทุกภาษาสามารถที่จะสร้าง COM อ็อบเจ็กต์ได้ด้วยการจัดโครงสร้างให้เหมือนกับแบบไบนารีหลังจากการคอมไพล์

ทุกๆ COM อ็อบเจ็กต์นั้นยังจะต้องถูกสืบทอดมาจาก COM อินเตอร์เฟสมาตรฐาน ซึ่งเรียกว่า IUnknown โดย Parent อินเตอร์เฟสจะมีอยู่ด้วยกัน 3 เมธอดคือ AddRef, Delete และ QueryInterface ซึ่ง 2 เมธอดแรกจะจัดการอ็อบเจ็กต์ด้วยการอ้างอิงถึงจำนวน โดยการนับจำนวนของอินสแตนซ์ตั้งแต่เริ่มต้น ส่วนเมธอดสุดท้ายจะจัดการเกี่ยวกับการสอบถามกับ COM อ็อบเจ็กต์ตัวอื่นๆ สำหรับอินเตอร์เฟสที่ต้องการและอินเตอร์เฟสที่มีอยู่ จากนั้นจะคืนค่าพอยเตอร์ที่ชี้ไปที่อินเตอร์เฟสนั้น

COM อ็อบเจ็กต์ที่ทำการร้องขอนั้น เรียกว่า COM ไคลเอนต์และอ็อบเจ็กต์ที่ตอบสนองการร้องขอนั้นคือ COM เซิร์ฟเวอร์ เมื่อไคลเอนต์ได้รับการเชื่อมต่อแล้วไคลเอนต์นั้นสามารถที่จะเรียกเมธอดใดก็ได้ใน COM เซิร์ฟเวอร์นั้น โดยใช้การอ้างอิงของอินเตอร์เฟสพอยเตอร์ ส่วนการเชื่อมต่ออื่นๆ จะออกไปเพื่อทำให้งานของอ็อบเจ็กต์ที่เรียกเพื่อเข้ามาใช้งานง่ายขึ้น ในการเชื่อมต่อนี้จะมีเมธอดที่ทำการแสดงรายชื่อของเมธอดที่สนับสนุน เพื่อที่จะช่วยให้เมธอดสามารถเรียกใช้งานได้ ก่อนที่จะรวมเข้าไว้ด้วยกัน ด้วยเทคนิคนี้จะอนุญาตให้สามารถสร้างชนิดของไลบรารีใดๆ และแอพพลิเคชัน เช่น Visual Basic ซึ่งมีข้อดีในการสร้างคุณสมบัติของ COM โปรแกรมที่มีความทนทานต่อความผิดพลาด



รูปที่ 2-2 ภาพโดยรวมของ COM



รูปที่ 2-3 อินเตอร์เฟสของ COM อ็อบเจ็กต์

2.3.2 COM และไดเร็กเอ็กซ์

ทุกๆ อินเตอร์เฟสของไดเร็กเอ็กซ์นั้นจะทำการสืบทอดมาจาก IUnknown และมีพื้นฐานในการพัฒนาจาก COM ด้วยเช่นกัน

การใช้ COM ของไมโครซอฟท์ทำให้ได้ชุดคอมโพเน็นต์ของไดเร็กเอ็กซ์ที่มีประสิทธิภาพเพิ่มขึ้น ซึ่งสามารถเข้ากันได้กับไดเร็กเอ็กซ์เวอร์ชั่นก่อนๆ ได้อย่างสมบูรณ์ ด้วยความอิสระ ไม่ขึ้นกับภาษาที่ใช้เขียนโปรแกรมของ COM ทำให้นักพัฒนาสามารถที่จะตัดสินใจเลือกภาษาที่ใช้เขียนโปรแกรมตามที่เหมาะสมกับนักพัฒนาเองมากที่สุด ซึ่งอาจเป็นภาษาที่ถนัด และยังทำให้ได้ประสิทธิภาพตามที่ต้องการ นอกจากนั้นนักพัฒนายังสามารถที่จะเลือกภาษาอะไรก็ตามที่ COM สนับสนุน และสามารถพัฒนาสภาพแวดล้อมได้ตามที่นักพัฒนาต้องการ ด้วยโปรแกรมมิ่งโมเดลของ COM โปรแกรมเมอร์จะแน่ใจได้ว่าแม้ว่าผู้ใช้จะอัพเกรดหรือเปลี่ยนติดตั้งฮาร์ดแวร์ใหม่ หรือจะเปลี่ยนไดเร็กเอ็กซ์เวอร์ชั่นใหม่ก็ตาม ซอฟต์แวร์ก็ยังคงจะสามารถใช้งานได้ตามปกติ โดยไม่จำเป็นต้องทำการติดตั้งใหม่

ในความเป็นจริงไดรเวอร์ของอุปกรณ์บน Windows ทุกตัวจะรวมกันเป็น WDM (Windows Driver Model) ซึ่งโครงสร้างภายในก็เป็น COM คอมโพเน็นต์ ดังนั้นจึงมีการโต้ตอบที่ดีกว่าระหว่างไดรเวอร์โมเดลกับหน้าที่ของไดรเวอร์ ในสถาปัตยกรรมของไดเร็กเอ็กซ์นั้นทำให้แน่ใจได้ว่าการพัฒนาสำหรับนักพัฒนาเกมส์จะมีความสะดวกมากขึ้น และลดเวลาในการทดสอบลงด้วย

2.3.3 HAL (Hardware Abstraction Layer)

HAL เป็นส่วนที่อยู่ล่างสุดของไดเร็กเอ็กซ์ ซึ่งประกอบไปด้วยส่วนควบคุมของฮาร์ดแวร์ที่ถูกทำโดยผู้ผลิตฮาร์ดแวร์ ที่จะควบคุมฮาร์ดแวร์โดยตรง โดยชั้นนี้จะให้ประสิทธิภาพอย่างมาก เพราะว่าสามารถที่จะติดต่อกับฮาร์ดแวร์ได้โดยตรง ในการทำงานจริงจะไม่สามารถติดต่อกับ HAL ได้เอง แต่ไดเร็กเอ็กซ์จะทำการจัดการให้โดยอัตโนมัติ

2.3.4 HEL (Hardware Abstraction Layer)

HEL จะอยู่บนชั้น HAL โดยทั่วไปไดเร็กเอ็กซ์จะถูกออกแบบให้สามารถใช้ข้อดีของฮาร์ดแวร์ต่างๆ ได้ แต่ถึงอย่างไรไดเร็กเอ็กซ์ก็ไม่สามารถที่จะทำงานได้กับอุปกรณ์ทุกชนิด ตัวอย่างเช่น การเขียนโค้ดแสดงผลกราฟิก สมมติว่าฮาร์ดแวร์ที่กำลังทำงานอยู่สนับสนุนการหมุน (Rotate) และปรับขนาด (Scale) ภาพบิตแมป ซึ่งการเรียกใช้งานไดเร็กเอ็กซ์ เพื่อที่จะปรับขนาดและหมุนภาพบิตแมปได้นั้นจะต้องมีฮาร์ดแวร์ที่สนับสนุนการปรับขนาดและหมุนภาพ ซึ่งถ้าฮาร์ดแวร์สนับสนุนการทำงาน ก็จะทำงานได้เต็มความสามารถ และใช้งานฮาร์ดแวร์ได้อย่างมีประสิทธิภาพ แต่ถ้าฮาร์ดแวร์ไม่สนับสนุนการปรับขนาดและหมุนภาพ การทำงานของ HEL จะเข้ามาทำงานแทนที่ โดย HEL จะทำการจำลองหน้าที่การทำงานของ HAL ด้วยอัลกอริทึมทางซอฟต์แวร์ โดยเราจะไม่ทราบถึงความแตกต่างเลย แต่ถึงอย่างไรก็ตามโค้ดที่ได้ทำงานนั้น ก็จะทำงานได้ช้าลง เพราะว่าเป็นการจำลองการทำงาน จึงไม่สามารถที่จะทำงานได้เหมือนอย่างฮาร์ดแวร์ร้อยเปอร์เซ็นต์

ไดเร็กเอ็กซ์ไดรเวอร์นี้จะรวมเข้าด้วยกันกับไดเร็กเอ็กซ์ API ผ่านลำดับของบัฟเฟอร์อ็อบเจ็กต์ ผลลัพธ์ส่วนมากจะถูกสร้างโดยคอมโพเน็นต์ ไดเร็กเอ็กซ์ API จะถูกเขียนขึ้นเพื่อตอบสนองบัฟเฟอร์อ็อบเจ็กต์ โดยบัฟเฟอร์นี้จะถูกจัดการ และถูกทำให้มีประสิทธิภาพขึ้นตามฮาร์ดแวร์ที่มีอยู่ โดย 2 เลเยอร์ของ

สถาปัตยกรรมของไดเร็กเอ็กซ์และแปลงไปยังเอาท์พุตดีไวซ์ที่เหมาะสมอย่างเป็นระเบียบ ในระดับของภาษาที่เครื่องคอมพิวเตอร์สามารถเข้าใจได้

คุณสมบัติของการจำลองการทำงานของฮาร์ดแวร์ ก็คือชุดของฟังก์ชันที่ทำการจำลองคำสั่งทาง 3 มิติ โดยใช้ซอฟต์แวร์ ถ้าฟังก์ชันที่ต้องการโดยเกมนั้น ไม่สามารถที่จะเข้าถึงฮาร์ดแวร์ได้โดยผ่าน HAL ไดเร็กเอ็กซ์นั้นจะสลับการทำงานมายัง HEL เพื่อที่จะจำลองการทำงานโดยใช้ซอฟต์แวร์แทน ด้วยคุณสมบัตินี้จะช่วยให้นักพัฒนาสามารถที่จะทดสอบหรือตรวจสอบคุณสมบัติทาง 3 มิติในโหมดของซอฟท์แวร์ได้ด้วยตนเอง

สิ่งเหล่านี้จะทำให้ไดเร็กเอ็กซ์มีความยืดหยุ่นในการใช้งานอย่างมาก ซึ่งจะช่วยสนับสนุนการติดตั้งใช้งานฮาร์ดแวร์ใดๆ ไม่ว่าจะเป็นรุ่นเก่าหรือใหม่ ได้อย่างหลากหลาย เช่น ถ้าเรามีแอพพลิเคชันที่ต้องการคุณสมบัติใหม่ๆ ที่อยู่ในฮาร์ดแวร์รุ่นใหม่ ก็ยังจะสามารถที่จะทำงานในฮาร์ดแวร์รุ่นเก่าได้ โดยผ่านชั้น HEL ของไดเร็กเอ็กซ์เพื่อที่จะจำลองความสามารถใหม่ที่ต้องการ ดังนั้น ถ้าเรายิ่งติดตั้งไดเร็กเอ็กซ์รุ่นใหม่เท่าใด จะช่วยในการทำงานของฮาร์ดแวร์ในคอมพิวเตอร์ ดังนั้นเกมที่ถูกเขียนมาในรูปแบบหนึ่ง จะได้รับข้อดีของคุณสมบัติใหม่ของ ฮาร์ดแวร์ที่ติดตั้งอยู่ในเครื่อง โดยการรวมเข้าด้วยกันกับไดเร็กเอ็กซ์

ตัวอย่าง เช่น ถ้ามีไดเร็กเอ็กซ์ที่เวอร์ชั่นใหม่กว่า ที่ออกแบบมาให้สามารถทำงานกับคอมโพเน็นต์ได้อย่างมีประสิทธิภาพโดยผ่าน MMX ดังนั้นซอฟต์แวร์ทั้งหมดที่ถูกเขียนขึ้นมาก่อนที่ MMX จะออกจะสามารถใช้งาน MMX ได้ แม้ว่าแนวความคิดของ MMX ในสมัยนั้น จะยังไม่เกิดขึ้นในตอนที่ซอฟต์แวร์นั้นถูกเขียนขึ้น ในทางกลับกันซอฟต์แวร์ที่ถูกเขียนขึ้น ในตอนหลังที่มี MMX แล้ว ก็ยังสามารถทำงานได้ในโปรเซสเซอร์รุ่นเก่าก่อนที่มี MMX ได้ โดย HEL ไดรเวอร์นั้นจะสามารถจำลองหน้าที่การทำงานของ MMX ให้มากที่สุดเท่าที่จะเป็นไปได้ โดยผ่านซอฟต์แวร์โดยอนุญาตให้ผู้ใช้ได้ใช้ความสามารถนั้นด้วย

ดังนั้นเหตุผลหลักของข้อดีเหล่านี้ เป็นเพราะว่าการออกแบบสถาปัตยกรรมให้เป็น COM ซึ่งสามารถช่วยให้คอมโพเน็นต์อ็อบเจ็กต์สามารถติดต่อสื่อสารกันได้

2.4 ส่วนประกอบของไดเร็กเอ็กซ์

ไดเร็กเอ็กซ์จะประกอบไปด้วยชุดของคำสั่ง API ซึ่งเตรียมไว้ให้นักพัฒนาสามารถที่จะเข้าถึงการทำงานของฮาร์ดแวร์ที่ติดตั้งอยู่บนเครื่องคอมพิวเตอร์อย่างประสิทธิภาพได้โดยตรง ซึ่งชุดคำสั่ง API เหล่านี้จะทำหน้าที่ควบคุมการทำงานของอุปกรณ์ต่างๆ ซึ่งแบ่งออกเป็นส่วนประกอบต่างๆ ดังต่อไปนี้

- DirectX Graphics
  ทำหน้าที่ดูแลและจัดการการแสดงผลทั้งภาพ 2 มิติและ 3 มิติ ช่วยในการให้แสง จัดวางมุมกล้อง และทำภาพเคลื่อนไหว ซึ่งจะทำการควบคุมอุปกรณ์แสดงผล และดึงความสามารถของฮาร์ดแวร์ให้เต็มประสิทธิภาพ
- DirectX Audio
  ทำหน้าที่ดูแลและจัดการเกี่ยวกับเสียง ไม่ว่าจะเป็นเสียงแบบ 3 มิติหรือเสียงแบบเซอร์ราวด์ นอกจากนั้นยังช่วยในการทำเสียงเอฟเฟ็คท์ต่างๆ อีกด้วย

- DirectX Input

  ทำหน้าที่ดูแลและจัดการเกี่ยวกับอุปกรณ์อินพุตที่ใช้ควบคุมต่างๆ เช่น คีย์บอร์ด เมาส์ หรือจอยสติ๊ก เป็นต้น และยังสามารถทำงานกับอุปกรณ์ที่เป็นแบบป้อนกลับ (Force feedback) ได้อีกด้วย

- DirectX Play

  ทำหน้าที่ดูแลและจัดการเกี่ยวกับการติดต่อระหว่างคอมพิวเตอร์ผ่านระบบเครือข่าย ไม่ว่าจะเป็นระบบแลนหรือผ่านโมเด็ม

- DirectX Show

  ทำหน้าที่ดูแลและจัดการเกี่ยวกับการแสดงผลภาพวีดีโอ

- DirectX Setup

  ทำหน้าที่รวบรวมแอพพลิเคชันต่างๆ ที่พัฒนาขึ้นมาด้วยไดเร็กเอ็กซ์ เพื่อให้สามารถแจกจ่ายไปยังบุคคลอื่นๆ ได้ง่าย

บทที่ 3
ทฤษฎีและหลักการเกม 3 มิติ

ในการพัฒนาโปรแกรมเกม 3 มิติ จำเป็นอย่างยิ่งที่จะต้องมีความเข้าใจถึงทฤษฎีพื้นฐานที่จำเป็นสำหรับนำไปประยุกต์ใช้ โดยเฉพาะทฤษฎีทางด้านคอมพิวเตอร์กราฟิก ซึ่งเป็นพื้นฐานสำหรับการพัฒนาโปรแกรมเกม 3 มิติ

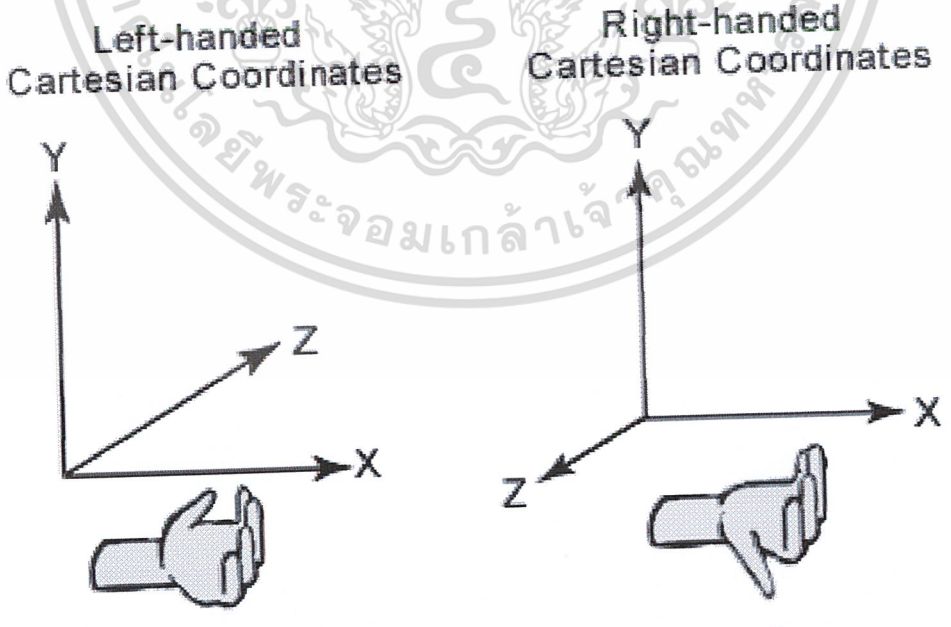
3. ระบบพิกัด 3 มิติ

3.1.1 ระบบพิกัดคาร์ทีเซียน (Cartesian Coordinate System)

เป็นระบบพิกัดที่ทำความเข้าใจได้ง่ายที่สุด เพราะเป็นระบบพิกัดที่คนส่วนใหญ่คุ้นเคย และมีใช้อย่างแพร่หลายที่สุด อีกทั้งยังเหมาะสมในการใช้สำหรับสร้างโปรแกรม ระบบพิกัดคาร์ทีเซียนประกอบด้วยสามแกนที่ตั้งฉากซึ่งกันและกันสำหรับกำหนดพิกัด โดยมักจะตั้งชื่อแกนดังกล่าวให้เป็น X, Y และ Z ระบบพิกัดนี้โดยทั่วไปมักมีการกำหนดแนวแกนได้ 2 แบบคือ

1. ระบบพิกัดคาร์ทีเซียนแบบมือซ้าย (Left-handed Cartesian Coordinate System)
2. ระบบพิกัดคาร์ทีเซียนแบบมือขวา (Right-handed Cartesian Coordinate System)

โดยแสดงได้ดังภาพดังต่อไปนี้

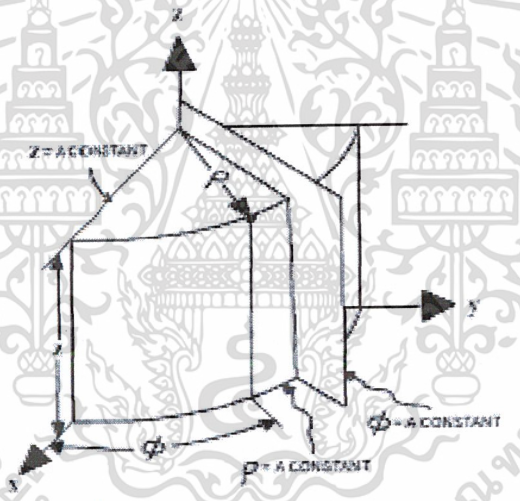


รูปที่ 3-1 แสดงระบบพิกัดคาร์ทีเซียนแบบมือซ้ายและแบบมือขวา

โดยทั่วไป DirectX Graphics API นั้นระบบพิกัดที่ใช้จะเป็นระบบมือซ้าย ในการแสดงผลภาพ 3 มิตินั้นเราจะต้องส่งเวอร์เท็กซ์อาเรย์ที่เราต้องการแสดงผลเข้าไปใน Geometry Pipeline สุดท้ายเราก็จะได้ภาพ 2 มิติออกมาแสดงบนจอภาพ เหตุที่เราต้องแปลงข้อมูลที่เป็น 3 มิติให้เป็น 2 มิตินั้นเพราะหน้าจอของเราไม่สามารถแสดงภาพ 3 มิติตรงๆ ได้จึงต้องทำการส่งข้อมูลเข้า Geometry Pipeline เพื่อให้ได้ผลลัพธ์เป็นภาพ 2 มิติที่สอดคล้องกับข้อมูลนั้นบนมุมมองที่ต้องการ

3.1.2 ระบบพิกัดทรงกระบอก (Cylindrical Coordinate System)

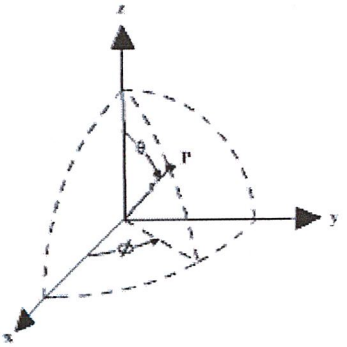
ในระบบพิกัดนี้ เราสามารถระบุพิกัดของจุดแต่ละจุดได้โดยใช้ระยะห่างระหว่างจุดนั้นๆ กับจุดกำเนิด (Origin) ซึ่งแทนด้วยสัญลักษณ์ ρ มุมระหว่างแกนอ้างอิงในแนวระดับ (มักสัมพันธ์กับแกน X และระนาบ XY ในระบบพิกัดคาร์ทีเชียน) กับเส้นตรงที่ลากจากจุดกำเนิดไปยังตำแหน่งของจุดนั้นๆ ในระนาบของจุดกำเนิดกับแกนอ้างอิงในแนวระดับ (ซึ่งสัมพันธ์กับระนาบ XY ในระบบพิกัดคาร์ทีเชียน) ซึ่งแทนด้วยสัญลักษณ์ ϕ และระดับความสูงของจุดนั้นๆ ซึ่งมักแทนด้วย Z ดังแสดงในรูป



รูปที่ 3-2 แสดงระบบพิกัดทรงกระบอก

3.1.3 ระบบพิกัดทรงกลม (Spherical Coordinate System)

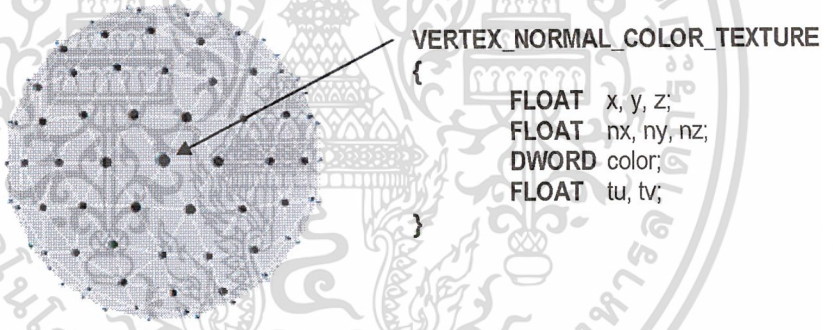
ในระบบพิกัดนี้ เราสามารถระบุพิกัดของจุดแต่ละจุดได้โดยใช้ระยะห่างระหว่างจุดนั้นๆ กับจุดกำเนิด ซึ่งแทนด้วยสัญลักษณ์ r มุมระหว่างแกนอ้างอิงในแนวระดับ (มักสัมพันธ์กับแกน X และระนาบ XY ในระบบพิกัดคาร์ทีเชียน) กับเส้นตรงที่ลากจากจุดกำเนิดไปยังตำแหน่งของจุดนั้นๆ ในระนาบของจุดกำเนิดกับแนวระดับ (ซึ่งสัมพันธ์กับระนาบ XY ในระบบพิกัดคาร์ทีเชียน) ซึ่งแทนด้วยสัญลักษณ์ ϕ และมุมระหว่างแกนตั้งฉากแนวระดับ (แกน Z) กับเส้นตรงที่ลากจากจุดกำเนิดไปยังตำแหน่งของจุดนั้นๆ ดังแสดงในรูป



รูปที่ 3-3 แสดงระบบพิกัดทรงกลม

3.2 เวอร์เท็กซ์ (Vertex)

เวอร์เท็กซ์เป็นจุดพิกัดในระบบ 3 มิติ ซึ่งใช้ในการอ้างอิงตำแหน่ง ซึ่งในการประมวลผลทางด้านกราฟิกนั้นเรามักจะกำหนดคุณสมบัติเพิ่มเติมให้กับเวอร์เท็กซ์ เช่น Color, Normal, Texture Coordinate เป็นต้น ซึ่งคุณสมบัติเหล่านี้เมื่อประกอบกับตำแหน่งของเวอร์เท็กซ์แล้วจะนำไปใช้ในการเก็บข้อมูลที่แทนแบบจำลอง 3 มิติได้ โดยแสดงได้ดังรูปตัวอย่าง

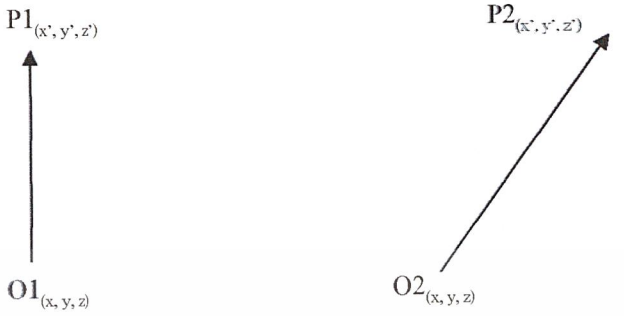


รูปที่ 3-4 แสดงตัวอย่างการนำเวอร์เท็กซ์มาใช้ในการเก็บข้อมูลที่แทนแบบจำลอง 3 มิติ

3.3 เวกเตอร์ (Vector)

เวกเตอร์เป็นปริมาณที่มีทั้งขนาดและทิศทาง เป็นองค์ประกอบทางคณิตศาสตร์ที่มีความสำคัญมากด้านการประยุกต์ใช้ในงานการประมวลผลกราฟิก ซึ่งจะต่างกับปริมาณสเกลาร์ตรงที่เวกเตอร์นั้นจะมีทิศทาง หรือกล่าวได้อีกอย่างหนึ่งว่าเวกเตอร์มีองค์ประกอบมาจากปริมาณสเกลาร์ n ตัว เพื่อแทนขนาดและทิศทางในระบบ n มิติ

เรามักจะแทนเวกเตอร์โดยใช้สัญลักษณ์ $\overrightarrow{OP}$ โดยหมายถึงเวกเตอร์นี้มีทิศทางจากจุด O ไปยังจุด P และมีขนาดเท่ากับระยะห่างระหว่างจุด O และจุด P



รูปที่ 3-5 แสดงตัวอย่างเวกเตอร์

หากเราทำการบวกเวกเตอร์ทั้งสองจะได้ค่าดังนี้

$$R = V1 + V2$$
$$R = (V1_X + V2_X, V1_Y + V2_Y, V1_Z + V2_Z)$$

เมื่อทำการคูณเวกเตอร์ด้วยปริมาณสเกลาร์ ซึ่งทำได้โดยการคูณปริมาณสเกลาร์กับทุกคอมโพเน็นต์ของเวกเตอร์นั้น ๆ จะได้

$$V * s = (V_X * s, V_Y * s, V_Z * s)$$

สำหรับขนาด (Magnitude/length) ของเวกเตอร์ใดๆ นั้น สามารถเขียนแทนด้วย $|V|$ ซึ่งสามารถหาได้โดยใช้กฎของพิทาโกรัส ได้สมการดังนี้

$$|V| = \sqrt{V_X^2 + V_Y^2 + V_Z^2}$$

เวกเตอร์ที่ขนานกัน (Parallel Vector) หมายถึง เวคเตอร์คู่ใด ๆ ที่มีทิศทางเดียวกันหรือทิศทางตรงข้ามกัน

เวกเตอร์ร่วมระนาบ (Coplanar Vector) หมายถึง เวกเตอร์ตั้งแต่ 3 ตัวขึ้นไปที่อยู่ในระนาบเดียวกัน

3.4 เมทริกซ์และทรานส์ฟอร์มเมชัน 3 มิติ

3.4.1 เมทริกซ์ (Matrix)

เมทริกซ์มิติ m × n ประกอบด้วยจำนวนจริงที่เขียนเรียงเป็นแถว (Row) m แถว และเขียนในแนวดิ่ง n หลัก (Column) โดยปิดล้อมจำนวนจริงเหล่านี้ด้วยเครื่องหมาย [ ] หรือ ( ) จำนวนแต่ละจำนวนในเมทริกซ์ เรียกว่า สมาชิกของเมทริกซ์

$\begin{bmatrix} 2 & 1 & 0 \\ 5 & 3 & 2 \end{bmatrix}$ เป็นเมทริกซ์มิติ 2 × 3 $\begin{bmatrix} 1 \\ 0 \\ 2 \\ 4 \end{bmatrix}$ เป็นเมทริกซ์มิติ 4 × 1

ถ้าเมทริกซ์ที่มีจำนวนแถวและจำนวนหลักเท่ากันเท่ากับ n จะเรียกเมทริกซ์นั้นว่า เมทริกซ์จัตุรัสมิติ n และเรียก $a_{11}, a_{12}, ..., a_{nn}$ว่าเป็นสมาชิกในแนวทแยงของ A เมื่อ A เป็นเมทริกซ์จัตุรัสมิติ n

a_{11}, a_{22}, a_{33}เป็นสมาชิกในแนวทแยงของ A หากเมทริกซ์จัตุรัส $A = [a_{ij}]$ ซึ่งมีสมาชิกทุกตัวนอกจากแนวทแยงเป็นศูนย์ทั้งหมด (นั่นคือ $a_{ij} = 0$ ถ้า $I \neq j$) เรียก A ว่าเป็น เมทริกซ์ทแยง (Diagonal Matrix)

ให้ $A = [a_{ij}]_{m \times n}$ และ $B = [b_{ij}]_{m \times n}$ เป็นเมทริกซ์ที่มีมิติเท่ากัน เรียก A เท่ากับ B ก็ต่อเมื่อ $a_{ij} = b_{ij}$ สำหรับ $1 \leq i \leq m, 1 \leq j \leq n$ และเขียนแทนด้วย $A = B$

ถ้า $A= [a_{ij}]_{m \times n}$ และ $B = [b_{ij}]_{m \times n}$ แล้วผลบวกของ A และ B เขียนแทนด้วย A + B หมายถึง $C = [c_{ij}]_{m \times n}$ซึ่ง

$$c_{ij} = a_{ij} + b_{ij} \ (1 \leq I \leq m, 1 \leq j \leq n)$$

ให้ $A = [a_{ij}]_{m \times n}$ เป็นเมทริกซ์ และ k เป็นจำนวนจริงใดๆ แล้ว ผลคูณสเกลาร์ kA จะเป็นเมทริกซ์ $[ka_{ij}]_{m \times n}$ เช่น

ให้ $A = \begin{bmatrix} 2 & 1 & 0 \\ 5 & 3 & 2 \end{bmatrix}$ และ $k = 3$

ดังนั้น $kA = \begin{bmatrix} 3(2) & 3(1) & 3(0) \\ 3(5) & 3(3) & 3(2) \end{bmatrix} = \begin{bmatrix} 6 & 3 & 0 \\ 15 & 9 & 6 \end{bmatrix}$

ให้ $A = [a_{ij}]_{m \times p}$ และ $B = [b_{ij}]_{p \times n}$ แล้ว ผลคูณของ A และ B เขียนแทนด้วย AB หมายถึง

$c = [a_{ij}]_{m \times n}$ โดย $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{i2} + \ldots + a_{ip}b_{ip}$

$$c = \sum_{k=1}^{p} a_{ik}\, b_{kj} \quad (1 \leq i \leq m,\ 1 \leq j \leq n)$$

เช่น $$AB = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} & a_{11}b_{13} + a_{12}b_{23} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} & a_{21}b_{13} + a_{22}b_{23} \end{bmatrix}$$

ผลคูณ AB ไม่มีหรือไม่นิยาม (Undefined) ถ้า A เป็นเมทริกซ์ขนาด $m \times p$ และ B เป็นเมทริกซ์ขนาด $q \times n$ เมื่อ $p \neq q$

การคูณเมทริกซ์นั้น ไม่มีคุณสมบัติการสลับที่ซึ่งหมายถึง $A \times B \neq B \times A$ เมื่อ A และ B เป็นเมทริกซ์จัตุรัส

เมทริกซ์เอกลักษณ์ (Identity Matrix) คือ เมทริกซ์จัตุรัสที่มีสมาชิกในแนวทแยงเป็น 1 ทั้งหมด ซึ่งเขียนแทนด้วย I หรือ I_n แทนเมทริกซ์เอกลักษณ์มิติ n เช่น

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

ถ้า A เป็นเมทริกซ์มิติ $m \times n$ แล้วจะได้ว่า $AI_n = A$ และ $I_m A = A$

B เป็นอินเวิร์สการคูณของเมทริกซ์ A (B เป็นอินเวิร์สของ A) เมื่อ B เป็นเมทริกซ์ซึ่ง $AB = BA = I$ โดยที่ A เป็นเมทริกซ์จัตุรัสใดๆ

ให้ A เป็นเมทริกซ์จัตุรัสมิติ n จะกล่าวว่า A มีตัวผกผัน (Invertible) หรือ A เป็นเมทริกซ์ซึ่งมิใช่เอกฐาน (Non-Singular Matrix) ถ้าสามารถหาเมทริกซ์จัตุรัส B ได้ ซึ่งทำให้

$$AB = I_n = BA$$

และเรียกเมทริกซ์ B ว่าเป็นอินเวิร์สการคูณ ของ A เขียนแทนด้วยสัญลักษณ์ A^{-1} นั่นคือ ถ้า A เป็นเมทริกซ์ซึ่งมิใช่เมทริกซ์เอกฐานมิติ n แล้ว จะได้ว่า

$$AA^{-1} = A^{-1}A = I_n$$

และถ้า A ไม่มีอินเวิร์ส แล้วจะเรียก A ว่าเป็นเมทริกซ์เอกฐาน (Singular Matrix)

3.4.2 Translation

เป็นการเคลื่อนย้ายตำแหน่งของวัตถุ ซึ่งใช้ Translation Matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix}$$

Translation Matrix

3.4.3 Rotation

เป็นการหมุนวัตถุรอบแกน X, Y หรือ Z ซึ่งใช้ Rotation Matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation Around X Axis Matrix

$$\begin{bmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation Around Y Axis Matrix

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation Around Z Axis Matrix

3.4.4 Scaling

เป็นการย่อหรือขยายวัตถุ ซึ่งใช้ Scaling Matrix

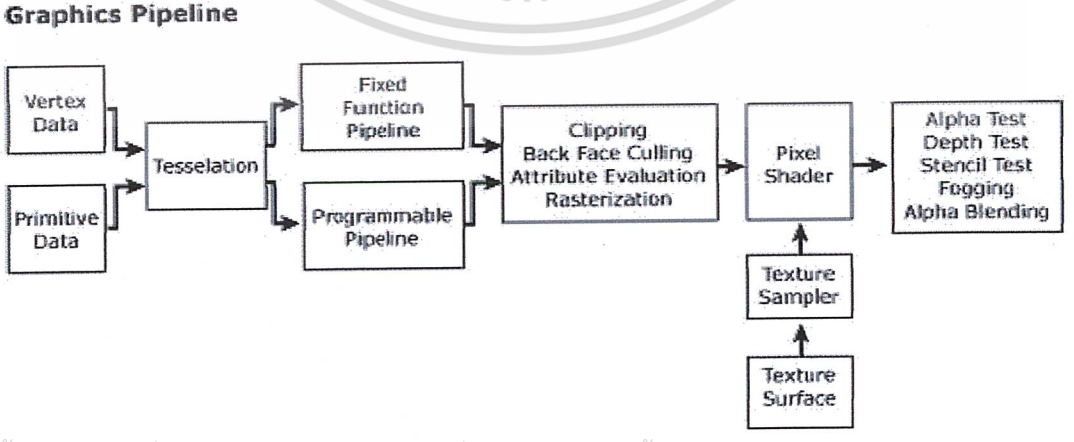
$$\begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Scaling Matrix

3.5 การแสดงผลภาพ 3 มิติด้วย Direct3D

3.5.1 สถาปัตยกรรมของ Direct3D

สถาปัตยกรรมของ Direct3D ประกอบด้วยหน้าที่การทำงานต่างๆ ซึ่งจะรวมกันทำงานแบบ Pipeline ดังที่แสดงตามรูป



รูปที่ 3-6 รูปแสดง Pipeline การทำงานของ Direct3D

3.5.2 ทรานส์ฟอร์มเมชัน (Transformation)

3.5.2.1 การแปรไปสู่พิกัดเวิลด์ (World Transformation)

ขั้นตอนนี้จะเป็นขั้นตอนในการแปลง Local Coordinate ไปเป็น World Coordinate ซึ่งจะเป็นตำแหน่งจริงๆ ในปริภูมิ 3 มิติ ซึ่งวิธีการนั้นเราจะใช้การคูณตำแหน่งในเวอร์เท็กซ์ด้วย Transformation Matrix ต่าง ๆ เพื่อให้ได้ตำแหน่งที่เราต้องการ

ในการแปลงเวอร์เท็กซ์จาก Local Coordinate ให้เป็น World Coordinate นั้นในบางครั้งถ้าการแปลงของเรามีความซับซ้อนมากเราอาจจะต้องใช้เมทริกซ์หลายตัวในการแปลงคูณต่อกันไป (Matrix Concatenation) เพื่อสร้างเมทริกซ์สุดท้ายสำหรับใช้เป็น Transformation Matrix ก่อนแล้วจึงจะเอามาคูณกับตำแหน่งในเวอร์เท็กซ์จริงเพื่อให้ได้ผลลัพธ์สุดท้ายก็ได้

การคูณ Matrix กับตำแหน่งของ Vertex จะเป็นดังต่อไปนี้

$$[x' \; y' \; z' \; 1] = [x \; y \; z \; 1] \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix}$$

$$x' = (x \times M_{11}) + (y \times M_{21}) + (z \times M_{31}) + (1 \times M_{41})$$

$$y' = (x \times M_{12}) + (y \times M_{22}) + (z \times M_{32}) + (1 \times M_{42})$$

$$z' = (x \times M_{13}) + (y \times M_{23}) + (z \times M_{33}) + (1 \times M_{43})$$

จากรูป x, y, z คือ ตำแหน่งของเวอร์เท็กซ์พอคูณกับเมทริกซ์ดังรูปแล้ว จะได้ผลลัพธ์ เป็น x', y', z' ซึ่งเป็นผลลัพธ์ของการแปลงพิกัด

3.5.2.2 การแปรไปสู่พิกัดวิว (View Transformation)

ขั้นตอนนี้ใช้สำหรับแปลง World Coordinate ให้สอดคล้องกับการเห็นของผู้สังเกต เนื่องจาก World Coordinate บอกเพียงตำแหน่งจริงๆ ในพิกัด 3 มิติเท่านั้น ยังไม่มีการอ้างอิงจากผู้สังเกตเลย ดังนั้นจึงต้องมีการบอก ว่าผู้สังเกตอยู่ที่ไหน และมองไปทางไหน ขั้นตอนนี้มักใช้ในการปรับมุมมองของกล้อง โดยการนำวิวเมทริกซ์มาคูณกับ World Coordinate จากการแปรไปสู่พิกัดเวิลด์

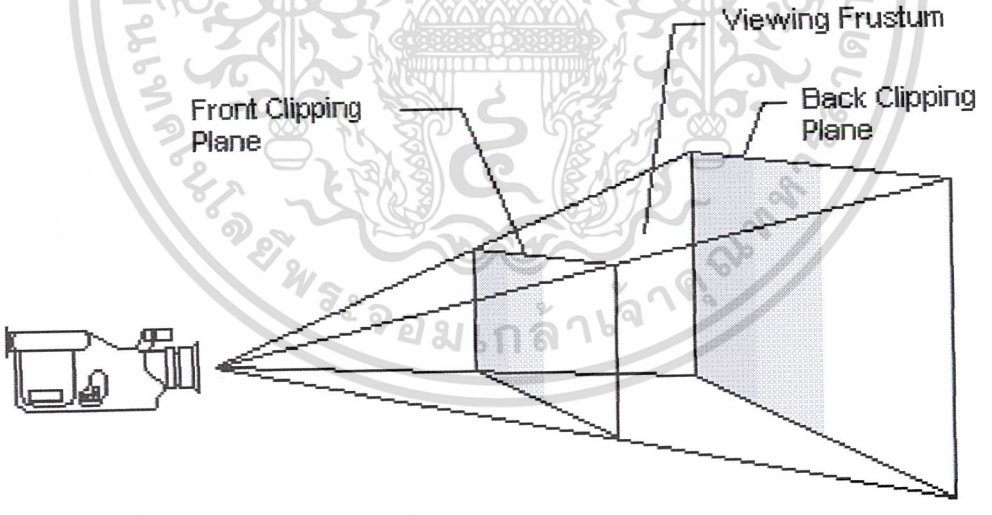
$$\begin{bmatrix} u_x & v_x & n_x & 0 \\ u_y & v_y & n_y & 0 \\ u_z & v_z & n_z & 0 \\ -(u \cdot c) & -(v \cdot c) & -(n \cdot c) & 1 \end{bmatrix}$$

จากรูปวิวเมทริกซ์ประกอบด้วยเวกเตอร์ 3 ตัวบอกทิศทางของผู้สังเกตว่าหันหน้าไปทางใด ซึ่งมีเวกเตอร์ u, v, n บอกถึง Up, Right, และ View Direction ตามลำดับ และเวกเตอร์ c ซึ่งบอกตำแหน่งของผู้สังเกตในพิกัด World

วิวเมทริกซ์นั้นให้ข้อมูลที่จำเป็นในการปรับมุมมองของกล้องในเกมเอนจินได้ ซึ่งใช้ในการบอกตำแหน่งและทิศทางของกล้อง

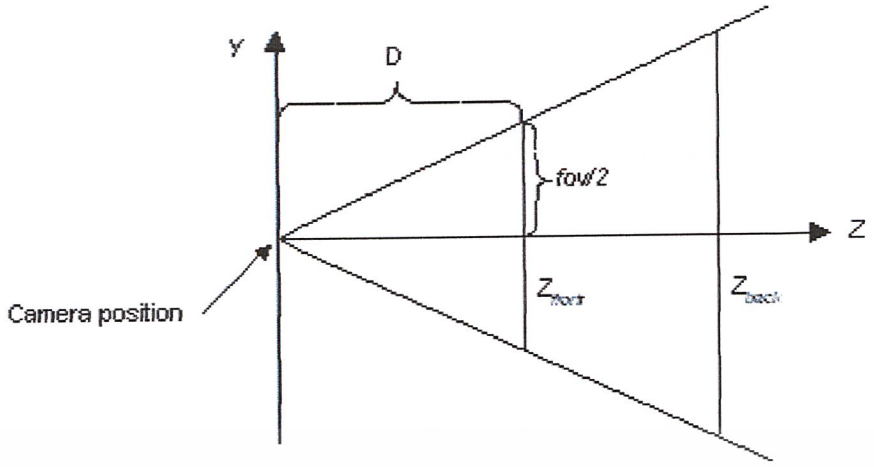
3.5.2.3 การแปรไปสู่พิกัดโปรเจ็กต์ชัน (Projection Transformation)

เมื่อเราได้ตำแหน่งซึ่งอ้างอิงกับผู้สังเกตเรียบร้อยแล้ว ขั้นตอนต่อไปก็คือการแปลง View Coordinate มาเป็นพิกัด 2 มิติ ซึ่งสามารถนำมาวาดได้จริงๆ บนจอภาพ ซึ่งโดยทั่วไปแล้วเราต้องการให้ได้ภาพที่สมจริง คือ วัตถุที่อยู่ไกลจากสายตาจะมีขนาดเล็กลง และวัตถุที่อยู่ใกล้สายตาจะมีขนาดใหญ่ขึ้น ซึ่งเรียกว่า Perspective Projection Transformation ซึ่งเป็นการสร้างภาพฉายมาตกกระทบฉาก คล้ายๆ กับการฉายภาพจากโปรเจคเตอร์

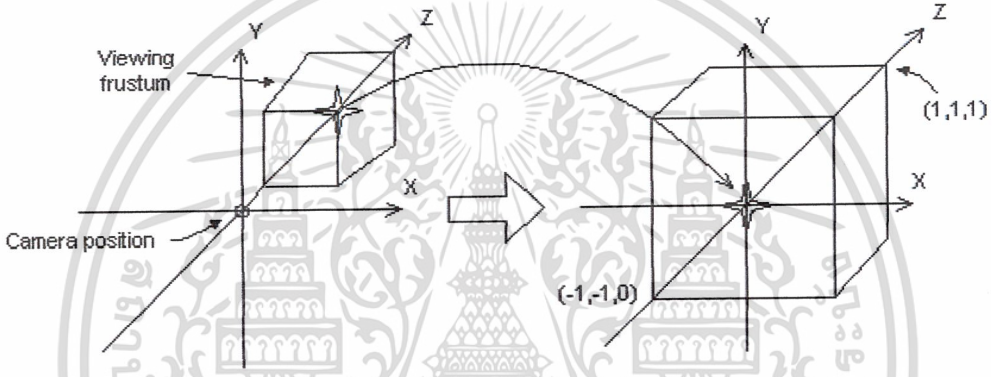


รูปที่ 3-7 Viewing Frustum

วัตถุที่อยู่ใน Viewing Frustum จะถูกนำมาแสดงผล เนื่องจากเป็นส่วนที่จอภาพสามารถเห็นได้ ส่วนวัตถุที่อยู่ภายนอก Viewing Frustum จะถูกตัดออกระหว่างการประมวลผลก่อนที่จะนำไปแสดงผล



รูปที่ 3-8 Viewing Frustum มองจากแนวแกน X



รูปที่ 3-9 การแปลงจาก Viewing Frustum (จากภาพด้านซ้าย) ไปเป็น Cuboid shape (จากภาพด้านขวา) ซึ่งทำให้วัตถุที่อยู่ไกลมีขนาดเล็กลง และวัตถุที่อยู่ใกล้มีขนาดใหญ่ขึ้นตามสัดส่วนของระยะห่างจากสายตาผู้สังเกต

เราสามารถทำ Perspective Projection Transformation ได้โดยการใช้ Perspective Projection Matrix ดังนี้

$$\begin{bmatrix} w & 0 & 0 & 0 \\ 0 & h & 0 & 0 \\ 0 & 0 & Q & 1 \\ 0 & 0 & -QZ_n & 0 \end{bmatrix}$$

$$w = \cot\left(fov_w/2\right)$$

$$h = \cot\left(fov_h/2\right)$$

$$Q = \frac{Z_f}{Z_f - Z_n}$$

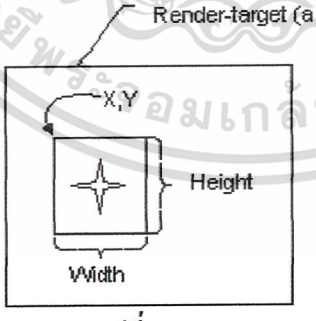
โดยทั่วไปแล้วกราฟิกเอนจินนั้นมักจะมีฟังก์ชันที่ใช้สำหรับทำการคำนวณหา Perspective Projection Matrix โดยให้เรากำหนดค่า Field of View (fov) ตามแนวความสูงของฉาก (โดยทั่วไปมักเป็นแกน Y) ค่า Aspect Ratio (w/h) ค่า Near Clipping Plane (Z_n) และค่า Far Clipping Plane (Z_f)

นอกจากการทำ Perspective Projection Transformation แล้วยังมีการทำ Projection Transformation แบบอื่นๆ อีก เช่น Isometric Projection Transformation, Oblique Projection Transormation เป็นต้น

3.5.3 Clipping and Viewport Scaling

Viewport ในที่นี้หมายถึงพื้นที่สี่เหลี่ยมผืนผ้าที่ใช้ในการฉายภาพที่เกิดจากการแปรไปสู่พิกัดโปรเจ็กต์ชัน ซึ่งเป็นขั้นตอนสุดท้ายใน Geometry Pipeline และทำให้ได้ภาพฉายตกลงบน Viewport เพื่อทำกระบวนการ Rasterization ต่อไปในการสร้างภาพเสมือนจริงบนหน้าจอ

Viewport อาจหมายถึงหน้าจอทั้งหน้าจอหรือบางส่วนของหน้าจอก็ได้ ทั้งนี้ขึ้นอยู่กับโปรแกรม ซึ่งโดยทั่วไปแล้วเกมมักจะแสดงผลแบบเต็มจอ แต่การใช้ Viewport อาจมีประโยชน์เมื่อเราต้องการการแสดงผล 3 มิติหลายหน้าต่าง



รูปที่ 3-10 Direct3D Viewport

โดยทั่วไปแล้วเราจะทำการขริบ (Clipping) สิ่งที่มองไม่เห็นบนหน้าจอออกไป ในกรณีที่ Viewport แสดงถึงบางส่วนของหน้าจอนั้น เราจะขริบส่วนของภาพที่อยู่นอกกรอบ Viewport ออกไป ทั้งนี้รวมถึงการขริบในแนวแกน Z ด้วย (เราจะตัดทอนส่วนของภาพที่อยู่ไกลเกิน Viewport ที่กำหนด หรือ ส่วนของภาพที่อยู่ด้านหลัง Near Clipping Plane ของ Viewing Frustum ออก)

กระบวนการนี้โดยมากมักจัดการโดยกราฟิกเอนจินโดยเราเพียงแค่กำหนดค่าขอบเขตของ Viewport และระยะการตัดทอนตามแนวแกน Z ก็เพียงพอแล้ว

บทที่ 4
เกมเอนจิน

4.1 เกมเอนจินคืออะไร

เกมเอนจินคือเครื่องมือที่ช่วยให้ผู้พัฒนาสามารถสร้างเกมได้สะดวก และรวดเร็วมากขึ้น ซึ่งช่วยดูแลและจัดการในเรื่องของการแสดงผล การควบคุมอุปกรณ์อินพุต เสียงเพลง เสียงเอฟเฟ็คต์ต่างๆ และการติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์ผ่านระบบเครือข่าย โดยไม่จำเป็นต้องศึกษาการทำงานอย่างละเอียด นอกจากนั้นเกมเอนจินยังจะช่วยให้ผู้พัฒนาไม่ต้องเสียเวลาในการพัฒนาเกม เพราะเกมเอนจินจะช่วยจัดการการทำงานบางส่วนให้กับผู้พัฒนาเกมแล้ว เช่น การทำงานในส่วนแสดงผลอาจใช้เพียงแค่คำสั่งเดียวก็สามารถทำงานตามที่ต้องการได้แล้ว

4.2 รูปแบบของเกมเอนจิน

เกมเอนจินโดยทั่วไป จะสามารถแบ่งตามลักษณะของรูปแบบการทำงานของเกมเอนจินได้ 2 ชนิด คือ

4.2.1 เกมเอนจินแบบไลบรารี (Library Game Engine)

จะเป็นเกมเอนจินแบบที่จะรวบรวมชุดคำสั่งที่จำเป็นในการพัฒนาเกมเข้าไว้ด้วยกัน โดยจะแบ่งการทำงานออกเป็นส่วนๆ ตามการทำงาน ซึ่งในการใช้งานจะต้องทำการเพิ่มส่วนของเกมเอนจินเข้าไปรวมกับส่วนของโปรแกรม เพื่อให้ส่วนของโปรแกรมสามารถเรียกใช้งานชุดคำสั่งที่เกมเอนจินได้จัดเตรียมไว้ ซึ่งลักษณะของเกมเอนจินแบบไลบรารีจะสามารถแบ่งได้ออกเป็น 2 ลักษณะ คือ

4.2.1.1 สเตติกไลบรารี

ลักษณะของเกมเอนจินแบบนี้ คือ เวลาใช้งานเกมเอนจินจะรวมส่วนของเกมเอนจินเข้าไปกับส่วนของเกม ดังนั้นถ้ามีการเปลี่ยนแปลงส่วนใดส่วนหนึ่งของเกมเอนจินนั้นจะต้องทำการคอมไพล์ส่วนของเกมใหม่ เพื่อที่จะปรับปรุงการทำงานส่วนของเกมเอนจิน และการใช้งานเกมเอนจินลักษณะนี้จะทำให้ส่วนของเกมมีขนาดที่ใหญ่ เพราะรวมเอาส่วนของเกมเอนจินเข้าไปในส่วนของเกมด้วย

4.2.1.2 ไดนามิกไลบรารี

ลักษณะของเกมเอนจินแบบนี้ คือ เวลาใช้งานจะทำการรวมเข้ากับส่วนของเกม และเวลาเกมเริ่มทำงาน เมื่อใดที่ส่วนของเกมต้องการใช้งานเกมเอนจิน ก็จะทำการไปเรียกส่วนการทำงานของเกมเอนจินขึ้นมาใช้งานในขณะนั้น ดังนั้นเมื่อทำการเปลี่ยนแปลงส่วนใดส่วนหนึ่งของเกมเอนจิน ก็ไม่จำเป็นต้องคอมไพล์ส่วนของเกมใหม่ จึงทำให้ง่ายในการปรับปรุงการทำงานส่วนของเกมเอนจิน และการใช้งานเกม

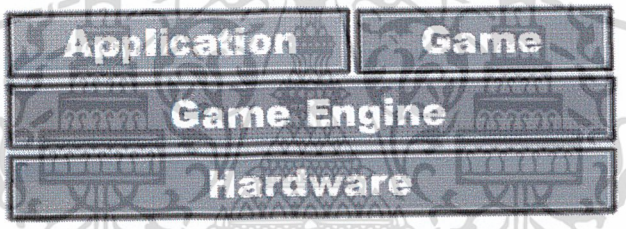
เอนจินลักษณะนี้ส่วนของเกมจะมีขนาดเล็ก เพราะจะแยกส่วนของเกมเอนจินออกไปเป็นอีกส่วนนึง ซึ่งจะไม่เข้าไปรวมกับส่วนของเกม และเป็นลักษณะที่แพร่หลายในการใช้งาน

4.2.2 เกมเอนจินแบบกราฟิก (GUI Game Engine)

เกมเอนจินแบบกราฟิกนั้นจะมีลักษณะเป็นแอพพลิเคชันที่ให้ผู้พัฒนาเกมสามารถสร้างเกมได้โดยไม่จำเป็นต้องรู้ถึงวิธีในการเขียนโปรแกรม และสามารถพัฒนาเกมออกมาออกมาได้ในระยะเวลาอันสั้น ผู้พัฒนาเกมเพียงแค่กำหนดรูปแบบของเกม เนื้อเรื่องของเกมที่ต้องการ จากนั้นใช้เกมเอนจินในการกำหนดส่วนต่างๆ ของเกมให้เป็นไปตามรูปแบบที่กำหนดไว้ จากนั้นก็สามารถทำงานได้แล้ว

4.3 ส่วนประกอบของเกมเอนจินที่พัฒนา

เกมเอนจินที่พัฒนานั้นจะมีลักษณะการทำงาน ซึ่งรวบรวมขึ้นมาเป็นชุดคำสั่ง มีการใช้งานที่ง่าย ซึ่งเกมเอนจินที่พัฒนานี้พัฒนาโดยใช้ไดเร็กเอ็กซ์เป็นพื้นฐาน จะมีส่วนประกอบต่างๆ ซึ่งจะแบ่งตามลักษณะของการทำงาน โดยจะแบ่งออกเป็นส่วนประกอบหลักๆ ดังนี้



รูปที่ 4-1 แสดงระดับของเกมเอนจิน

4.3.1 เกมเอนจินส่วนแอพพลิเคชัน

เกมเอนจินส่วนแอพพลิเคชันเป็นส่วนเริ่มต้นและควบคุมการทำงานของโปรแกรมเกม โดยจะควบคุมกระบวนการทำงานของเกม

4.3.2 เกมเอนจินส่วนกราฟิก

เกมเอนจินส่วนกราฟิกจะช่วยในการแสดงผล การสลับหน้าจอ การให้แสง การโหลดโมเดล กำหนดตำแหน่งกล้อง การทำงานกับวัตถุ 3 มิติ

4.3.3 เกมเอนจินส่วนอินพุต

เกมเอนจินส่วนอินพุตเป็นส่วนติดต่อกับอุปกรณ์อินพุตต่างๆ เช่น คีย์บอร์ด เมาส์ จอยสติ๊ก ซึ่งสามารถที่จะควบคุมการทำงานของอุปกรณ์ต่างๆ

4.3.4 เกมเอนจินส่วนเสียง

เกมเอนจินส่วนเสียงจะดูแลการทำงานในการเอาท์พุตเสียงออกลำโพง และสามารถเล่นเสียงได้หลายช่องทาง ใส่เสียงเอฟเฟ็คต์ และเสียงแบบ 3 มิติ

4.3.5 เกมเอนจินส่วนเน็ตเวิร์ค

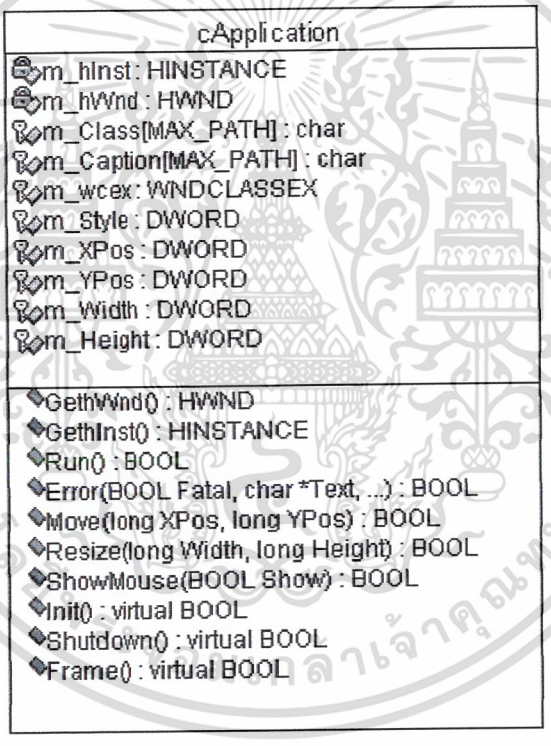
เกมเอนจินส่วนเน็ตเวิร์คจะจัดการติดต่อกับการ์ดเน็ตเวิร์ค โดยจะควบคุมกระบวนการทำงานในการติดต่อสื่อสารกันระหว่างเครื่องคอมพิวเตอร์

บทที่ 5
เอนจินส่วนแอพพลิเคชัน

เอนจินส่วนแอพพลิเคชันเป็นส่วนเริ่มต้นและควบคุมการทำงานของโปรแกรมเกม โดยจะควบคุมกระบวนการทำงานของเกม โดยจะเป็นตัวจัดการสร้างหน้าต่างขึ้นมาให้ โดยมีคลาสที่เกี่ยวข้องดังนี้

5.1 คลาส cApplication

เป็นคลาสที่สร้างและควบคุมการทำงานของโปรแกรม คลาสนี้จะทำการ Register วินโดว์คลาสและสร้างหน้าต่างขึ้นมาให้ และคอยควบคุมดูแลเมสเสจของแอพพลิเคชันที่เราสร้างขึ้นมา



รูปที่ 5-1 คลาสไดอะแกรมของ cApplication

การใช้งานคลาสนี้จำเป็นต้องทำการสืบทอดมาเป็นคลาสใหม่ และจะมีอินสแตนซ์ของคลาสนี้ได้เพียงอินสแตนซ์เดียว เพราะว่าคลาสนี้จะเป็นคลาสหลักในการสร้างแอพพลิเคชันโปรแกรม ซึ่งจะกำหนดขนาดของหน้าต่าง ตำแหน่งของหน้าต่าง ซึ่งจะกำหนดใน Constructor ของคลาสที่สืบทอดจากคลาส cApplication และยังทำการ Register วินโดว์คลาสและทำการสร้างหน้าต่างของแอพพลิเคชันขึ้นมา ถ้าต้องการเปลี่ยนขนาดของหน้าต่างภายหลัง จะใช้เมธอด Resize() และถ้าต้องการย้ายตำแหน่งของหน้าต่างก็จะใช้เมธอด Move() ไปยังตำแหน่งที่ต้องการ

ในการสืบทอดคลาส cApplication นั้น เราจะทำการ Override เมธอดหลักๆ 3 เมธอด คือ Init() ซึ่งจะถูกเรียกทำงานโดยอัตโนมัติ ซึ่งจะถูกเรียกเป็นเมธอดแรกหลังจากทำงานในส่วน Constructor แล้ว เมธอดถัดมาจะเป็นเมธอด Shutdown() ซึ่งจะเป็นเมธอดสุดท้ายที่ถูกเรียกหลังจากเลิกใช้งานแอพพลิเคชัน โดยส่วนใหญ่กำหนดการทำงานในส่วนของเมธอด Init() จะทำเพื่อกำหนดค่าเริ่มต้น หรือทำการจอง ทรัพยากรที่ต้องการก่อนการใช้งานแอพพลิเคชัน ส่วนเมธอด Shutdown() จะทำเพื่อคืนทรัพยากรที่ทำการจองไว้คืนแก่ระบบ ส่วนเมธอดสุดท้ายคือ Frame() จะเป็นเมธอดที่ถูกเรียกทุกครั้งในการทำงาน ซึ่งจะถูกเรียกใช้งานเมื่อไม่มีเมสเสจเข้ามา โดยจะวนทำงานไปจนกระทั่งผู้ใช้ยกเลิกการทำงานของแอพพลิเคชัน เมธอดนี้จะถูกเรียกใช้ภายในเมธอด Run() ซึ่งอยู่ภายในคลาส cApplication ซึ่งส่วนใหญ่การทำงานของเมธอด Frame() นั้นจะถูกใช้งานเพื่อทำการเรนเดอร์ภาพ 3 มิติ และแสดงผลกราฟิกต่างๆ

นอกจากนั้นยังสามารถที่จะกำหนดการประมวลผลเมสเสจของแอพพลิเคชันได้โดยการ Override เมธอด MsgProc() ซึ่งจะเป็นเมธอดในการจัดการเมสเสจ โดยผู้ใช้สามารถกำหนดการทำงานของเมสเสจได้ตามที่ผู้ใช้ต้องการ

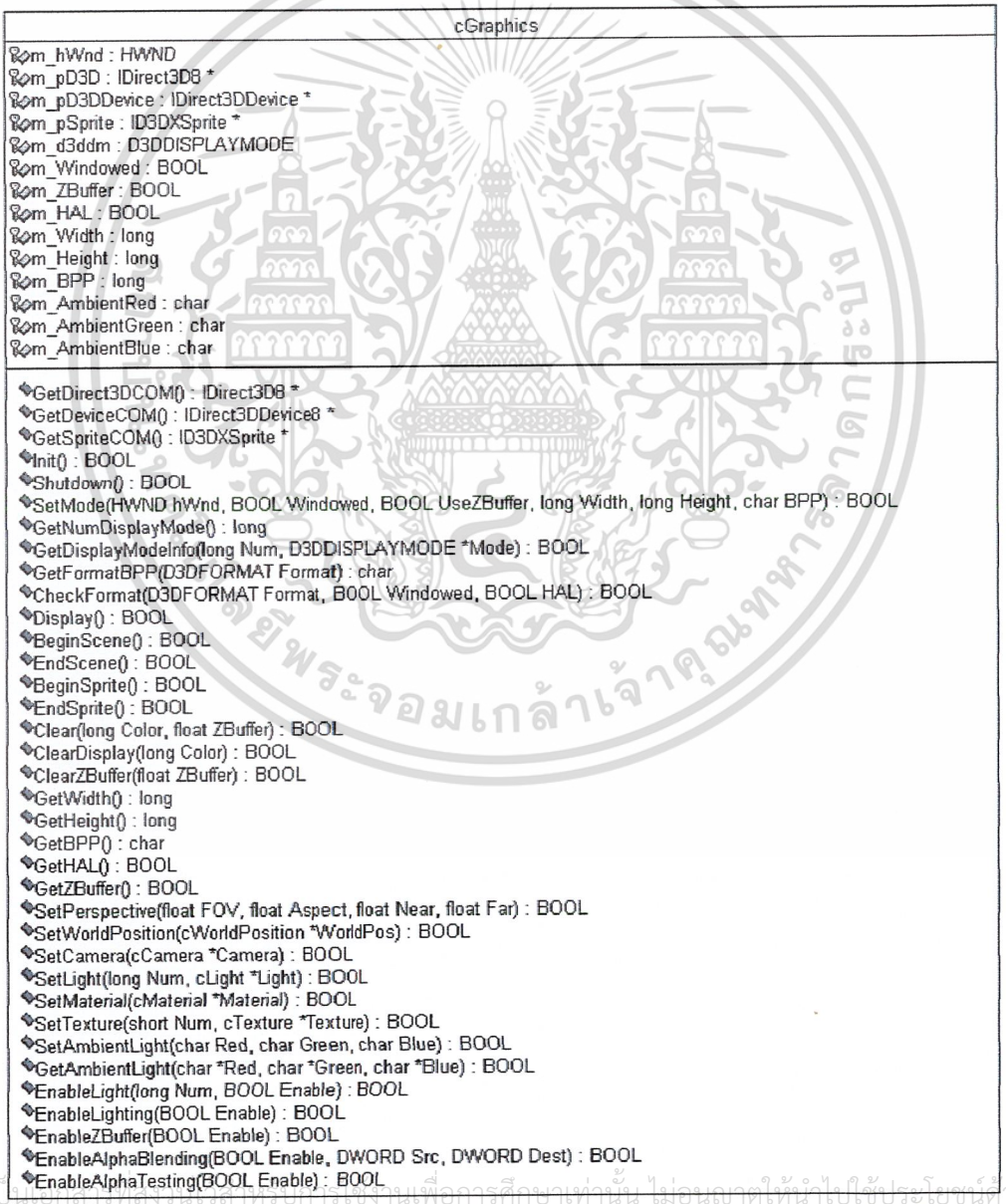
การทำงานของคลาสที่สืบทอดมาจากคลาส cApplication นั้นจะมีแค่เพียงการสร้างอินสแตนซ์ของคลาสที่สืบทอดมา และทำการเรียกเมธอด Run() เพื่อเริ่มการทำงานของแอพพลิเคชัน

บทที่ 6
เอนจินส่วนกราฟิก

เป็นส่วนที่ใช้ควบคุมอุปกรณ์แสดงผล และแสดงผลภาพ 3 มิติ จัดการโมเดล 3 มิติ และการจัดวางมุมกล้องต่างๆ ทำให้การแสดงผลมีประสิทธิภาพ โดยจะแบ่งออกเป็นคลาสดังต่อไปนี้

6.1 คลาส cGraphics

คลาสนี้ทำหน้าที่สร้างสภาพแวดล้อมของแอพพลิชันให้สามารถแสดงผลภาพ 3 มิติได้อย่างมีประสิทธิภาพ และจัดการการแสดงผลภาพ 3 มิติ



รูปที่ 6-1 คลาสไดอะแกรมของ cGraphics

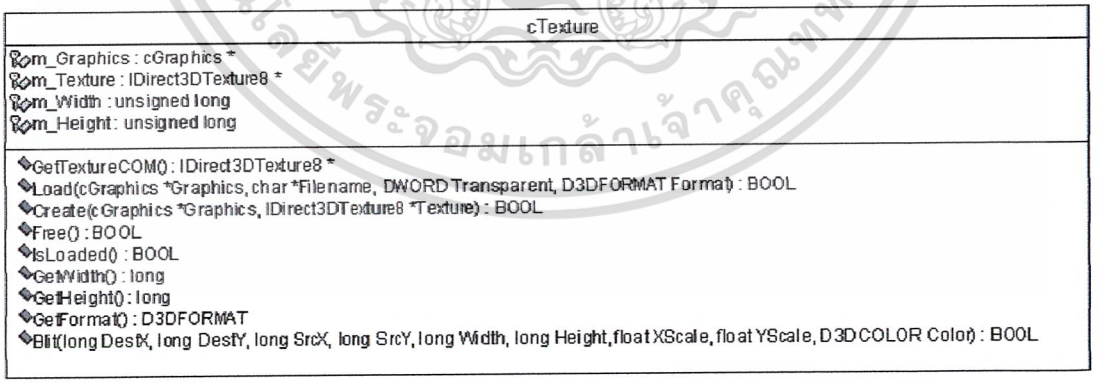
คลาสนี้จะทำการติดต่อกับอุปกรณ์แสดงผล เช่น การ์ดเร่งความเร็วกราฟิก เป็นต้น ซึ่งจะจัดการการแสดงผลในรูปแบบ 3 มิติ ซึ่งจะเริ่มการใช้งานโดยเรียกเมธอด Init() ซึ่งจะกำหนดค่าต่างๆ ให้กับสภาพแวดล้อมที่จะต้องใช้ในการแสดงผลภาพ 3 มิติ จากนั้นจะเริ่มทำงานในการแสดงผล โดยใช้เมธอด SetMode() ซึ่งจะมีลักษณะการแสดงผลอยู่ 2 แบบ คือแบบวินโดว์ (Window) และแบบเต็มหน้าจอ (Full Screen)

การแสดงผลภาพ 3 มิติจะทำโดยใช้เมธอด BeginScene() เพื่อทำการเรนเดอร์ภาพลงใน BackBuffer เมื่อทำการเรนเดอร์ภาพเสร็จเรียบร้อย จะต้องทำการเรียกเมธอด EndScene() เพื่อทำการจบการทำงานในส่วนการเรนเดอร์ภาพลงใน BackBuffer จากนั้นจะทำการแสดงผลภาพ 3 มิติโดยใช้เมธอด Display() ในการสลับ BackBuffer มายัง FrontBuffer เพื่อให้ภาพที่ทำการเรนเดอร์ที่ BackBuffer แสดงผลออกทางจอภาพ ซึ่งในก่อนการเรนเดอร์ภาพใน BackBuffer จะต้องมีการลบภาพที่อยู่ใน BackBuffer ออกเสียก่อนโดยใช้เมธอด Clear() เพื่อที่จะได้เรนเดอร์ภาพเฟรมถัดไป

นอกจากนั้นยังสามารถเปลี่ยนลักษณะการเรนเดอร์ได้โดยใช้เมธอด EnableLighting() ซึ่งจะทำการเปิดปิดการเรนเดอร์แสง เมธอด EnableZBuffer() จะทำการเปิดปิดลักษณะการเรนเดอร์ว่าให้เรนเดอร์ภาพแบบมีความลึกหรือไม่ ส่วนเมธอด EnableAlphaTesting() จะทำการเปิดปิดลักษณะการเรนเดอร์ภาพให้มีลักษณะโปร่งใส (Tranparent) หรือไม่

6.2 คลาส cTexture

คลาสนี้จะทำหน้าที่ในการเก็บ Texture รวมถึงรายละเอียดต่างๆ ของ Texture เช่น ความกว้าง ความสูง เป็นต้น ซึ่งอินสแตนซ์ของคลาสนี้ จะใช้แทน 1 Texture



รูปที่ 6-2 คลาสไดอะแกรมของ cTexture

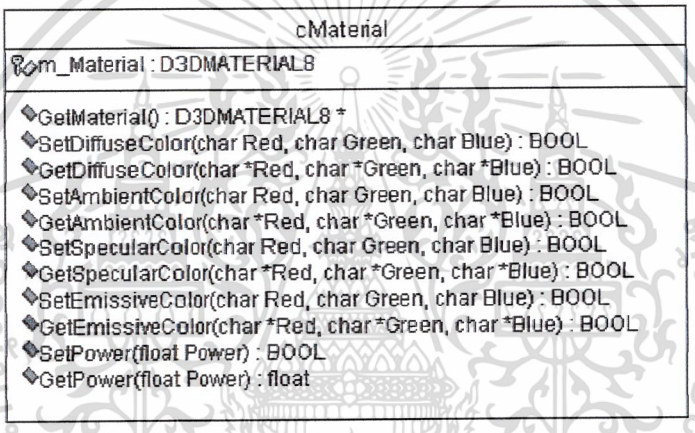
การใช้งานคลาสนี้จะมีอยู่ 2 วิธีคือถ้าต้องการโหลดภาพ Texture จากไฟล์จะใช้เมธอด Load() ซึ่งจะทำการโหลดไฟล์ขึ้นมาเป็น Texture ส่วนอีกวิธีคือ ถ้ามีการโหลด Texture ขึ้นมาแล้ว ซึ่งเก็บอยู่ในอินเตอร์เฟสของ Direct3DTexture8 ก็จะใช้เมธอด Create() ในการสร้าง Texture

คลาสนี้ยังสามารถที่จะวาด Texture ลงในส่วนแสดงผลได้โดยใช้เมธอด Blit() ซึ่งสามารถที่จะย่อขยาย และวาดยังตำแหน่งใดก็ได้ โดยกำหนดค่าลงในพารามิเตอร์ของเมธอดนี้ นอกจากนั้นยังสามารถทำให้วาดแบบโปร่งใสได้ โดยกำหนดสีที่จะเป็นสีที่เป็นสีโปร่งใส

โดยส่วนใหญ่การใช้งานคลาสนี้จะใช้ร่วมกับคลาส cGraphics ในการแปะภาพ Texture ลงบนส่วนของโพลีกอน โดยใช้เมธอด SetTexture() ของคลาส cGraphics ทำให้โพลีกอนที่ได้ดูเหมือนจริงยิ่งขึ้น

6.3 คลาส cMaterial

คลาสนี้จะทำการเปลี่ยนสีที่ปรากฏอยู่บนพื้นผิวของการเรนเดอร์อ็อบเจ็กต์ ซึ่งจะทำให้อ็อบเจ็กต์ที่ถูกเรนเดอร์นั้นมีลักษณะสมจริงยิ่งขึ้น



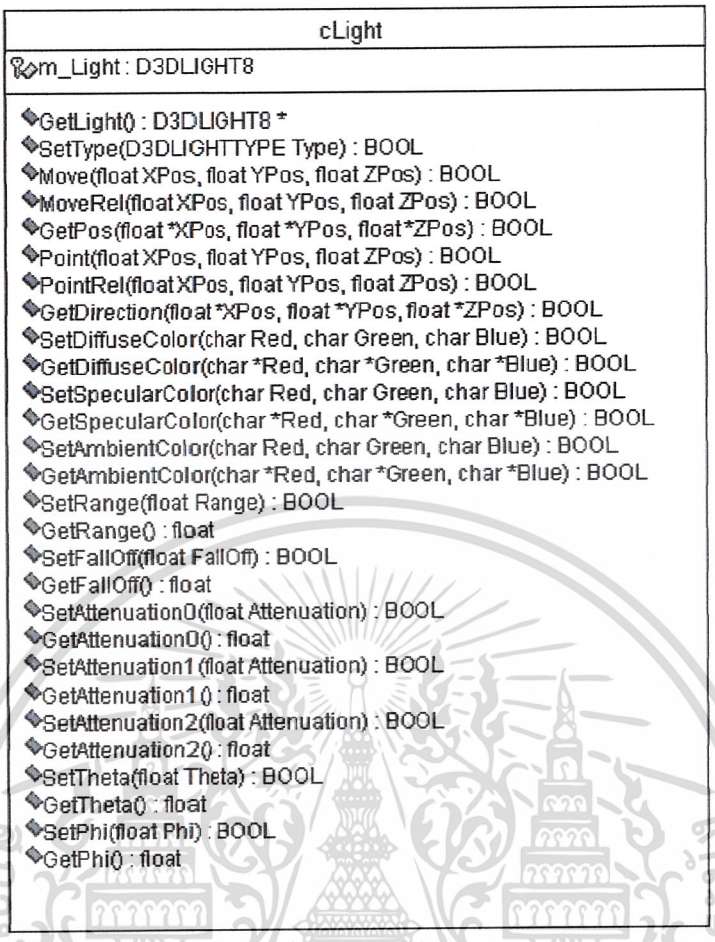
รูปที่ 6-3 คลาสไดอะแกรมของ cMaterial

คลาสนี้เพียงอินสแตนซ์เดียวจะเก็บได้เพียงโครงสร้างของ D3DMATERIAL เดียวเท่านั้น และจะมีเมธอดที่ใช้งานในการกำหนดลักษณะของสีของพื้นผิวที่จะเปลี่ยนไป โดยค่าของแต่ละสีจะมีค่าอยู่ระหว่าง 0 ถึง 255

คลาสนี้ไม่ค่อยมีการใช้งานมากนัก เนื่องจากจะใช้คลาส cTexture แทน เพราะการใช้ภาพแปะลงบนพื้นผิวของวัตถุจะดูสมจริงมากกว่าใช้สีวาดลงบนพื้นผิวของวัตถุ ซึ่งจะดูไม่สมจริงเท่า

6.4 คลาส cLight

คลาสนี้จะใช้สำหรับสร้างแสง สำหรับการเรนเดอร์ภาพให้ดูเหมือนจริงยิ่งขึ้น ซึ่งจะมีลักษณะของการสร้างแสงอยู่หลายลักษณะ



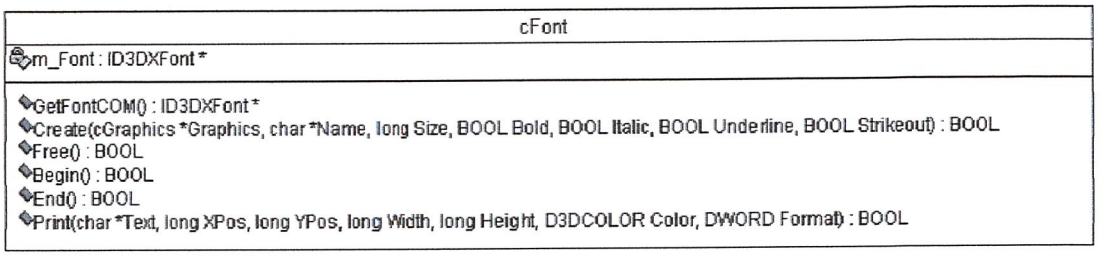
รูปที่ 6-4 คลาสไดอะแกรมของ cLight

ในการใช้งานคลาสนี้ จะใช้เมธอด SetType() ในการกำหนดรูปแบบของแสงที่ต้องการ ซึ่งสามารถกำหนดลักษณะออกเป็น 3 แบบ คือเป็นแบบจุด แบบกระจายออก และแบบทิศทาง และทำการกำหนดคุณสมบัติต่างๆ ของแสง เช่น สี รัศมีของแสง ความเข้ม โดยใช้เมธอดที่มีอยู่

คลาสนี้จะมีการใช้งานร่วมกับคลาส cGraphics ในการกำหนดให้มีการเรนเดอร์ลักษณะของแสงตามที่ได้กำหนดในคลาส cLight โดยใช้เมธอด SetLight() ของคลาส cGraphics

6.5 คลาส cFont

คลาสนี้จะใช้ทำการแสดงผลข้อความลงบนหน้าจอ โดยทำการเรนเดอร์ข้อความลงบน BackBuffer ก่อนที่จะมีการแสดงผล

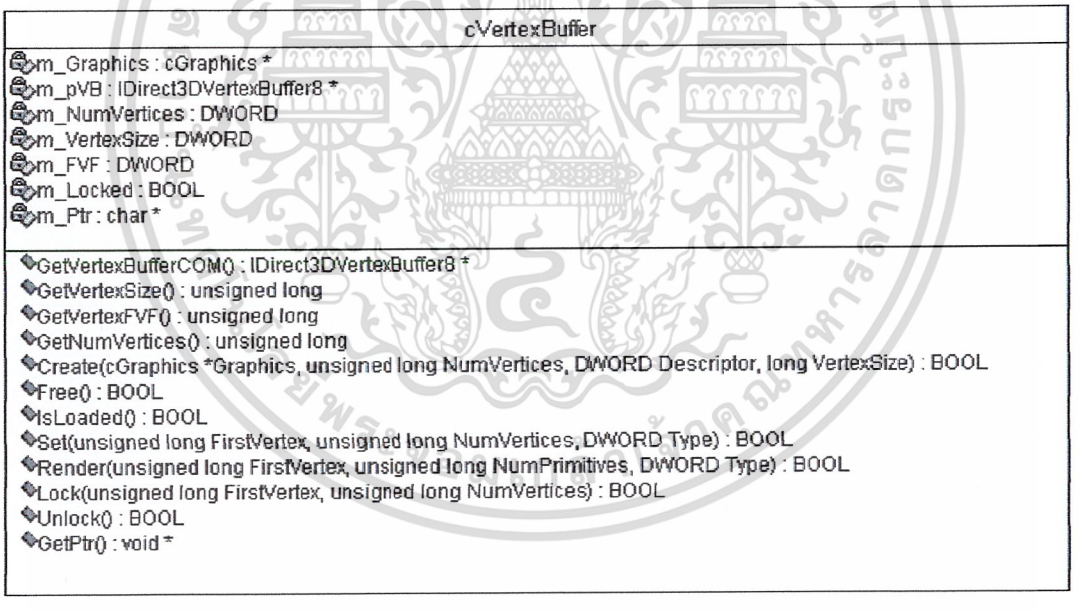


รูปที่ 6-5 คลาสไดอะแกรมของ cFont

ในการใช้งานคลาสนี้ จะทำโดยเรียกเมธอด Create() โดยกำหนดรูปแบบของตัวอักษรที่ต้องการ ขนาดของตัวอักษร และกำหนดลักษณะอื่นๆ เช่น ตัวหนา ตัวเอียง เป็นต้น และจะทำการเรนเดอร์ตัวอักษร โดยใช้เมธอด Print() โดยกำหนดข้อความที่ต้องการและตำแหน่งของข้อความที่ต้องการแสดงผล นอกจากนั้นยังกำหนดสีของข้อความที่ต้องการแสดงได้ด้วย

6.6 คลาส cVertexBuffer

คลาสนี้จะใช้ทำการสร้างเซตของจุด และสามารถเรนเดอร์ออกมาเป็นรูปต่างๆ ได้



รูปที่ 6-6 คลาสไดอะแกรมของ cVertexBuffer

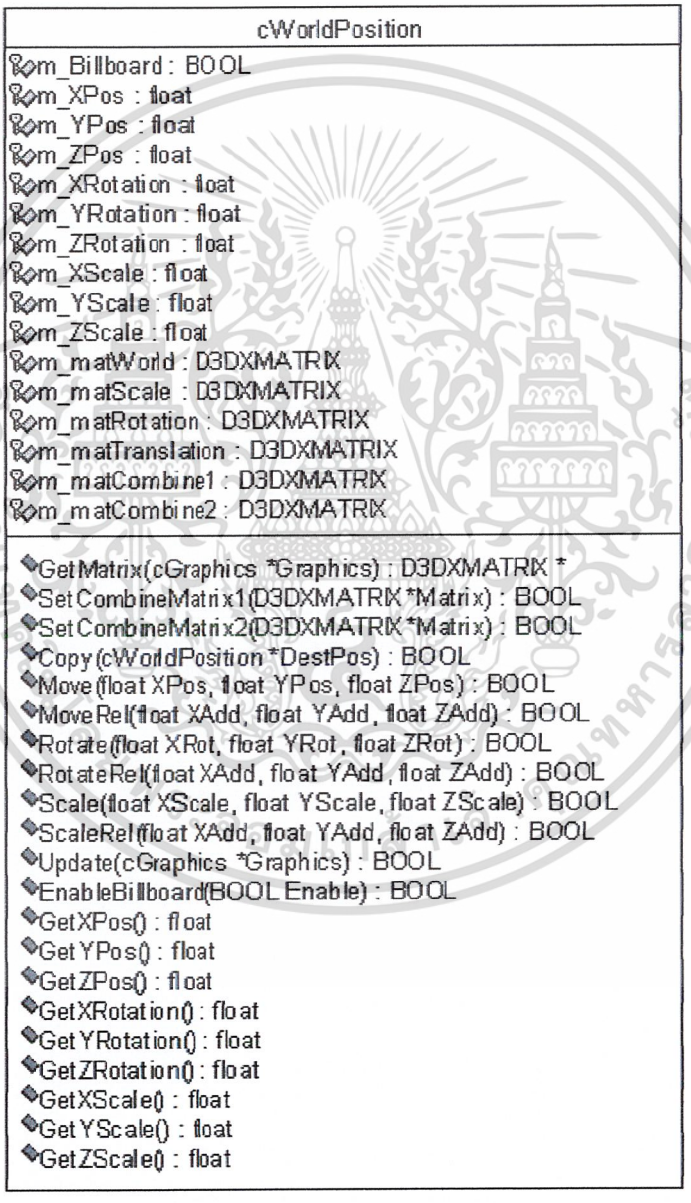
ในการใช้งาน จะทำการเรียกเมธอด Create() เป็นเมธอดแรก เพื่อทำการสร้างเวอร์เท็กซ์บัฟเฟอร์ ซึ่งเป็นหน่วยความจำที่ใช้เก็บลักษณะของเซตของจุด และเมื่อเลิกการใช้งานจะทำการคืนทรัพยากรให้กับระบบโดยใช้เมธอด Free()

เมื่อทำการสร้างบัฟเฟอร์ ก็จะทำการนำข้อมูลของจุดไปเก็บไว้ในบัฟเฟอร์ โดยใช้เมธอด Set() ในการกำหนดลักษณะของเซตของจุด จากนั้นจะทำการเรนเดอร์เซตของจุดโดยใช้เมธอด Render() ซึ่งจะ

ทำการวาดเซตของจุดให้เป็นโพลีกอน ซึ่งจะมีลักษณะของการวาดอยู่ 6 ลักษณะ คือ วาดแบบเป็นจุด วาดต่อกันเป็นเส้น วาดแต่ละจุดเชื่อมกันกับจุดก่อนหน้า วาดสามเหลี่ยมด้วยจุด 3 จุด วาดสามเหลี่ยมโดยใช้ 2 จุดก่อนหน้า และวาดสามเหลี่ยมโดยใช้จุดศูนย์กลางร่วมกัน

6.7 คลาส cWorldPosition

คลาสนี้ทำหน้าที่กำหนดตำแหน่งต่างๆ ของโพลีกอนในพิกัด 3 มิติ ซึ่งสามารถเปลี่ยนพิกัดตำแหน่งของโพลีกอนให้กลายเป็นพิกัดของโลก 3 มิติ



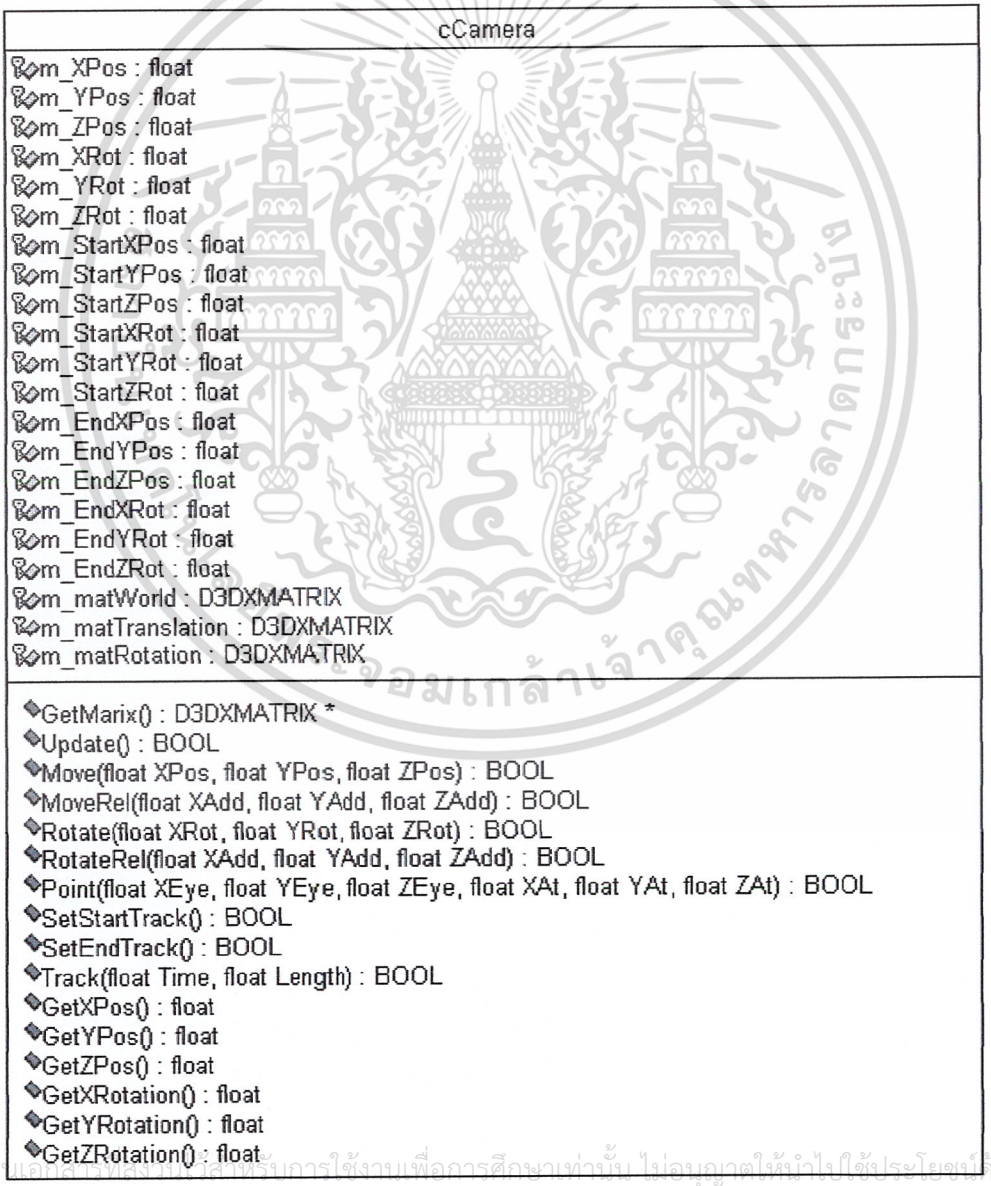
รูปที่ 6-7 คลาสไดอะแกรมของ cWorldPosition

คลาสนี้จะใช้งานโดยการใช้เมธอด Move() เพื่อทำการย้ายตำแหน่งของโพลีกอนไปยังตำแหน่งที่ต้องการในพิกัด 3 มิติ ถ้าต้องการหมุนโพลีกอนตามแกน X, Y และ Z ก็จะใช้เมธอด Rotate() โดยกำหนดค่าตามแกนที่ต้องการ และถ้าต้องการปรับเปลี่ยนขนาดของโพลีกอน ก็จะใช้เมธอด Scale() ในการปรับขนาด

นอกจากนั้นคลาสนี้ยังสามารถทำ Billboard ได้โดยการใช้เมธอด EnableBillboard() เพื่อให้โพลีกอนที่แสดงผล แสดงผลแบบ Billboard ได้

6.8 คลาส cCamera

คลาสนี้ใช้ในการจัดการเกี่ยวกับกล้อง เช่น การเปลี่ยนตำแหน่ง หรือการหมุนกล้องตามแกนต่างๆ ซึ่งจะทำให้มุมมองของเกมเปลี่ยนตามไปด้วย

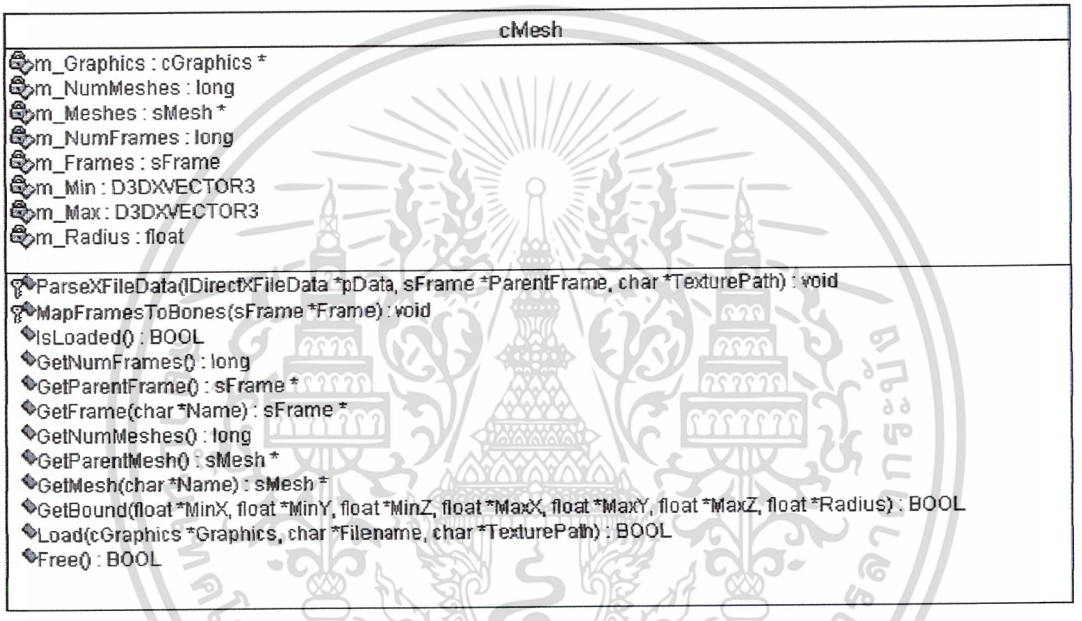


รูปที่ 6-8 คลาสไดอะแกรมของ cCamera

ในการใช้งานคลาสนี้ จะใช้เมธอด Move() เพื่อทำการเคลื่อนย้ายตำแหน่งของกล้องไปยังตำแหน่งที่ต้องการ ใช้เมธอด Rotate() เพื่อทำการหมุนกล้องตามแกนที่ต้องการ นอกจากนั้นยังสามารถกำหนดตำแหน่งกล้องและจุดที่กล้องโฟกัสได้โดยใช้เมธอด Point()

6.9 คลาส cMesh

คลาสนี้ช่วยในการจัดการโมเดล 3 มิติ ซึ่งจะใช้ตามรูปแบบไฟล์ที่มีนามสกุล X ซึ่งเป็นมาตรฐานของไดเร็กเอ็กซ์



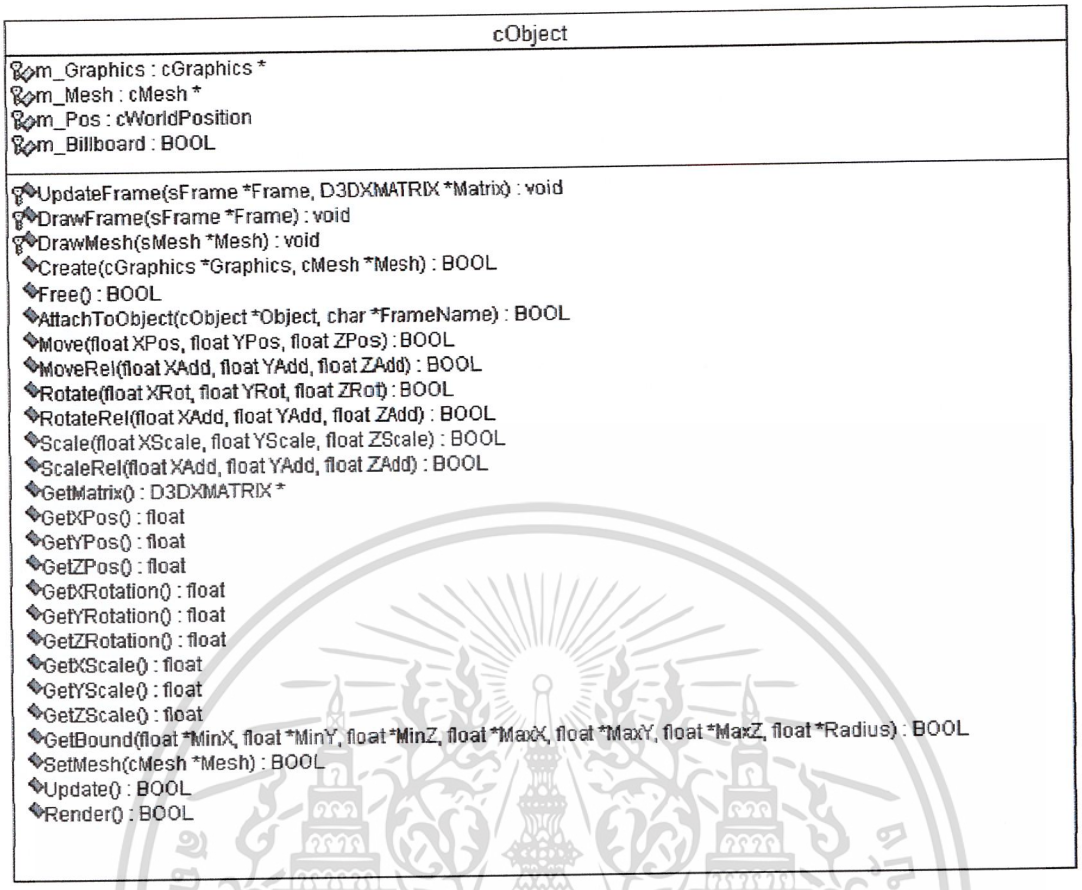
รูปที่ 6-9 คลาสไดอะแกรมของ cMesh

การใช้งานคลาสนี้จะใช้งานโดยเรียกเมธอด Load() ซึ่งจะทำการโหลดข้อมูลของโมเดล เช่น จำนวนเฟรมของโมเดล ชื่อของไฟล์ Texture ที่ใช้ในโมเดล ข้อมูลของจุดที่ใช้ในโมเดล เป็นต้น นอกจากนี้ยังมีเมธอด GetBound() ซึ่งใช้ในการหาขอบเขตของโมเดล ซึ่งส่วนมากจะใช้หาขนาดของโมเดลว่ามีขนาดเท่าใด

การใช้งานคลาส cMesh ส่วนใหญ่จะทำงานร่วมกันกับ cObject เพื่อใช้ในการแสดงโมเดล 3 มิติออกทางส่วนแสดงผล

6.10 คลาส cObject

คลาสนี้จะใช้ในการเรนเดอร์โมเดล 3 มิติให้ออกทางหน้าจอ ซึ่งจะช่วยให้สามารถใช้งานโมเดล 3 มิติได้อย่างมีประสิทธิภาพ และใช้งานหน่วยความจำในการเก็บโมเดลน้อยที่สุด



รูปที่ 6-10 คลาสไดอะแกรมของ cObject

คลาสนี้จะมีการใช้งานโดยเรียกใช้เมธอด Create() โดยใช้คลาส cMesh ในการสร้างอินสแตนซ์ของคลาสนี้ และจะทำการเปลี่ยนตำแหน่งโดยใช้เมธอด Move() ถ้าต้องการหมุนโมเดลตามแกนต่างๆ ก็จะใช้เมธอด Rotate() ส่วนถ้าต้องการปรับขนาดของโมเดล ก็จะใช้เมธอด Scale() เพื่อทำการปรับขนาดของโมเดล และเมื่อต้องการแสดงผลโมเดลออกทางส่วนแสดงผลก็จะใช้เมธอด Render() เพื่อแสดงผลออกทางหน้าจอ

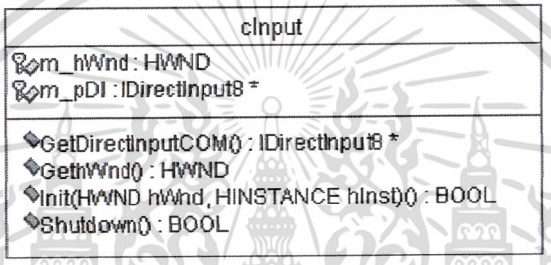
ข้อดีของการคลาสนี้ในการเรนเดอร์ก็คือ สามารถที่จะสร้างคลาส cObject ซึ่งใช้ Mesh เดียวกันได้ ทำให้ไม่เปลืองการใช้หน่วยความจำในการเก็บ Mesh และสามารถที่จะเปลี่ยนตำแหน่งโมเดล หมุนโมเดล หรือปรับขนาดได้โดยไม่กระทบอ็อบเจ็กต์อื่น

บทที่ 7
เอนจินส่วนอินพุต

เอนจินส่วนอินพุตเป็นส่วนติดต่อกับอุปกรณ์อินพุตต่างๆ เช่น คีย์บอร์ด เมาส์ จอยสติ๊ก ซึ่งสามารถที่จะควบคุมการทำงานของอุปกรณ์ต่างๆ โดยจะแบ่งออกเป็นคลาส ดังต่อไปนี้

7.1 คลาส cInput

เป็นคลาสที่จะทำการกำหนดค่าต่างๆ ที่จำเป็นต้องใช้ในการติดต่อกับอุปกรณ์อินพุต และเริ่มต้นติดต่อกับอุปกรณ์อินพุตที่ติดตั้งอยู่

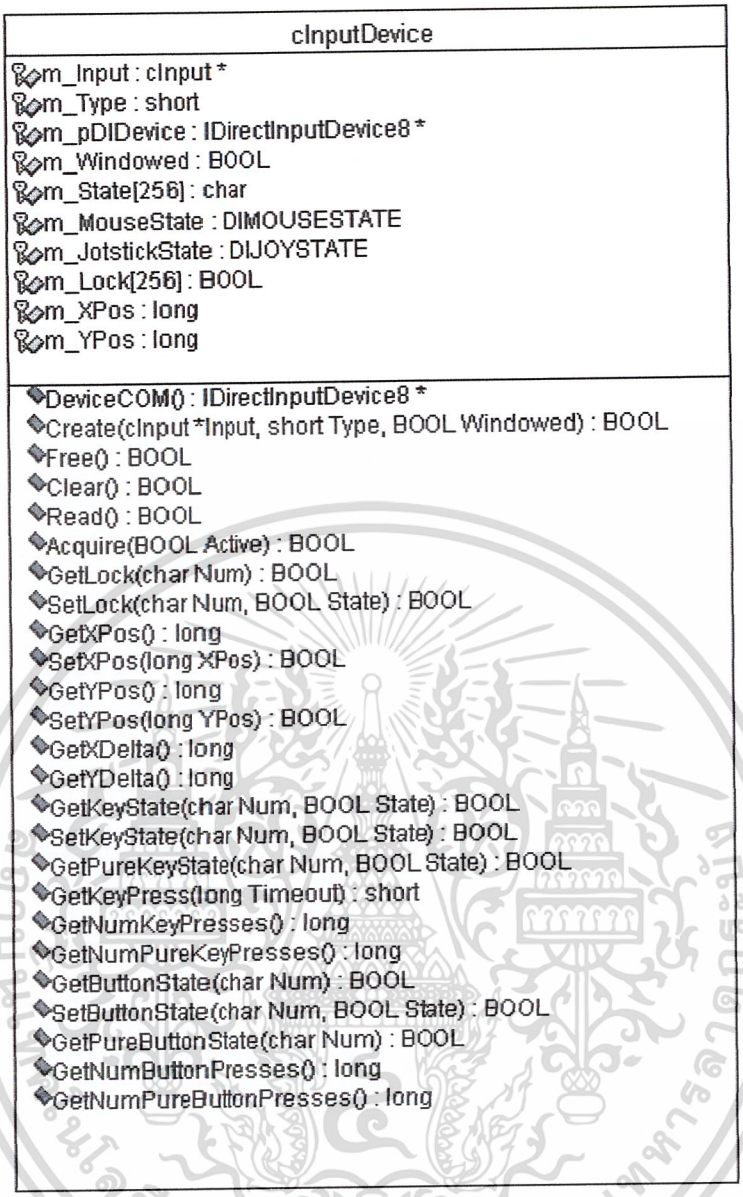


รูปที่ 7-1 คลาสไดอะแกรมของ cInput

เราจะทำการใช้งานโดยเรียกใช้เมธอด Init() ซึ่งจะทำการเริ่มต้นการติดต่อกับอุปกรณ์ต่างๆ โดยจะไปทำการกำหนดค่า และเริ่มต้นการติดต่อกับอุปกรณ์ผ่านอินเตอร์เฟสของ DirectInput และเมื่อต้องการยกเลิกการติดต่อกับอุปกรณ์จะทำการเรียกใช้งานเมธอด Shutdown() เพื่อยกเลิกการติดต่อกับอุปกรณ์อินพุตต่างๆ

7.2 คลาส cInputDevice

เป็นคลาสที่จะจำลองการทำงานของอุปกรณ์อินพุต ซึ่งแต่ละอินสแตนซ์ของคลาสจะแทนอุปกรณ์เพียงอันเดียว ซึ่งถ้าต้องการใช้หลายอุปกรณ์พร้อมกัน จำเป็นต้องสร้างหลายอินสแตนซ์เพื่อทำการใช้งานอุปกรณ์ต่างๆ



รูปที่ 7-2 คลาสไดอะแกรมของ cInputDevice

เราจะใช้งานโดยใช้เมธอด Create() โดยใช้คลาส cInput ที่ทำการกำหนดค่าแล้วมาติดต่อกับอุปกรณ์อินพุต และทำการกำหนดชนิดของอุปกรณ์ที่ต้องการ เช่น คีย์บอร์ด เมาส์ หรือจอยสติ๊ก เป็นต้น และเมื่อต้องการอ่านค่าจากอุปกรณ์อินพุต ก็จะใช้เมธอด Read() เพื่อทำการอ่านค่าสถานะของอุปกรณ์อินพุต ซึ่งจะไปทำการเก็บสถานะของอุปกรณ์ภายในตัวแปร m_State และเมื่อต้องการเลิกใช้อุปกรณ์นั้นก็จะทำการคืนทรัพยากรแก่ระบบโดยใช้เมธอด Free()

ในการใช้งาน เราจะแบ่งเมธอดออกเป็น 2 ส่วนใหญ่ๆ ซึ่งจะแบ่งตามชนิดการทำงานของอุปกรณ์ คือ คีย์บอร์ดกับเมาส์และจอยสติ๊ก ซึ่งส่วนของคีย์บอร์ดจะมีการใช้งานเมธอดหลักๆ คือ GetKeyState() โดยจะทำการอ่านค่าของปุ่มของคีย์บอร์ดที่ถูกกดอยู่ในขณะนั้น และส่วนของเมาส์และจอยสติ๊กนั้นจะมีการใช้เมธอดร่วมกัน เพราะมีการทำงานที่คล้ายกัน ซึ่งจะใช้เมธอดหลักๆ คือ GetXPos(), GetYPos() โดย

จะทำการอ่านค่าตำแหน่งของค่า X และ Y ที่เกิดจากการเลื่อนเมาส์ หรือกดปุ่มที่อยู่บนจอยสติ๊ก นอกจากนั้นยังมีเมธอด GetButtonState() จะทำการอ่านค่าปุ่มที่ถูกกดจากเมาส์หรือจอยสติ๊ก

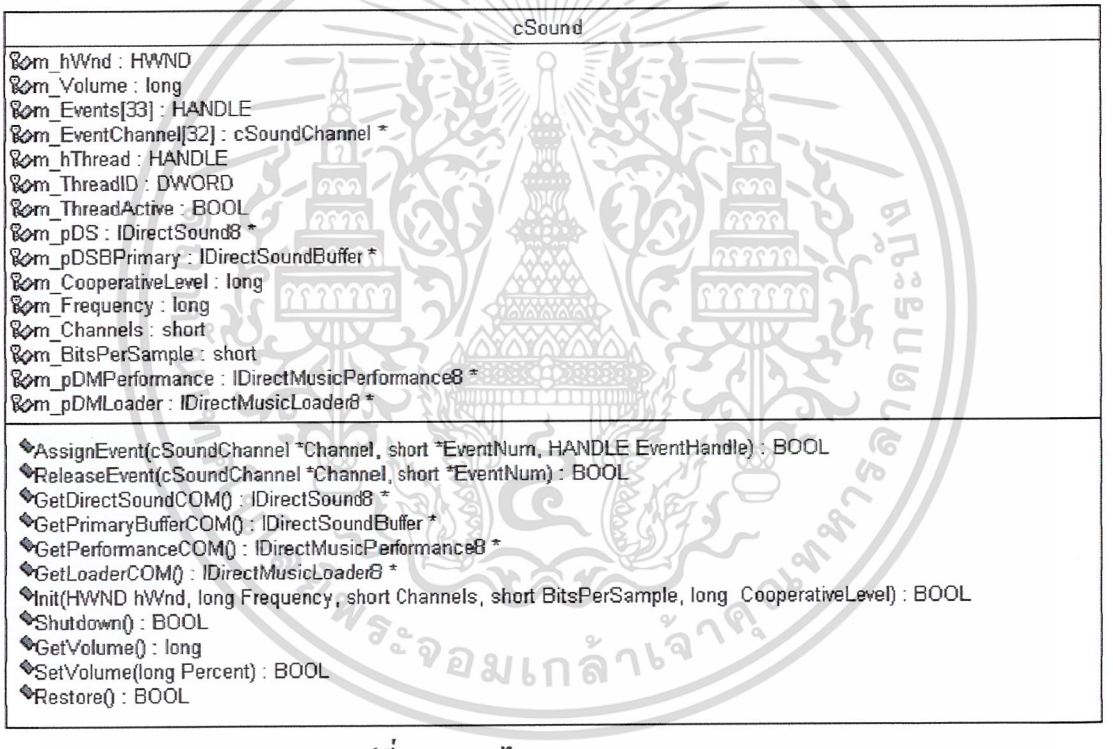
ในการอ่านค่าของปุ่มที่ถูกกด เราจะนำมาเทียบกับค่าคงที่ ซึ่งจะเป็นค่าของปุ่มต่างๆ ซึ่งค่าคงที่ของคีย์บอร์ดจะขึ้นต้นด้วย KEY_ แล้วตามด้วยชื่อของปุ่มที่ต้องการ เช่น ปุ่ม W ก็จะมีค่าคงที่เป็น KEY_W หรือถ้าต้องการปุ่ม ESC ก็จะมีค่าคงที่เป็น KEY_ESC ส่วนค่าคงที่ของเมาส์ จะขึ้นต้นด้วย MOUSE_ แล้วตามด้วยปุ่มของเมาส์ที่ต้องการ เช่น เมื่อกดเมาส์ปุ่มซ้าย จะได้ค่าคงที่เป็น MOUSE_LBUTTON โดยเราจะนำค่าคงที่เหล่านี้ส่งเข้าไปยังเมธอด เพื่อทำการเปรียบเทียบ จากนั้นเมธอดจะทำการคืนค่าผลลัพธ์ที่ได้ออกมาเป็น Boolean คือ TRUE เมื่อค่าคงที่ที่ส่งไปเป็นค่าเดียวกับปุ่มที่กด และจะเป็น FALSE เมื่อไม่ได้กดปุ่มตรงกับค่าคงที่ที่ส่งเข้าไป

บทที่ 8
เอนจินส่วนเสียง

เป็นส่วนที่ใช้ควบคุมและจัดการกับเสียงที่เราจะใส่เข้าไปในเกมของเรา โดยสามารถใส่เสียงเอฟเฟ็คต์เข้าร่วมได้ ซึ่งประกอบด้วยคลาส ดังนี้

8.1 คลาส cSound

คลาส cSound ทำหน้าที่ควบคุมการทำงานของอ็อบเจ็กต์ของ DirectSound และ DirectMusic และสามารถกำหนดระดับความดังของเสียง ซึ่ง cSound จะทำการสร้างอ็อบเจ็กต์ของ DirectSound ขึ้นมาให้โดยอัตโนมัติ

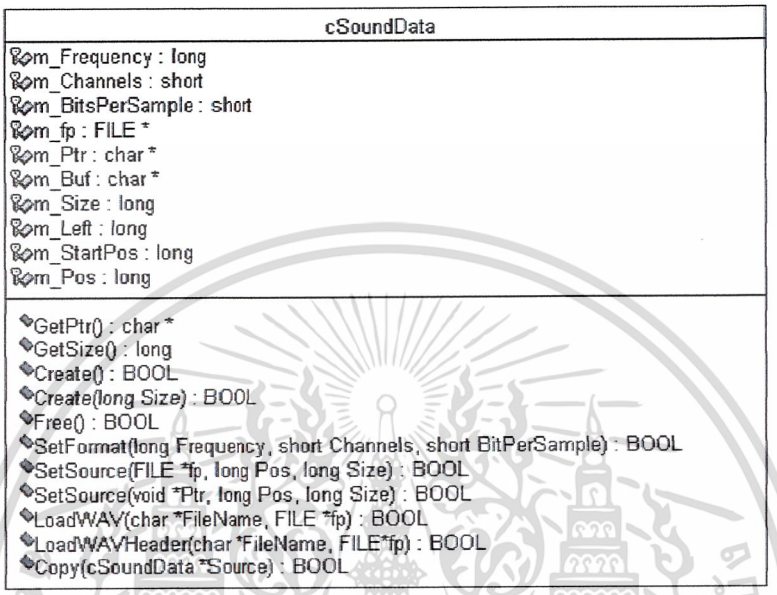


รูปที่ 8-1 คลาสไดอะแกรมของ cSound

ในการใช้งานจะมีเมธอดหลักในการใช้งานอยู่ 3 เมธอดด้วยกัน โดยจะเริ่มด้วยการเรียกเมธอด Init() เพื่อทำการกำหนดค่าเริ่มต้นให้กับเสียงที่ต้องการเล่น ซึ่งถ้าไม่กำหนดค่าใดจะมีค่าปกติคือ มีความถี่ 22,020 Hz ระบบเสียงเป็นแบบ Mono มีอัตราการสุ่มที่ 16 bit เมื่อใดที่ต้องการปรับความดังของเสียงที่เล่นจะใช้เมธอด SetVolume() ในการปรับความดังของเสียง ซึ่งจะใช้หน่วยเป็นเปอร์เซ็นต์แทนความดังของเสียง และมีค่าอยู่ระหว่าง 0 ถึง 100 และเมื่อทำการปิดแอพพลิเคชันต้องมีการเรียกเมธอด Shutdown() เพื่อคืนทรัพยากรที่เอามาจากระบบ

8.2 คลาส cSoundData

คลาสนี้ทำหน้าที่เก็บข้อมูลเสียง เช่น ความถี่เสียง, bit-per-sample, จำนวนของ Channel, ขนาดของเสียง โดยจะมีความสัมพันธ์กับคลาส cSoundChannel ซึ่งจะใช้คลาส cSoundData ในการเล่นเสียง โดยข้อมูลของเสียงจะได้มาจากที่เก็บข้อมูล 2 ที่ คือ ไฟล์และหน่วยความจำ



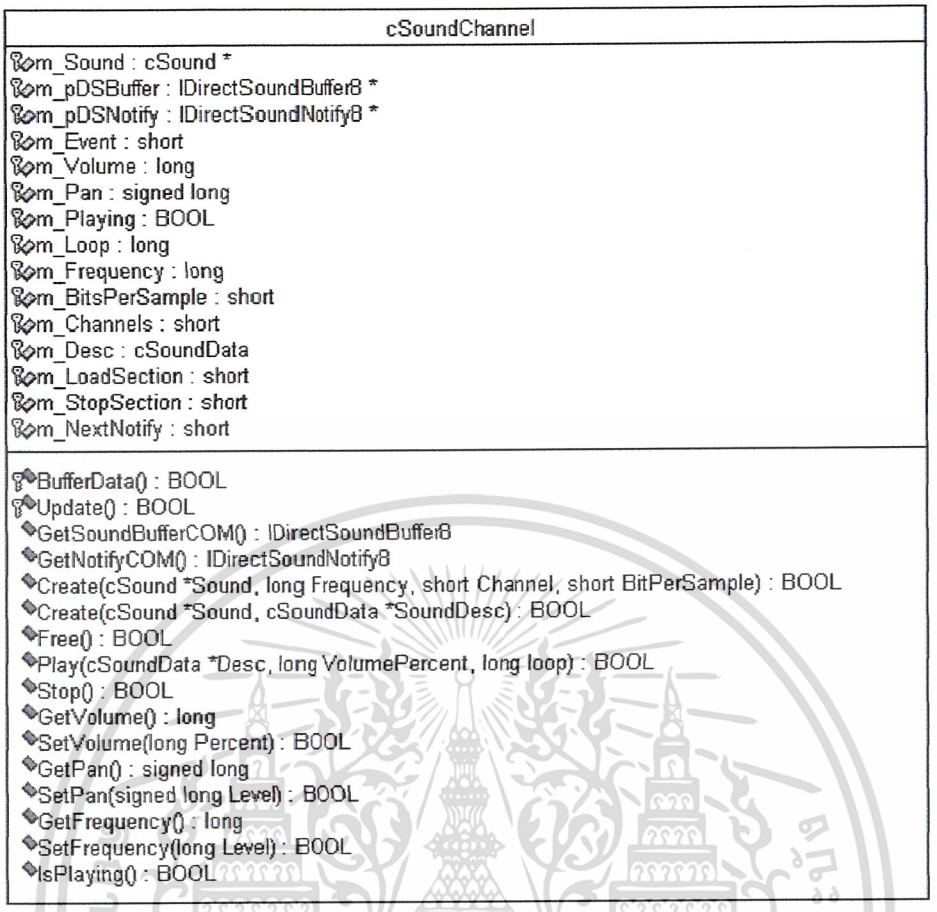
รูปที่ 8-2 คลาสไดอะแกรมของ cSoundData

การใช้งานคลาสนี้จะไม่ค่อยซับซ้อน เพราะว่าคลาสนี้เป็นคลาสที่ทำการ โหลดไฟล์เสียงขึ้นมาเก็บไว้เท่านั้น โดยทำงานแค่เพียงกำหนดรูปแบบของเสียงที่ต้องการ ซึ่งจะมีเมธอดที่จะทำการโหลดไฟล์เสียงที่มีนามสกุลเป็น WAV อย่างง่ายคือเมธอด LoadWAV() โดยสามารถโหลดได้ 2 วิธีคือ โหลดเสียงจากไฟล์ที่มีนามสกุล WAV โดยตรง หรือโหลดเสียงผ่านไฟล์พอยเตอร์

ถ้ามีการใช้เสียงที่มาจากที่เก็บข้อมูลแบบหน่วยความจำนั้น จะต้องมีการสร้างบัฟเฟอร์ โดยใช้เมธอด Create() ซึ่งจะกำหนดขนาดของบัฟเฟอร์ที่ต้องการ ซึ่งจะรู้ขนาดของบัฟเฟอร์ได้โดยการเรียกใช้เมธอด LoadWAVHeader() และจะอ้างอิงถึงบัฟเฟอร์ที่สร้างขึ้นมา โดยใช้เมธอด GetPtr()

8.3 คลาส cSoundChannel

คลาสนี้จะทำหน้าที่เล่นไฟล์เสียง ซึ่งเก็บอยู่ในออบเจ็กต์ของคลาส cSoundData โดยสามารถปรับรูปแบบการเล่นได้ตามที่ผู้ใช้ต้องการ



รูปที่ 8-3 คลาสไดอะแกรมของ cSoundChannel

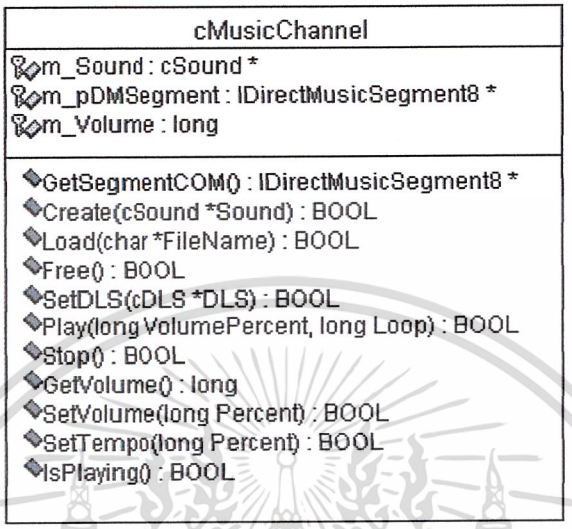
ในการใช้งานคลาส cSoundChannel จะใช้งานร่วมกับคลาส cSoundData ซึ่งใช้ในการเล่นเสียง ซึ่งเราสามารถสร้างอินสแตนซ์ของคลาสนี้ได้สูงสุด 32 อินสแตนซ์ ดังนั้นจึงสามารถสร้างเสียงได้ทั้งหมด 32 Channel ซึ่งสามารถเล่นได้พร้อมๆ กัน การใช้งานจะทำโดยเรียกเมธอด Create() ทำการสร้างรูปแบบของเสียงที่ต้องการเล่น หรือถ้าสร้างอินสแตนซ์ของคลาส cSoundData ไว้แล้วก็สามารถใช้ตามรูปแบบที่กำหนดไว้แล้วในอินสแตนซ์ โดยไม่ต้องกำหนดรูปแบบของเสียงที่จะเล่นใหม่

การทำงานส่วนใหญ่ของคลาสนี้จะเกี่ยวกับการเล่นเสียง ดังนั้นเมธอดหลักที่ใช้จะมีอยู่ 4 เมธอด คือ เมธอด Play() จะทำการเล่นเสียงให้ออกทางลำโพง ซึ่งเมื่อต้องการที่จะหยุดการเล่นก็จะใช้เมธอด Stop() เพื่อทำการหยุด ถ้าต้องการปรับความดังของเสียงที่ออกจะใช้เมธอด SetVolume() ซึ่งจะใช้หน่วยเป็นเปอร์เซ็นต์ โดยมีค่าอยู่ระหว่าง 0 ถึง 100 และถ้าต้องการตรวจสอบว่ากำลังเล่นเสียงอยู่หรือไม่โดยใช้เมธอด IsPlaying() เพื่อทำการตรวจสอบ

นอกจากนั้นยังสามารถปรับให้เสียงที่เล่นออกทางลำโพงข้างใดข้างหนึ่งดังกว่ากันก็ได้ โดยใช้เมธอด SetPan() โดยจะใช้หน่วยเป็นเปอร์เซ็นต์ ซึ่งมีค่าอยู่ระหว่าง –100 ถึง +100 ซึ่งค่าติดลบจะหมายความว่าให้เสียงออกลำโพงทางซ้ายมากกว่าลำโพงทางขวา ส่วนค่าบวกจะหมายความว่าให้เสียงออกลำโพงทางขวามากกว่าลำโพงทางซ้าย

8.4 คลาส cMusicChannel

คลาสนี้จะใช้ทำการเล่นไฟล์เสียงที่มีนามสกุล MID และ SGT ซึ่งเป็นรูปแบบของ DirectMusic native song

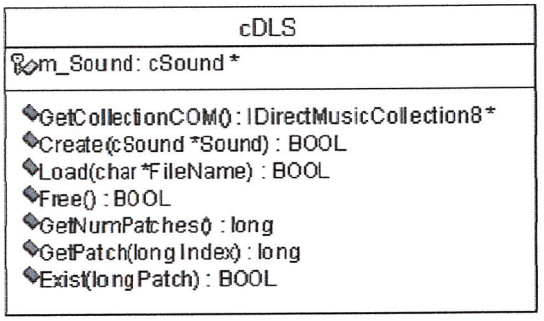


รูปที่ 8-4 คลาสไดอะแกรมของ cMusicChannel

ในการใช้งานคลาสนี้ จะทำการกำหนดค่าเริ่มต้นเพียงครั้งเดียวเท่านั้น โดยใช้เมธอด Create() และเมื่อเราต้องการเล่นไฟล์เสียงใด ก็จะใช้เมธอด Load() โดยจะเล่นได้กับไฟล์เสียงที่มีนามสกุล MID และ SGT เท่านั้น เมื่อต้องการเล่นไฟล์เสียงที่ทำการโหลดแล้วโดยใช้เมธอด Play() โดยกำหนดความดังของเสียงและกำหนดว่าต้องการให้เล่นวนใหม่อีกครั้งหรือไม่เมื่อเล่นจบ เมื่อต้องการยกเลิกการเล่น ก็จะใช้เมธอด Stop() เพื่อทำการหยุดเล่น เมื่อต้องการที่จะเลิกเล่นเสียงก็จะทำการคืนทรัพยากรให้กับระบบโดยใช้เมธอด Free() และถ้าต้องการตรวจสอบว่ามีการเล่นไฟล์เสียงอยู่หรือไม่ ก็จะทำการตรวจสอบโดยใช้เมธอด IsPlaying()

8.5 คลาส cDLS

คลาสนี้จะใช้ในการเก็บ DLS (Downloadable Sounds) ซึ่งจะถูกนำไปใช้โดยคลาส cMusicChannel



รูปที่ 8-5 คลาสไดอะแกรมของ cDLS

ในการใช้งานจะต้องสร้างอินสแตนซ์ของคลาสนี้ขึ้นมาก่อน โดยใช้เมธอด Create() จากนั้นจะทำการโหลดเซตของ DLS ขึ้นมาโดยใช้เมธอด Load() และเมื่อต้องการยกเลิกการใช้งานคลาส cDLS ก็จะทำการเรียกเมธอด Free() เพื่อทำการคืนทรัพยากรให้กับระบบ

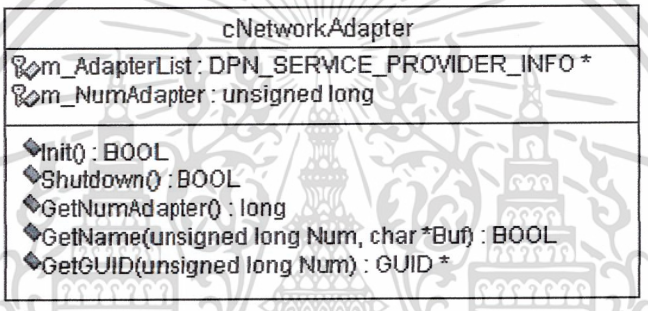
นอกจากนั้นยังมีเมธอดที่ใช้กับ Instrument ของเซตของ DLS โดยจะมีอยู่ทั้งหมด 3 เมธอดหลัก คือ เมธอด GetNumPatch() ซึ่งจะบอกว่ามี Instrument อยู่ภายในเซตของ DLS เท่าใด เมธอด GetPatch() ใช้หาหมายเลข Patch ของ Instrument ที่อยู่ในเซตของ DLS และเมธอดสุดท้ายคือ Exist() ซึ่งจะทำการตรวจสอบหมายเลข Patch ว่ามีอยู่ในเซตของ DLS หรือไม่

บทที่ 9
เอนจินส่วนเน็ตเวิร์ค

เอนจินส่วนเน็ตเวิร์คจะจัดการติดต่อกับการ์ดเน็ตเวิร์ค โดยจะควบคุมกระบวนการทำงานในการติดต่อสื่อสารกันระหว่างเครื่องคอมพิวเตอร์ และจะจัดการการทำงานในส่วนเซิร์ฟเวอร์ และไคลเอนต์ โดยมีคลาสที่เกี่ยวข้องดังนี้

9.1 คลาส cNetworkAdapter

เป็นคลาสที่จะทำการติดต่อกับการ์ดเน็ตเวิร์ค และจัดการกำหนดลักษณะการเชื่อมต่อระหว่างเครื่องคอมพิวเตอร์ เพื่อให้สามารถติดต่อสื่อสารกันได้



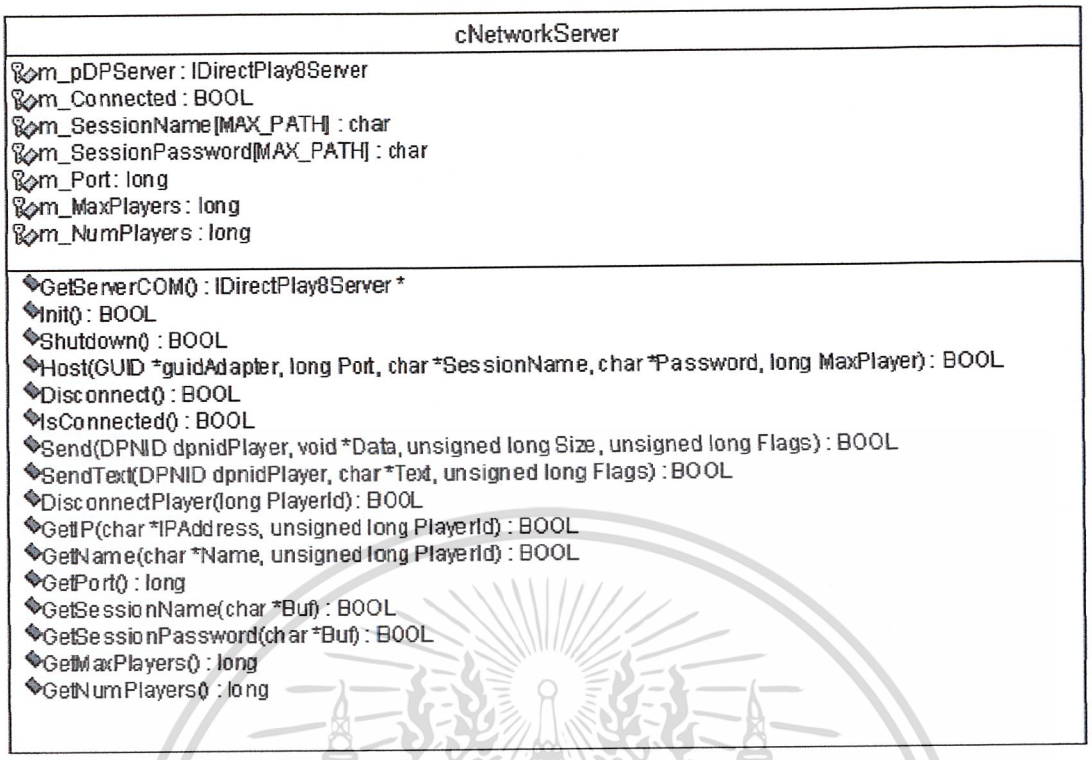
รูปที่ 9-1 คลาสไดอะแกรมของ cNetworkAdapter

คลาสนี้จะทำการติดต่อกับอุปกรณ์เน็ตเวิร์ค โดยผ่านโปรโตคอล TCP/IP ในการใช้งาน การติดต่อกันระหว่างคอมพิวเตอร์นั้น เราจะต้องรู้ GUID โดยเราจะทำการเรียกเมธอด Init() ซึ่งจะทำการกำหนดค่าเริ่มต้นที่จำเป็น และลักษณะการเชื่อมต่อต่างๆ โดยผ่านอินเตอร์เฟสของ DirectPlay ซึ่งเมื่อนำการเรียกเมธอดนี้ ก็จะได้ชื่อของ Adapter และ GUID ซึ่งถ้ามีหลาย Adapter ก็จะทำเก็บจำนวนของ Adapter ที่ตัวแปรชื่อ m_NumAdapter เราจะรู้ GUID ได้โดยใช้เมธอด GetGUID() โดยส่งหมายเลขของ Adapter ที่ต้องการ แล้วจะได้ผลลัพธ์เป็น GUID ของ Adapter ที่ต้องการ และเมื่อใดที่ต้องการเลิกการติดต่อ จะทำการเรียกเมธอด Shutdown() เพื่อทำการยกเลิกการเชื่อมต่อกับระบบเน็ตเวิร์ค

คลาสนี้จะเป็นคลาสหลักที่ใช้เริ่มต้นในการจะทำการติดต่อสื่อสารกันผ่านระบบเน็ตเวิร์ค ซึ่งจำเป็นต้องสร้างขึ้นมาก่อนที่จะทำการกำหนดว่าคอมพิวเตอร์เครื่องใดจะทำหน้าที่เป็นเซิร์ฟเวอร์ หรือไคลเอนต์

9.2 คลาส cNetworkServer

เป็นคลาสที่จะทำหน้าที่เป็นเซิร์ฟเวอร์ ซึ่งจะคอยควบคุมการเชื่อมต่อกันระหว่างไคลเอนต์ ดูแล และจัดการเมสเสจที่เข้าและออก เป็นต้น



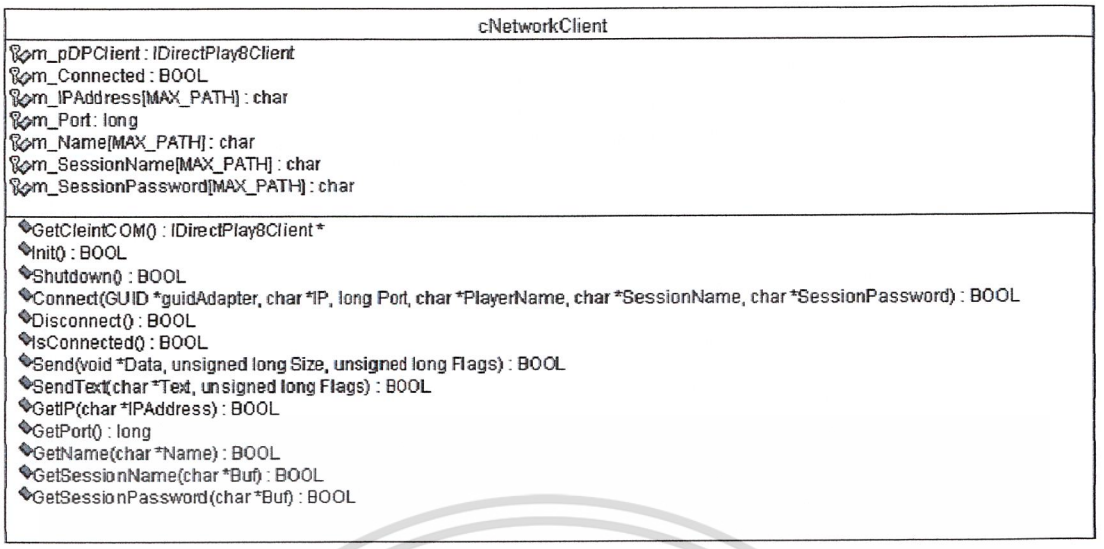
รูปที่ 9-2 คลาสไดอะแกรมของ cNetworkServer

คลาสนี้จะเริ่มใช้งานโดยเรียกเมธอด Init() ซึ่งจะทำการกำหนดค่าเริ่มต้นที่จำเป็นในการใช้งานในการทำงานเป็นเซิร์ฟเวอร์ จากนั้นจะทำการเรียกเมธอด Host() โดยจะส่ง GUID ของ Adapter ที่ต้องการ พอร์ตที่ต้องการใช้ติดต่อ ชื่อและรหัสของ Session ที่ต้องการ ซึ่งเมธอดนี้จะทำการคอยส่งและรับเมสเสจเพื่อทำการประมวลผลเมสเสจที่เข้ามา และทำการทำงานตามลักษณะของเมสเสจที่รับมา เราสามารถตรวจสอบว่ามีการเชื่อมต่ออยู่หรือไม่ โดยใช้เมธอด IsConnected() และเราสามารถส่งเมสเสจไปยังคอมพิวเตอร์เครื่องอื่นได้โดยใช้เมธอด Send() โดยเราจำเป็นต้องรู้ DPNID ของคอมพิวเตอร์ที่จะทำการส่ง ซึ่งเราจะทำรู้ได้จาก Header ของเมสเสจที่รับเข้ามา และเมื่อเราต้องการยกเลิกการทำงานของเซิร์ฟเวอร์ก็จะใช้เมธอด Shutdown() เพื่อยกเลิกการเชื่อมต่อกับคอมพิวเตอร์อื่นๆ

การใช้งานนั้นจะมีการกำหนดจำนวนผู้ใช้ไว้ เพื่อไม่ให้มีผู้ใช้มากเกินที่เซิร์ฟเวอร์จะรับได้ ซึ่งจะช่วยให้ไม่เกิดความล่าช้าในการประมวลผลเมสเสจ และสามารถตอบรับกลับไปยังไคลเอนต์ได้อย่างทันที เรายังสามารถดูจำนวนของผู้ใช้ที่ทำการเชื่อมต่อได้โดยใช้เมธอด GetNumPlayer() ซึ่งจะคืนค่าจำนวนผู้ใช้ที่เชื่อมต่ออยู่ในขณะนั้น

9.3 คลาส cNetworkClient

ทำหน้าที่เป็นไคลเอนต์ ซึ่งจะทำการส่งและรับเมสเสจจากเซิร์ฟเวอร์ โดยจะทำการเชื่อมต่อกับเซิร์ฟเวอร์ เพื่อสื่อสารกับไคลเอนต์อื่นๆ หรือเพื่อต้องการให้เซิร์ฟเวอร์ทำงานให้



รูปที่ 9-3 คลาสไดอะแกรมของ cNetworkClient

คลาสจะทำการเริ่มใช้งานโดยทำการเรียกเมธอด Init() เหมือนคลาส cNetworkServer และจะเริ่มทำการเชื่อมต่อไปยังเซิร์ฟเวอร์โดยใช้เมธอด Connect() ซึ่งจะต้องบอกหมายเลข IP ของเซิร์ฟเวอร์และบอกหมายเลขพอร์ตที่จะทำการเชื่อมต่อ จากนั้นบอกชื่อของ Session ที่ต้องการเข้าร่วมและถ้า Session นั้นมีรหัสในการเข้าร่วมก็ให้ใส่เข้าไปด้วย จากนั้นก็จะทำการเชื่อมต่อไปยังเซิร์ฟเวอร์นั้น ถ้าต้องการตรวจสอบว่ามีการเชื่อมต่ออยู่หรือไม่ จะใช้เมธอด IsConnected() เพื่อช่วยในการตรวจสอบ และถ้าต้องการยกเลิกการเชื่อมต่อกับเซิร์ฟเวอร์ ก็จะทำการเรียกใช้เมธอด Disconnect()

ในการส่งเมสเสจจะใช้เมธอดอยู่ 2 เมธอดก็คือ ถ้าต้องการส่งข้อมูลที่เป็นรูปแบบตามที่ต้องการ ก็จะใช้เมธอด Send() ซึ่งจำเป็นต้องบอกขนาดของเมสเสจด้วย แต่ถ้าต้องการส่งเพียงข้อความอย่างเดียวจะใช้เมธอด SendText() ซึ่งจะทำให้สะดวกต่อการส่งมากกว่าใช้เมธอด Send()

ในการใช้งานคลาส cNetworkServer และคลาส cNetworkClient นั้น จะต้องมีการสืบทอดคลาสดังกล่าวมาเป็นคลาสใหม่ เนื่องจากจำเป็นต้องมีการประมวลผลจัดการกับเมสเสจตามที่ผู้ใช้ต้องการ ดังนั้นการสืบทอดคลาสนั้นจะต้องทำการสร้างเมธอดใหม่ขึ้นมา เพื่อที่จะจัดการกับรูปแบบของเมสเสจที่ผู้ใช้ต้องการ ซึ่งส่วนใหญ่จะสร้างโดยให้มีชื่อเมธอดเป็น Receive() โดยจะรับเมสเสจมา และทำการตรวจสอบว่าเป็นเมสเสจชนิดใด และทำการประมวลผลเมสเสจ จากนั้นอาจทำการตอบสนองเมสเสจ หรือทำการอย่างใดอย่างหนึ่งตามที่ผู้พัฒนาต้องการตามชนิดของเมสเสจนั้น

บทที่ 10
เอนจินของกล้องระดับที่ 2

ในคลาส cCamera ที่ได้จัดทำขึ้นในเอนจิน ถือเป็นคลาสที่มีความสามารถในการจัดการเรื่องกล้องที่สืบทอดต่อจากไดเร็กเอ็กซ์ได้ดีระดับหนึ่ง ซึ่งมีความสามารถในการย้าย (Move) , การหมุน (Rotation) แต่ในการเคลื่อนไหวของกล้องที่จะเอามาใช้ในเกมจริงๆ นั้น สามารถพัฒนาให้ใช้งานง่ายขึ้นไปอีก ตามแบบอย่างของเกมที่จะสร้างขึ้นมา ดันนั้น จึงมีการนำเอา คลาส cCamera ซึ่งใช้พื้นฐานของไดเร็กเอ็กซ์มาพัฒนาต่อยอดเป็น คลาส cCameraLv2 เพื่อที่จะได้เอาเอนจินที่กล่าวมานี้ไปใช้ในเกมที่จะสร้างขึ้นมาได้

เนื่องจากการเคลื่อนไหวของกล้องนั้น แบ่งได้เป็น 3 แบบ หลักๆ ดังนั้น การที่จะทำให้กล้องเคลื่อนไหวได้อย่างมีอิสระนั้น จึงจำเป็นต้องรู้ชนิดของการเคลื่อนไหวก่อน ซึ่งการเคลื่อนไหวนั้น แบ่งออกเป็น 3 ชนิดดังนี้

1. First – Person View
2. Free – Look Camera
3. Camera around Object

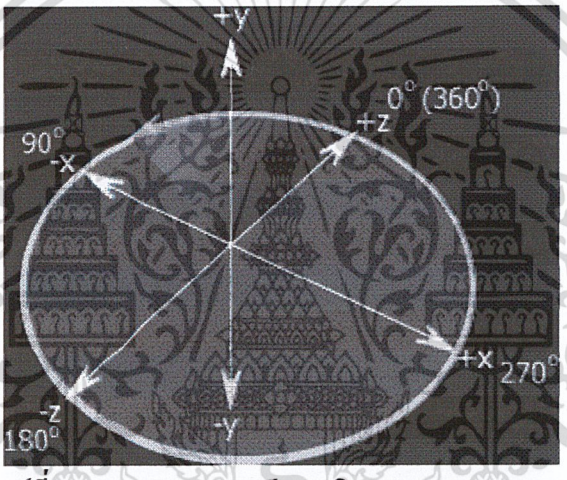
10.1 First – Person View

First – Person View หรือ การเคลื่อนไหวกล้องแบบ "มุมมองบุคคลที่ 1" นั้น จะเป็นการเคลื่อนที่ของกล้อง โดนวางตำแหน่งของกล้องไว้ ณ.ที่จุดดวงตาของตัวละคร ดังนั้น เราจะเห็นในสิ่งที่ตัวละครเห็น เมื่อตัวละครเคลื่อนที่ไปข้างหน้า กล้องก็จะเคลื่อนที่ตามไปด้วย ซึ่งโดยส่วนมาก มุมมองบุคคลที่ 1 นั้น จะไม่สามารถมองเห็นตัวละครที่เรากำลังเล่นได้ ดังนั้น จึงเขียนโค้ดเพื่อรับค่าจากเมาส์และคีย์บอร์ด แล้วสั่งให้กล้องเคลื่อนไหวโดยตรงตามที่เราต้องการเลย โดยการสั่งให้กล้องเคลื่อนไหวนั้นจะมีความสามารถดังนี้

- หมุนกล้องไปทางซ้ายและขวา (อาจใช้โดยการเลื่อนเมาส์ไปทางซ้ายและขวาตามลำดับ)
- แหงนมองขึ้นข้างบนและมองลงข้างล่างได้ (อาจใช้โดยการเลื่อนเมาส์ไปข้างบนและข้างล่างตามลำดับ) นอกจากนี้การมองไปด้านบนและด้านล่างนั้นต้องกำหนดให้มุมของการมองไม่สามารถเกิน 90 องศาได้ คือสามารถมองขึ้นได้สูงสุดคือที่เหนือหัวของตัวเองเท่านั้น ไม่สามารถที่จะมองเอนไปข้างหลังได้ และจุดที่มองต่ำสุดคือเท้าของตัวเองหรือตำแหน่งที่กำลังยืนอยู่
- เคลื่อนที่ไปข้างหน้า , เคลื่อนที่ไปข้างหลัง , ขยับไปทางซ้าย , ขยับไปทางขวา (อาจใช้โดยการกดปุ่มบน คีย์บอร์ด ตามที่กำหนด)

10.1.1 การหมุนกล้องไปทางซ้ายและขวา

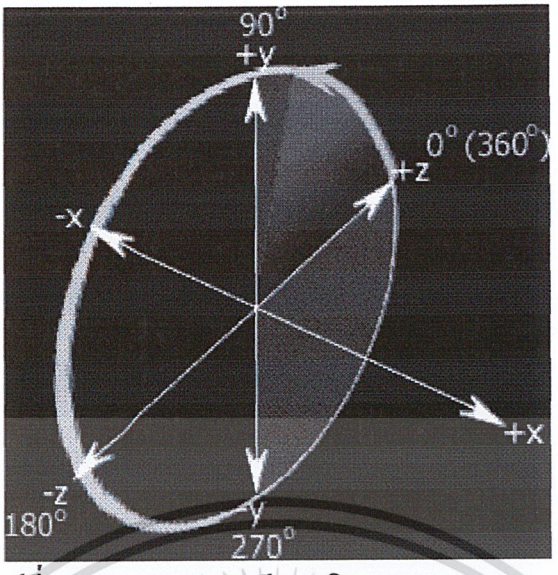
การหมุนกล้องไปทางซ้ายและขวานั้น (หรือเรียกอีกอย่างว่าการ Yaw) ในโลกสามมิติ คือการหมุนกล้องรอบแกน y นั่นเอง ดังนั้น เราต้องมีการสั่งไว้ว่า ให้หมุนไปกี่องศาโดยเก็บค่าของการหมุนไว้ (ใน คลาส ใช้ตัวแปรเป็นชนิด float ชื่อว่า fAngleY) ปรกติแล้ว การหมุนรอบแกน Y จะเป็นการหมุนแบบทวนเข็มนาฬิกา ดังนั้น จึงจะรับค่าองศาของการหมุนแต่ละทีตามค่าแรงเคลื่อนที่ของเมาส์กล่าวคือ ถ้าเมาส์เคลื่อนที่เร็ว แรง องศาที่จะเพิ่มเข้าไปก็จะมีมาก และจะหมุนได้เร็ว เมื่อได้ค่าแรงการเคลื่อนที่ของเมาส์ แล้วก็นำมาบวกให้แก่ตัวแปร fAngleY ไปเรื่อยๆ แต่เนื่องจากมุมของการหมุนนั้น อยู่ในช่วง 0.0 ถึง 360.0 องศา ดังนั้น จึงต้องกำหนดให้ตัวโปรแกรมเช็คตรงส่วนนี้ไม่ให้ค่า fAngleY มีน้อยไปกว่า 0.0 และไม่มากไปกว่า 360.0 องศา สุดท้ายคือการสั่งให้กล้องหมุนตามค่า fAngleY นั่นเอง (การหมุนกล้องจะใช้เมธอดของคลาส cCamera ในการหมุน)



รูปที่ 10-1 การหมุนทวนเข็มนาฬิกาของมุมรอบแกน y

10.1.2 การแหงนมองขึ้นข้างบนและมองลงข้างล่าง

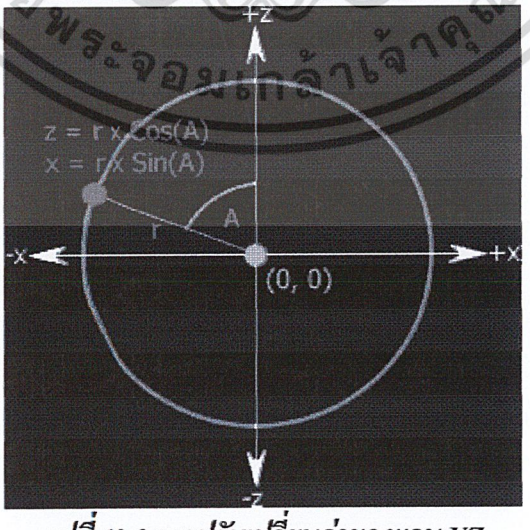
ส่วนการแหงนมองขึ้นข้างบนและมองลงข้างล่าง (หรือเรียกอีกอย่างว่า Pitch) นั้นจะเป็นการหมุนรอบแกน x เราจึงทำเหมือนการหมุนรอบแกน y คือเก็บค่าองศาของการหมุนไว้ (ในคลาสใช้ตัวแปรที่ชื่อ fAngleX) โดยการหมุนรอบแกน x มันจะหมุนแบบทวนเข็มนาฬิกา เราจึงใช้ความแรงในการเลื่อนเมาส์มาเพิ่มหรือลบให้กับค่าของตัวแปร fAngleX เหมือนกับการหมุนรอบแกน y แต่การหมุนนั้น ต้องกำหนดมุมเงยขึ้นและเงยลงด้วย ดังนั้นค่าขององศาที่จะรับได้ จะอยู่ที่ 0.0 ถึง 90.0 องศา (มองขึ้น) และ 270.0 ถึง 360.0 องศาแล้วจึงสั่งให้กล้องหมุนตามองศาของ fAngleX เพียงเท่านี้ กล้องก็จะสามารถหมุนขึ้นบนลงล่างได้แล้ว



รูปที่ 10-2 การหมุนทวนเข็มนาฬิกาของมุมรอบแกน X

10.1.3 การเคลื่อนที่ในแนวระนาบ (ซ้าย , ขวา , หน้า , หลัง)

การเคลื่อนที่ในแนวระนาบของมุมมองบุคคลที่ 1 นั้น จะเป็นการเคลื่อนที่ในแกนแนว X และแกน Z ของเวอร์สเปส (World Space) (เพราะแกน X และแกน Z นั้นตั้งฉากกันอยู่ในแนวระนาบ) โดยแกน Z จะเป็นการเคลื่อนที่เข้าหาหน้าจอ แกน – Z พุ่งไปด้านหลัง แก X อยู่ทางซ้าย และแกน – X อยู่ทางด้านขวา ซึ่งในการเคลื่อนที่นั้น ไม่ใช่ว่าถ้าให้เดินหน้า แล้วเราจะสามารถบวกค่าคงที่เข้าไปในแนวแกน Z ได้เลย เพราะการทำอย่างนั้น จะถูก ก็ต่อเมื่อมุมกล้องหันเข้าหาแกน +Z ของเวอร์สเปสตลอดเวลาเท่านั้น ถ้าเราหันไปทางอื่น เช่น หันไปทางขวา พอเวลาเดินหน้าแล้วเรา + ค่าไปแค่แกน Z จะเห็นได้ว่าตัวเราจะพุ่งไปทางซ้ายบน ไม่ใช่พุ่งเดินหน้าไปตรงๆเลย ดังนั้น จึงต้องมีการเช็คการหมุนตัว จึงต้องมีการเพิ่มและลดค่าของทั้งสองแกนพร้อมๆกันตามองศาของการหันหน้าดังรูปที่ 10-3



รูปที่ 10-3 การปรับเปลี่ยนค่าของแกน XZ

การเคลื่อนที่ในขณะที่หมุนนั้น จากรูปจะเห็นวงกลมที่มีจุดศูนย์กลาง (Origin) อยู่ที่ตำแหน่ง (0 , 0) มีรัศมี r หน่วยวิ่งไปโดยมีมุม A ดังนั้น

- ตำแหน่งของ z = r x Cos(A)
- ตำแหน่งของ x = r x Sin(A)

โดยที่ตำแหน่ง (0 , 0) นั้นคือตำแหน่งของกล้อง r คือระยะทางที่เราต้องการเคลื่อนที่ไปภายใน 1 ครั้ง (ความเร็วของการเคลื่อนที่กล้อง) และมุม A คือมุมของการหมุนรอบแกน Y หรือมุม fAngleY นั่นเอง

ส่วนการเคลื่อนที่ไปด้านข้างทางซ้ายนั้น เพียงแค่บวกมุมตามเข็มนาฬิกาไป อีก 90 องศา (ทำให้ตั้งฉากกับการเคลื่อนที่ไปข้างหน้า) แล้วจึงคำนวณแบบเดียวกับการเคลื่อนที่ไปข้างหน้า จึงจะได้สูตรว่า

- ตำแหน่งของ z = r x Cos(A+90)
- ตำแหน่งของ x = r x Sin(A+90)

ในทำนองเดียวกัน การเคลื่อนที่ไปทางด้านขวานั้น เพียงแต่ต่างกันตรงมุมที่จะบวกเข้าไป คือ ให้หันไปอีกด้านต้องลบด้วยมุม 90 องศา (หรือจะบวกเข้าไปอีก 270 องศาก็ได้) จึงได้สูตรว่า

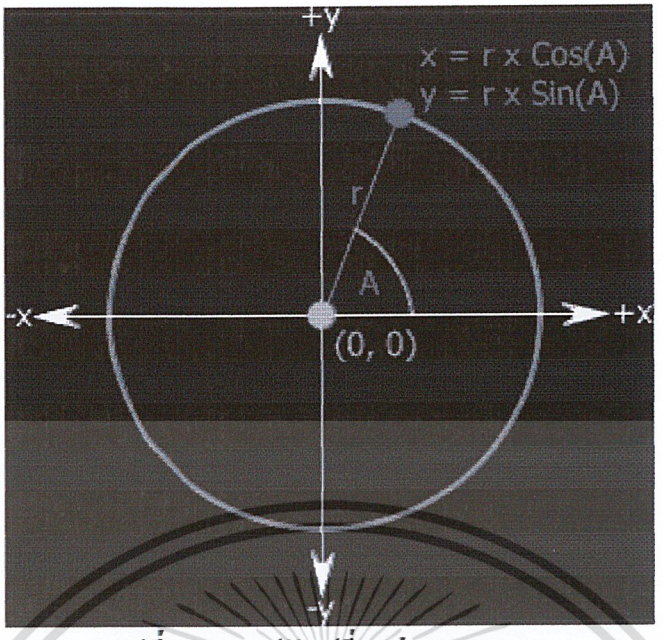
- ตำแหน่งของ z = r x Cos(A-90)
- ตำแหน่งของ x = r x Sin(A-90)

เมื่อเราได้จุด x และ z ที่เราจะเคลื่อนที่ไปแล้ว ต่อไปจึงเอาจุดนั้น ไปวางไว้ที่ตำแหน่ง Eye – Point เพื่อที่จะให้กล้อง เคลื่อนที่ไปยังทิศทางการควบคุมดังที่เราได้กำหนดไว้ ส่วน Look – At Point ก็เพียงแค่บวกตำแหน่งของกล้องในแนว Z เข้าไปเท่านั้น เพราะกล้องต้องเล็งไปข้างหน้าหรือเข้าหาจอภาพตลอดเวลา สำหรับทิศทางการตั้งของกล้องจะตั้งขึ้นในแนวแกน + y ตลอด และเนื่องจากเป็นการเดินในแนวระนาบ จึงไม่มีค่าความสูง (Y) เข้ามาเกี่ยวข้องใน Eye – Point และ Look – At Point เลย

10.2 Free – Look Camera

Free – Look Camera หรือมุมกล้องแบบอิสระคือการเคลื่อนที่แบบอิสระที่ทำให้เราสามารถบินไปได้ทั่วฉาก สามารถเคลื่อนที่ไปยังจุดไหนก็ได้ภายในเวอร์สเปสโดยใช้เมาส์และคีย์บอร์ดในการเคลื่อนที่ของกล้อง

จะเห็นได้ว่า Free – Look Camera ก็จะเหมือนการเอามุมกล้องบุคคลที่ 1 มาคำนวณเพิ่มเติมในด้านของความสูงกล่าวคือ มุมกล่องบุคคลที่ 1 สามารถเคลื่อนที่ได้ในระนาบ XZ โดยมีมุมหมุนรอบแกน Y มากำหนดทิศทาง แต่ Free – Look Camera จะสามารถเคลื่อนที่ได้ในระนาบ XYZ โดยมีมุมหมุนรอบแกน X และ Y มาช่วยคอยกำหนดทิศทาง



รูปที่ 10-4 การปรับเปลี่ยนค่าของแกน XY

จากรูปที่ 10-4 จะเห็นได้ว่า ถ้ามีการมองเงยขึ้นมา ซึ่งมุม B (ในรูปคือ A) เป็นมุมที่หมุนรอบแกน Z โดยมีรัศมี r เป็นความยาว จากสูตรของมุมมองบุคคลที่ 1 จึงสามารถเปลี่ยนเป็นสูตรของมุมกล้องแบบอิสระได้ว่า

- ตำแหน่งของ z = r x Cos(A) x Cos (B)
- ตำแหน่งของ x = r x Sin(A) x Cos (B)
- ตำแหน่งของ y= r x Cos (B)

เมื่อเราได้จุด x และ z ที่เราจะเคลื่อนที่ไปแล้ว ต่อไปจึงเอาจุดนั้น ไปวางไว้ที่ตำแหน่ง Eye – Point เหมือนการเคลื่อนที่แบบบุคคลที่ 1 และ Look – At Point ก็ทำเหมือนกันคือบวกตำแหน่งของกล้องในแนว Z เข้าไปเท่านั้น เพราะกล้องต้องเล็งไปข้างหน้าหรือเข้าหาจอภาพตลอดเวลา และเนื่องจากเป็นการเดินในแนวอิสระ จึงไม่มีค่าความสูง (Y) เข้ามาเกี่ยวข้องใน Eye – Point และ Look – At Point ต่างกับการเคลื่อนที่แบบบุคคลที่ 1 ซึ่งจะ Fix ค่า Y ไว้เลย

10.3 Camera around Object

Camera around Object หรือการเคลื่อนกล้องรอบวัตถุ การเคลื่อนที่แบบนี้ จะเหมือนการที่เราเอากล้อง ถ่ายไปยังวัตถุที่เราสนใจ แล้วหมุนกล้องเพื่อให้ได้มุมต่างๆตามที่เราต้องการ โดยที่จุดที่กล้องส่องไปจะเป็นจุดกึ่งกลางของวัตถุนั้นเสมอ ไม่ว่าวัตถุนั้นจะเคลื่อนที่ไปทางไหนก็ตาม กล้องจะเคลื่อนที่ตามไปด้วย ดังนั้น เราจะเห็นวัตถุนั้นอยู่นิ่ง แต่ฉากรอบข้างจะเปลี่ยนไป นั้นเป็นเพราะ กล้องของเราเคลื่อนที่ไปพร้อมกับวัตถุนั่นเอง

ในส่วนของการควบคุมนั้น เนื่องจากการเคลื่อนที่ของกล้องจะเคลื่อนที่ไปตามตำแหน่งของวัตถุอยู่แล้วเราจึงไม่จำเป็นต้องมีการควบคุมการเคลื่อนที่ของกล้อง แต่จะเป็นการควบคุมการหมุนรอบวัตถุซะมากกว่า ซึ่งการหมุนรอบวัตถุนั้น สามารถควบคุมได้โดยการรับมุมเข้ามาเหมือนองศาของมุมกล้องแบบอิสระ แล้วนำมาคำนวณเพื่อหาค่า ตำแหน่งที่กล้องจะเคลื่อนที่ไปโดยใช้สูตรเหมือนกับมุมกล้องแบบอิสระคือ

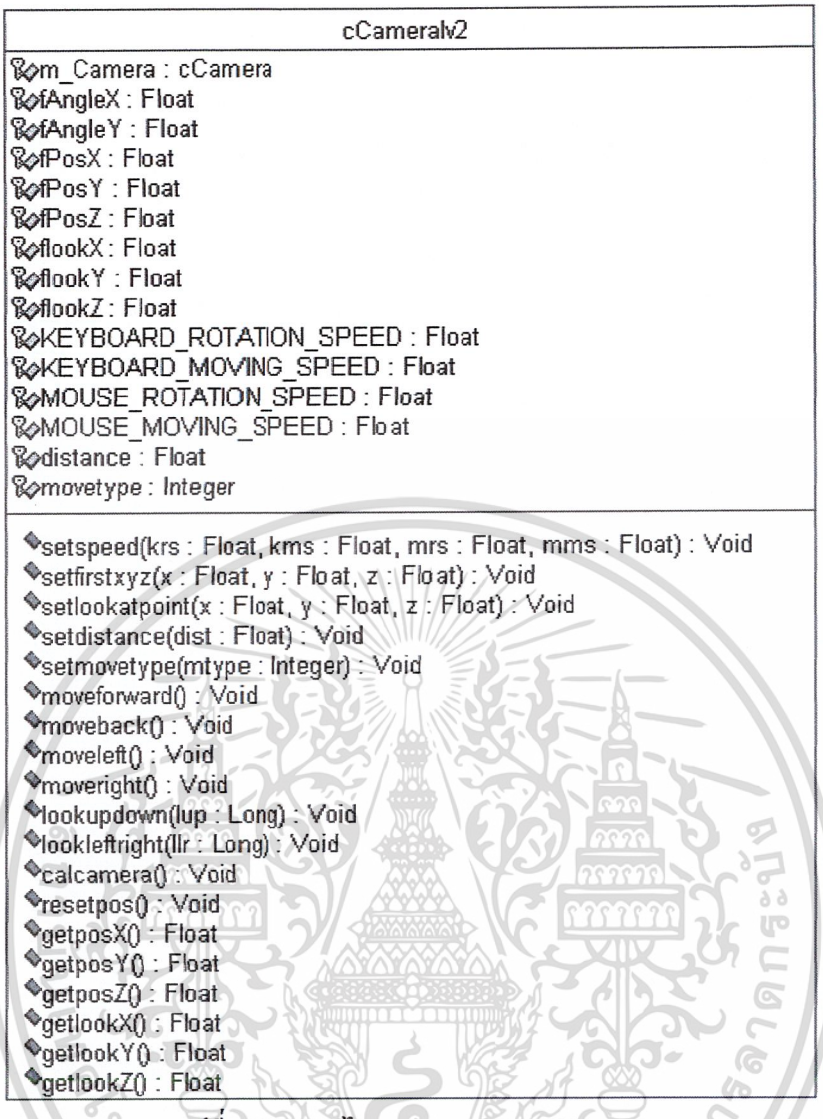
- ตำแหน่งของ z = r x Cos(A) x Cos (B)
- ตำแหน่งของ x = r x Sin(A) x Cos (B)
- ตำแหน่งของ y= r x Cos (B)

แต่เมื่อเราจะเอาตำแหน่งนั้นเข้าสู่ กล้อง จะต่างกันตรงที่ว่าตอนนี้จุดที่มอง (Look – At Point) จะไม่ใช่มองไปข้างหน้า หรือบวกตำแหน่งของกล้องในแนวแกน Z แต่จะเป็นจุดที่วัตถุอยู่ เพื่อที่จะทำให้กล้องของเรา ถ่ายไปยังวัตถุนั้นๆ ส่วนตำแหน่งที่วางกล้อง ก็ได้จากค่าตำแหน่งของ XYZ ที่เราหาจากมุม A และ B แต่เนื่องจากกล้องต้องเคลื่อนที่ตามวัตถุ เราจึงจำเป็นต้องบวกตำแหน่งของวัตถุที่เรามองอยู่เข้าไปด้วย กล้อง จึงจะสามารถเคลื่อนที่ตามวัตถุไปได้

10.4 คลาส cCameralv2

คลาส cCameralv2 นั้น เป็นคลาสที่ดึงเอาความสามารถของคลาส cCamera มาเพื่อให้มีการคำนวณการเคลื่อนไหวของกล้องทั้ง 3 แบบให้ใช้งานได้ง่ายขึ้นโดยผู้ใช้ไม่ต้องมาคำนวณว่าตำแหน่ง x , y , z และมุมที่ทำกับแกน x , y , z จะมีค่าเป็นเท่าไร เพียงแค่เช็คว่าเมื่อไรรับค่าจากเมาส์หรือคีย์บอร์ด แล้วให้เรียกว่าจะให้กล้องหมุนซ้าย , หมุนขวา , เดินหน้า , ถอยหลัง เพียงแค่นี้ในคลาส cCameralv2 ก็จะคำนวณตำแหน่ง x , y , z และมุมรอบแกน x , y , z ให้และส่งค่านั้นไปยังคลาส cCamera เพื่อนำกล้องไปวางตามตำแหน่งที่เราต้องการต่อไป

ในส่วนของค่าต่างๆในคลาส จะมีการเก็บค่ามุมที่ทำกับแนวแกน XYZ ตำแหน่งกล้องในแนวแกน XYZ เก็บค่าตำแหน่งของวัตถุที่กล้องจะจับไว้ในแนวแกน เก็บค่าความเร็วของการเคลื่อนที่และการหมุนของคีย์บอร์ดและเมาส์ และเก็บการค่าชนิดของการเคลื่อนไหวที่เราต้องการ จะให้คำนวณ



รูปที่ 10-5 คลาสไดอะแกรมของ cCameraIv2

แต่ละเมธอดมีรายละเอียดดังนี้

- setspeed() ใช้ในการกำหนดความเร็วของการหมุน และการเคลื่อนที่ให้กับคีย์บอร์ดและเมาส์
- setfirstxyz() ใช้ในการกำหนดค่าเริ่มต้น ให้กับตำแหน่งของกล้องในเวอร์สเปส
- setlookatpoint() ใช้ในการกำหนดค่าที่กล้องจะมองหรือตำแหน่งในเวอร์สเปสของวัตถุที่เราต้องการจะติดตามในมุมมองแบบ Camera around Object
- setdistance() ใช้ในการกำหนดค่าความห่างระหว่างกล้องกับวัตถุที่เราจะหมุนรอบ
- setmovetype() กำหนดชนิดของการเคลื่อนที่ว่าให้เคลื่อนที่แบบไหน โดยที่

  0 = First – Person View

  1 = Free – Look Camera

  2 = Camera around Object

- moveforward(), moveback(), moveleft(), moveright()ให้กล้องเคลื่อนที่ไปตามความเร็วที่กำหนดไว้ในเมธอด setspeed
- lookupdown() สั่งให้กล้องหมุนซ้ายหรือขวาตามความเร็วที่กำหนดไว้ในเมธอด setspeed และมีการรับค่าจากเมาส์มาเพื่อบอกแรงของการหมุนด้วย
- lookleftright() ให้คำนวณการเคลื่อนไหวของจุด X , Y , Z ตามชนิดของการเคลื่อนที่ของกล้อง (Movetype) และจึงนำค่าเหล่านั้น เรียกใช้ในเมทอดที่สืบทอดมาจาก คลาส cCamera ของตัวแปร m_Camera นั่นเอง
- calcamera() ให้คำนวณการเคลื่อนไหวของจุด X , Y , Z ตามชนิดของการเคลื่อนที่ของกล้อง (Movetype) และจึงนำค่าเหล่านั้น เรียกใช้ในเมทอดที่สืบทอดมาจาก คลาส cCamera ของตัวแปร m_Camera นั่นเอง
- resetpos() สั่งให้ตำแหน่งต่างๆ ของกล้อง กลับมาอยู่ที่จุดเริ่มต้น
- getposX, getposY, getposZ ในการรีเทริน (return) ค่า fPosX, fPosY, fPosZ ตามลำดับ
- getlookX, getlookY, getlookZ ใช้ในการรีเทรินค่า flookX, flookY, flookZ ตามลำดับ

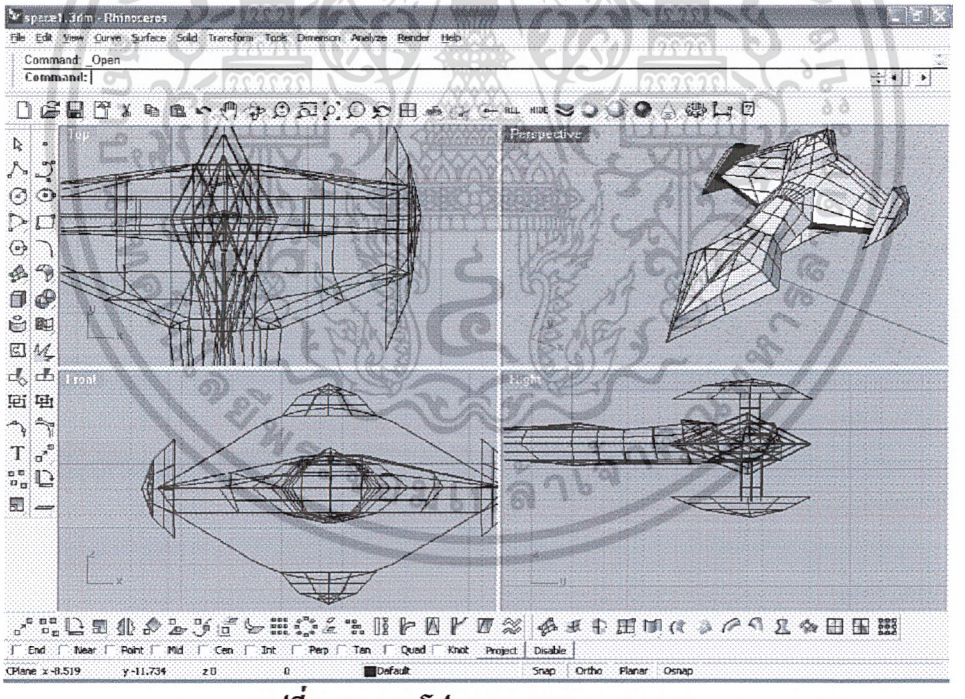
บทที่ 11
การทำโมเดล

โมเดลที่ใช้ในโครงงานนี้ ส่วนใหญ่จะเป็นรูปของยาน ซึ่งการจะสร้างจะต้องอาศัยเครื่องมือหรือโปรแกรมช่วย ซึ่งใช้โปรแกรมดังต่อไปนี้

- Rhinoceros 3.0
- 3D Studio Max 5.0
- 3D Exploration
- Adobe Photoshop

11.1 การขึ้นรูปโมเดลด้วย Rhinoceros 3.0

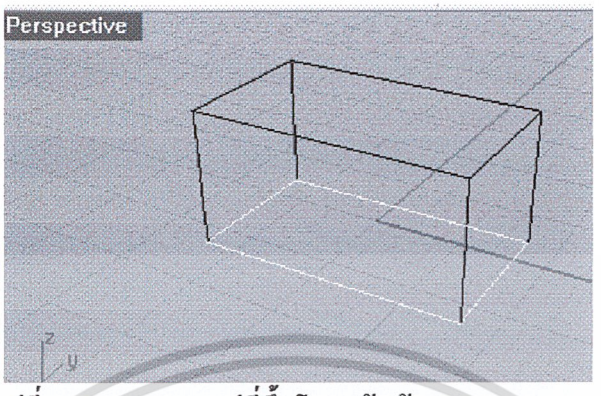
เหตุผลที่เลือกใช้โปรแกรมนี้ เนื่องจากความถนัด ความง่ายในการขึ้นรูปชิ้นงาน คำสั่งไม่ซับซ้อนมาก ซึ่งโปรแกรมนี้จะมีลักษณะหน้าตาดังรูปที่ 11-1



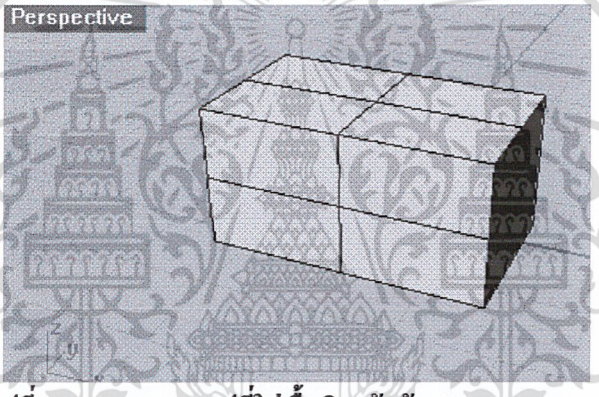
รูปที่ 11-1 ภาพโปรแกรม Rhinoceros 3.0

โปรแกรมนี้จะเซฟเป็นไฟล์ที่มีนามสกุล 3dm เป็นหลัก และสามารถนำเข้าและส่งออกรูปแบบของไฟล์ที่เป็นรูปแบบอื่น ๆ เพื่อใช้งานกับโปรแกรมทางด้าน 3 มิติมากมาย เช่น 3D Studio Max , LightWave เป็นต้น ในส่วนของโครงงานชิ้นนี้เริ่มต้นจะใช้โปรแกรมนี้แค่การขึ้นรูปชิ้นงานเท่านั้น ซึ่งแม้ว่าโปรแกรม 3D Studio Max 5.0 สามารถสร้างโมเดลได้เหมือนกัน โดยจะสร้างจากวัตถุแบบ 2 มิติ หรือ 3 มิติ รูปทรงง่ายๆ แล้วนำมาปรับแต่งรูปร่างให้เป็นไปตามต้องการ ซึ่งต้องอาศัยความชำนาญในการ

ขึ้นรูป แต่ในส่วนของโปรแกรม Rhinoceros 3.0 การสร้างโมเดลจะทำโดยการสร้างเส้น 2 มิติลากต่อกันขึ้นเป็นโครงรูปร่างก่อนดังรูปที่ 11-2 แล้วค่อยก็สร้างพื้นผิวจากโครงอีกทีดังรูป 11-3



รูปที่ 11-2 ภาพแสดงรูปที่ขึ้นโครงเส้นด้วย Rhinoceros 3.0

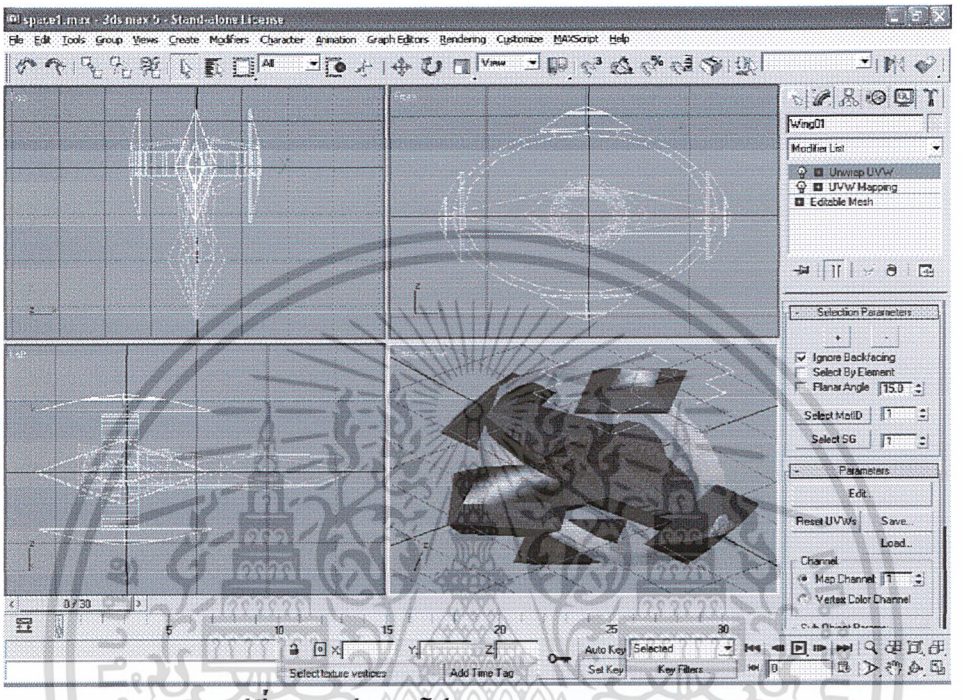


รูปที่ 11-3 ภาพแสดงรูปที่ใส่พื้นผิวแล้วด้วย Rhinoceros 3.0

แม้ว่าการใช้โปรแกรม Rhinoceros จะสามารถทำโมเดลออกมาได้ แต่เนื่องจากโปรแกรมมีข้อจำกัด คือมีลูกเล่นไม่มาก และรูปแบบไฟล์ไม่สนับสนุนการนำไปแปลงเป็นไฟล์สำหรับทำเกม ดังนั้นจึงต้องมีการนำมาแปลงเป็นรูปแบบของ 3D Studio Max ซึ่งแปลงโดยนำรูปทรงที่มีพื้นผิวจาก Rhinoceros ทำการส่งออกด้วยคำสั่ง File -> Export Selected ในเมนูทำการจัดเก็บเป็นรูปแบบ 3ds ซึ่งเป็นรูปแบบไฟล์ของโปรแกรม 3D Studio Max เพื่อนำไปปรับแต่งอีกที ให้เหมาะสม

11.2 การปรับแต่งและใส่พื้นผิวด้วย 3D Studio Max 5.0

โปรแกรม 3D Studio Max เป็นโปรแกรมที่มีคนใช้งานอย่างแพร่หลายมาก มีคำสั่งการใช้งานที่มากมาย ยืดหยุ่นในการใช้งานสูง ดังนั้นการใช้โปรแกรมนี้จึงเป็นทางเลือกหนึ่งของคนที่พัฒนาเกม ซึ่งลักษณะหน้าตาของโปรแกรมจะเป็นดังรูปที่ 11-4 ซึ่งเป็นเวอร์ชั่นที่ 5

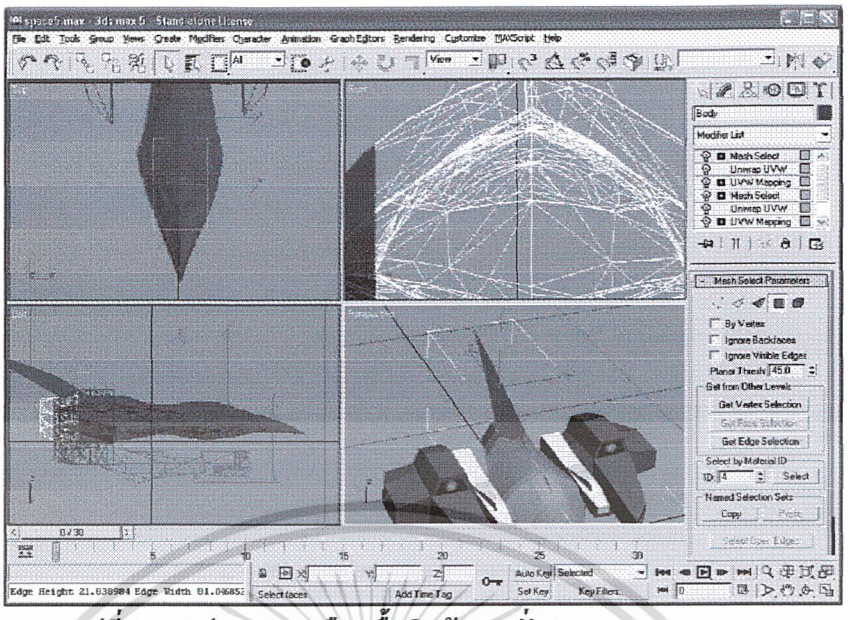


รูปที่ 11-4 รูปแสดงโปรแกรม 3D Studio Max 5.0

ในส่วนของโครงงานนี้ เมื่อนำเข้าไฟล์ 3ds ด้วยเมนู Files -> Import แล้วจะได้เป็นโมเดลที่ใกล้เคียงกับที่ทำด้วยโปรแกรม Rhinoceros อาจมีข้อแตกต่างในเรื่องของลักษณะพื้นผิว ซึ่งหากเมื่อแปลงมาอาจมีลักษณะการเชื่อมต่อของพื้นผิวไม่สมบูรณ์ เป็นรู ก็ต้องนำมาปรับแต่งให้สมบูรณ์อีกครั้งด้วย 3D Studio Max

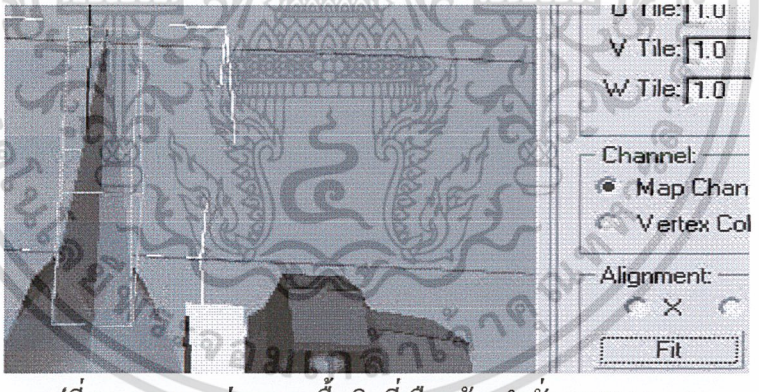
เมื่อปรับแต่งจนสมบูรณ์แล้วก็จะถึงขั้นใส่พื้นผิววัตถุ ในส่วนนี้จะเป็นการแสดงการใส่พื้นผิวตามพื้นผิวเฉพาะที่เลือก ไม่ใช่ทั้งหมด ดังนั้นจึงมีขั้นตอนที่ซับซ้อนเล็กน้อย และทำดังนี้

เลือกไปที่วัตถุที่จะใส่พื้นผิวจากนั้นเลือกไปที่ปุ่ม Modify  แล้วที่ Modifier List เลือกไปที่ Mesh Select จากนั้นที่แถบของ Mesh Select Parameter ให้กดที่ปุ่ม Polygon  แล้วเลือกพื้นผิวที่ต้องการจะใส่รูปพื้นผิว ดังรูปที่ 11-5



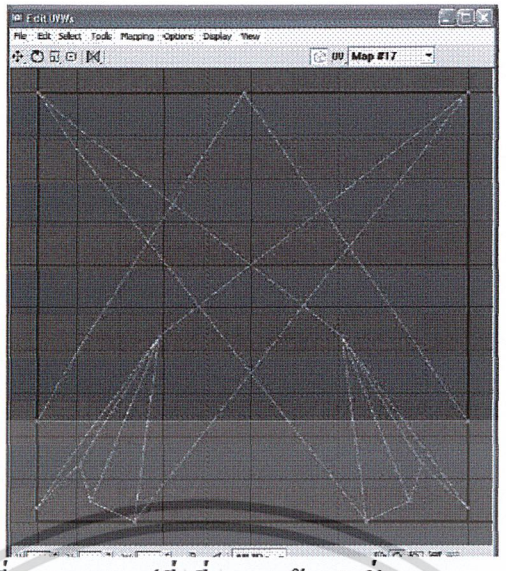
รูปที่ 11-5 รูปแสดงการเลือกพื้นผิวด้วยคำสั่ง Mesh Select

จากนั้นที่ Modifier List เลือกไปที่ UVW Mapping เพื่อกำหนดลักษณะการคลี่ของวัตถุที่เลือก โดยปรับค่าใน Parameters ในที่นี้เลือกลักษณะการคลี่เป็น Box แล้วใน Alignment กดที่ปุ่ม Fit เพื่อให้ได้กล่องครอบคลุมเฉพาะส่วนที่เราเลือกไว้ จะได้ผลดังรูปที่ 11-6



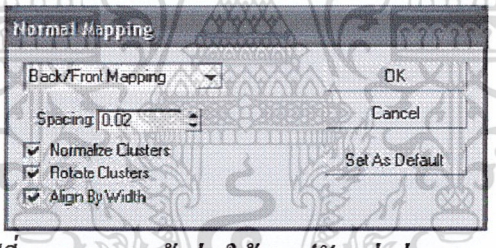
รูปที่ 11-6 แสดงกล่องคลุมพื้นผิวที่เลือกด้วยคำสั่งUVW Mapping

จากนั้นไปที่ Modifier List อีกครั้ง แล้วเลือก Unwarp UVW เพื่อทำการคลี่กล่องวัตถุที่เราเคยเลือกไว้ โดยกดที่ปุ่ม Edit ตรง Parameter จะขึ้นหน้าต่าง ดังรูปที่ 11-7

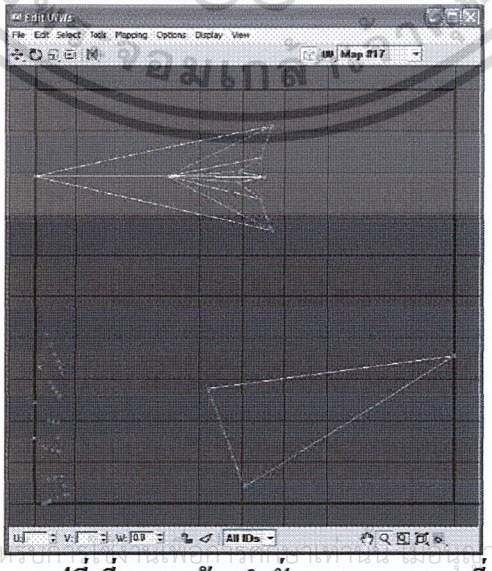


รูปที่ 11-7 แสดงรูปที่คลี่ออกมาด้วยคำสั่ง Unwarp UVW

รูปที่คลี่ออกมานี้อาจดูยาก หรือทับกัน เราอาจเปลี่ยนมุมมองการคลี่ หรือย้ายขยับเส้นไปมาได้ และเปลี่ยนมุมมองโดยเลือก Mapping ของหน้าต่างรูป 11-7 จะปรากฏหน้าต่างดังรูป 11-8 ให้ปรับค่า โดยใน List จะเป็นเลือกมุมมอง เช่น Back/Front Mapping คือแสดงรูปคลี่ด้านหน้าและด้านหลัง ส่วน Spacing เป็นค่ากำหนดระยะห่างของวัตถุที่คลี่ออกมา

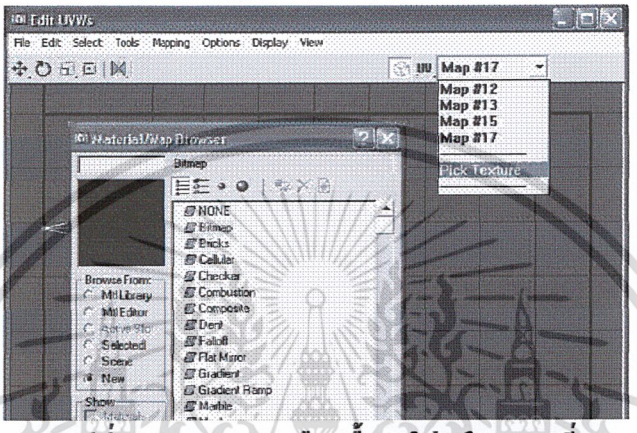


รูปที่ 11-8 แสดงหน้าต่างให้การปรับแต่งค่า Mapping

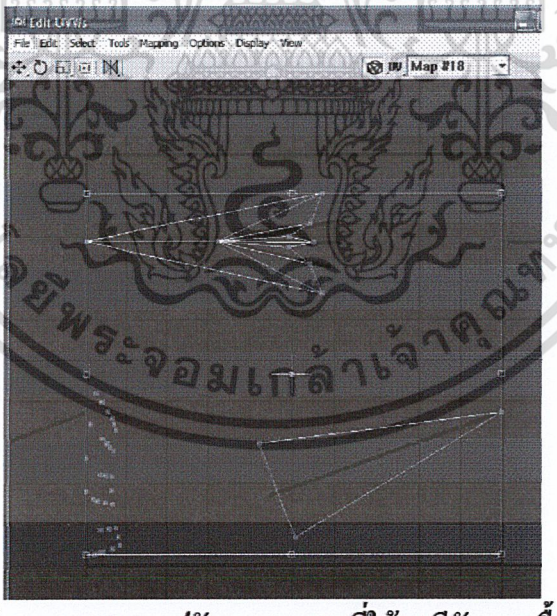


รูปที่ 11-9 แสดงรูปที่คลี่ออกมาด้วยคำสั่ง Unwarp UVW ที่ปรับแต่งแล้ว

เมื่อได้ที่พอใจแล้ว ก็ทำการ Print Screen หน้าจอรูปที่ 11-9 และนำมาวางและปรับแต่งพื้นผิวที่จะใส่ด้วยโปรแกรม Adobe Photoshop ซึ่งขั้นตอนการทำพื้นผิวนี้หากทำดีๆ โมเดลก็จะออกมาดู เมื่อเสร็จแล้วก็กลับมาที่โปรแกรม 3D Studio Max แล้วทำการ Pick Texture และเลือก Bitmap และเลือกรูปผ่านการทำ Texture ด้วย Adobe Photoshop แล้วดังรูปที่ 11-10 และปรับขนาดของภาพคลี่ให้พอดีกับพื้นผิวที่เลือกมา ดังรูปที่ 11-11 ตามลำดับ

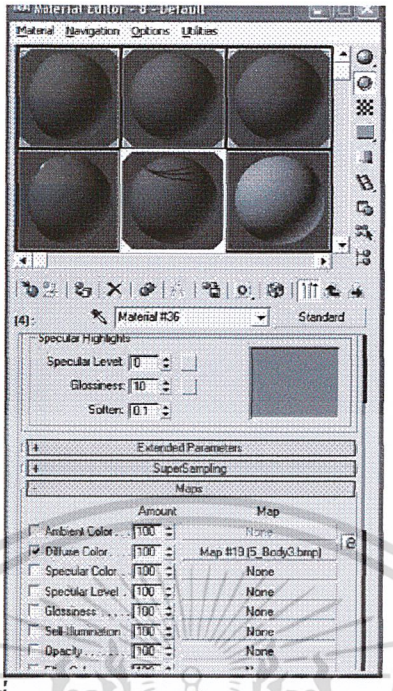


รูปที่ 11-10 แสดงการเลือกพื้นผิวใส่ลงในภาพคลี่



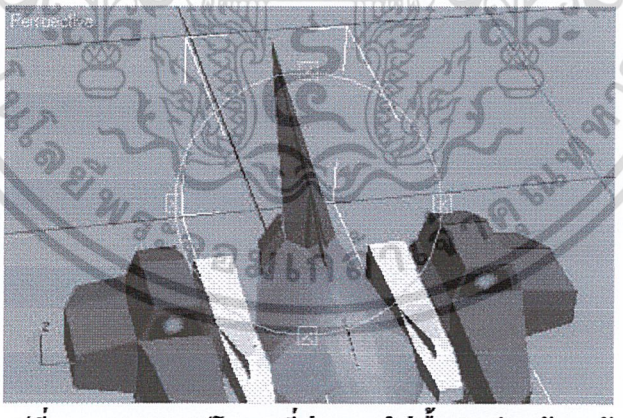
รูปที่ 11-11 แสดงการปรับขนาดภาพคลี่ให้พอดีกับภาพพื้นผิว

จากนั้นก็ทำการเลือก Material โดยกดที่แถบด้านบน เลือก Rander -> Material Editer จะปรากฏหน้าต่างดังรูปที่ 11-12



รูปที่ 11-12 แสดงการทำงาน Material Editer

ที่ตรง Map ด้านหลัง Diffuse Color เลือกรูปที่ต้องการจะทำเป็นภาพที่เป็นพื้นผิว แล้วกดที่ปุ่ม Assign Material to Selection  เพื่อทำการใส่พื้นผิวที่เลือกลงไปในวัตถุ และกด Show Map in Viewport  เพื่อให้แสดงพื้นผิวคร่าวๆ ในวัตถุ ซึ่งการทำนี้ จะได้โมเดลที่แปะภาพพื้นผิวเรียบร้อยดังรูปที่ 11-13

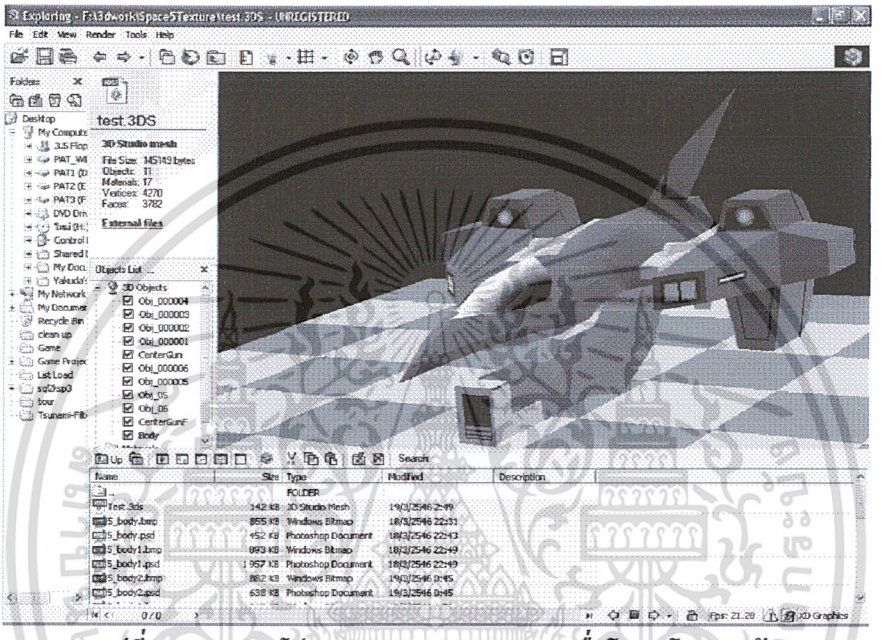


รูปที่ 11-13 แสดงรูปโมเดลที่ผ่านการใส่พื้นผิวเรียบร้อยแล้ว

11.3 การแปลงไฟล์เป็นรูปแบบของ DirectX ด้วยโปรแกรม 3D Exploration

เมื่อทำการปรับแต่งวัตถุและใส่พื้นผิวเรียบร้อยตามต้องการ ก็นำมาแปลงให้อยู่ในรูปแบบ .X ในโครงงานนี้ได้เลือกใช้โปรแกรม 3D Exploration

เริ่มจากโปรแกรม 3D Studio MAX ให้ส่งออกมาในรูปแบบไฟล์ 3ds ก่อน จากนั้นก็ใช้โปรแกรม 3D Exploration เปิดไฟล์ 3ds นั้นขึ้นมา จากรูปที่ 11-14 จะเป็นลักษณะหน้าตาของโปรแกรมนี้ และภาพที่เห็นแสดงการเปิดโมเดลไฟล์ 3ds ขึ้นมาแสดง



รูปที่ 11-14 แสดงโปรแกรม 3D Exploration เมื่อโหลดโมเดลแล้ว

จากนั้น ก็ทำการเลือก Save As... แล้วเลือกรูปแบบไฟล์เป็น .X ก็จะได้ไฟล์ที่พร้อมจะนำไปใส่ในตัวเกม

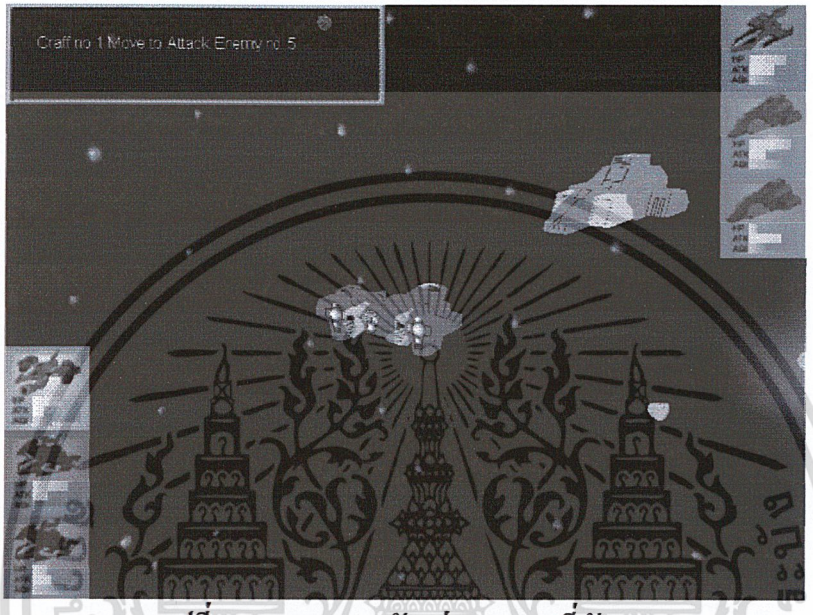
บทที่ 12
การพัฒนาเกมเพื่อทดสอบเกมเอนจิน

12.1 ประเภทเกมที่พัฒนา

เกมที่พัฒนานั้น เป็นเกมแนว Real-time 3D Simulation shooting โดยมีมุมมองของกล้องในแบบ Camera around Object หรือกล้องจะหมุนอยู่รอบยานที่จะบังคับโดยอัตโนมัติ การบังคับอัตโนมัติที่ว่านี้ คือ เพียงแค่เรากำหนดให้ยานของเรา วิ่งเข้าไปโจมตียานของศัตรูลำไหน แล้วยานของเรา ก็จะวิ่งเข้าไปเพื่อโจมตีศัตรูได้โดยอัตโนมัติ โดยเกมแบบ Real-time 3D Simulation shooting ที่ว่านี้ มีความหมายดังนี้

- Real-time หมายถึงช่วงเวลาของการเล่นเป็นแบบเวลาต่อเนื่องไม่มีหยุด ถึงแม้เราจะสั่งให้ยาน A วิ่งไปโจมตียาน B แล้วเราเปลี่ยนมุมกล้องมามองที่ยาน C แต่ยาน A นั้น ก็ยังจะวิ่งไปโจมตียาน B ต่อไปโดยไม่สนใจสิ่งรอบข้าง ความสนุกของตรงนี้ คือการแข่งขันกับเวลา ว่าเราจะสามารถบังคับยานแต่ละลำได้อย่างพร้อมๆกันหรือไม่ ถ้าเราลืมบังคับยานลำไหน ยานลำนั้นก็จะเสียเวลาการเคลื่อนที่ หรือหยุดนิ่งรอศัตรูมาโจมตีจนกระทั่ง Game Over ได้
- 3D หมายถึงเกมที่พัฒนานั้น มีระนาบเป็นแบบ 3 มิติ สามารถดูตัวละครได้ทุกด้าน ไม่จำกัดเพียงแต่เฉพาะด้านหน้าหรือด้านหลัง
- Simulation หมายถึงเกมประเภทวางแผน ในเกมนี้ จะเป็นการวางแผนการรบการซึ่งเราจะเอาชนะเกมนี้ได้นั้นขึ้นอยู่กับว่าเราสามารถวางแผนใช้ตัวละคร , ความสามารถของตัวละครได้อย่างดีหรือไม่ ถ้าเราวางแผนได้ดี ก็จะสามารถชนะเกมได้ง่าย เช่น ให้ตัวละครทุกตัวรุมโจมตีตัวที่อ่อนที่สุดก่อนเพื่อเป็นการตัดกำลังรบของคู่ต่อสู้ จะดีกว่าแต่ละคนแยกกันไปโจมตีเพราะการที่เราช่วยกันโจมตีย่อมดีกว่าที่เราแยกกันโจมตีอยู่แล้ว และเนื่องจากเกมนี้ เป็นแบบ Real-time เราจึงต้องคิดอย่างรวดเร็วว่าให้ยานลำไหนทำหน้าที่อะไร โดยต้องคิดแข่งกับเวลาเพื่อที่จะได้แนวทางการต่อสู้ที่ดีที่สุด
- Shooting เกมนี้เป็นเกมวางแผนการรบประเภท Shooting คือการยิง คือบังคับให้ยานของเราไปยิงกับยานลำอื่นๆ แต่ไม่ใช้ให้เราบังคับเองตามแบบที่กล่าวไว้ข้างต้น อนึ่ง ที่ต้องบอกว่าเป็นเกมแนว Shooting เพราะเกมแบบ Simulation มีหลายแบบ เช่น เกมประเภท Sim City จะเป็นการวางแผนการสร้างเมือง เกมสามก๊ก เป็นเกมที่วางแผนในการทำสงครามยึดเมือง เป็นต้น

นอกจากนี้ ปัจจัยหลักที่ทางกลุ่มผมเลือกเอาเกมประเภทนี้มาสร้างนั้นก็คือเกมชนิดนี้ มีความสนุกนอกจากการที่เราจะได้วางแผนการรบแบบ Real – time ยังสามารถใช้มุมกล้องที่หมุนรอบวัตถุ เพื่อดูความสวยงามของโมเดล หรือการเคลื่อนไหวผ่านทางตัวเกม เหมือนดูหนังการต่อสู้ทางยานอยู่เรื่องนึ่ง แตกต่างกันตรงที่ว่า เราสามารถที่จะย้ายกล้อง ไปยังจุดต่างๆ เพื่อที่จะดูรายละเอียดทั้งหมดของเกมได้ ซึ่งในจุดนี้ การดูหนังธรรมดาไม่สามารถทำได้



รูปที่ 12-1 ภาพแสดงตัวอย่างของเกมที่พัฒนา

12.2 อุปกรณ์และโปรแกรม ในการพัฒนาเกม

อุปกรณ์ฮาร์ดแวร์ของเครื่อง Computer

- CPU Pentium 4 1.8 GHz
- DDRRam 256 MB
- VGA : Geforce 2MX 32Mb / Geforce 4MX 64Mb DDR

อุปกรณ์ซอฟแวร์ในการพัฒนาเกม

- OS Microsoft Windows XP/Microsoft Windows 2000 Advanced Server
- ซอฟแวร์ในการเขียน โปรแกรม
  - Microsoft Visual C++ 6.0
  - Microsoft DirectX 8.1 SDK
- ซอฟแวร์ในการพัฒนากราฟฟิก
  - Rhinoceros 3.0
  - 3D Studio Max 5.0
  - Adobe Photoshop 7.0
  - 3D Exploration

12.3 ขั้นตอนการพัฒนาเกม

เมื่อกำหนดได้ว่าจะทำเกมเป็นประเภท Real-time 3D Simulation Shooting แล้ว ซึ่งส่วนหลัก ที่จะทำให้เกมนี้สนุกได้นั้น คือการเคลื่อนไหวของกล้องต้องราบรื่น และไม่มีข้อผิดพลาด (Bug) ในการหมุนแต่ละครั้ง และสามารถมองได้ทุกทิศทาง ดังนั้น จึงเริ่มจากการพัฒนาคลาส cCameralv2 สืบทอดต่อจากคลาส cCamera และสิ่งที่ได้รับคือ เวลาเรียกโค้ดเราเพียงแค่เขียนโค้ดหลัก เพียงไม่กี่บรรทัด ก็สามารถ เรียกใช้กล้องได้แล้ว เพียงแค่กำหนดจุดเริ่ม จุดที่มอง และรับค่าการเคลื่อนไหวเท่านั้นเอง

ในส่วนของโมเดล ขั้นแรกก็จะมีการออกแบบรูปร่างหน้าตาของยานก่อน และมองความเป็นไปได้ในการขึ้นรูปให้เป็น 3 มิติ ถ้ามีความเป็นไปได้ในการสร้าง ก็จะนำมาขึ้นรูปด้วยโปรแกรม Rhinoceros โดยการขึ้นโครงเป็นเส้น และก็ใส่ให้มีพื้นผิว จากนั้นทำการส่งออกไปยังโปรแกรม 3D Studio Max เพื่อทำการแต่งวัตถุและใส่รูปพื้นผิวและใช้งานควบคู่กับโปรแกรม Adobe Photoshop ในการสร้างตกแต่งภาพพื้นผิวที่จะนำมาใส่กับวัตถุโมเดลนั้น เมื่อทำการปรับแต่งและใส่พื้นผิวเป็นที่เรียบร้อยหมดแล้ว ก็จะส่งไปให้โปรแกรม 3D Exploration ซึ่งจะทำหน้าที่แปลงจากไฟล์ .3ds ให้เป็น ไฟล์ .X ซึ่งเป็นไฟล์โมเดลที่พร้อมที่จะนำมาใช้ในเกมได้

เมื่อได้โมเดล 3 มิติที่เป็นรูปแบบ .X แล้ว สามารถที่จะเอามาแสดงผลลงในตัว โปรแกรม ได้ หลังจากนั้น จึงเป็นการออกแบบระบบของเกม ออกแบบ อินเตอร์เฟส ในการติดต่อกับผู้ใช้ ออกแบบคลาสที่ใช้ช่วยในการเขียนเกม ซึ่งคลาสที่เขียนนี้ คือ คลาส Craff ในที่นี้ไม่ขอกล่าวถึงรายละเอียด และความสามารถต่างๆของเมธอดเพราะเป็นการออกแบบ เพื่อให้ง่ายต่อการเขียนเกมในด้านนี้เท่านั้น ไม่สามารถจัดว่าเป็นเอนจินได้ แต่ถ้าต้องการศึกษา เพื่อนำไปประยุคใช้ ก็สามารถดูโค้ดแล้วนำไปปรับปรุงแก้ไขตามที่ต้องการได้

เมื่อเขียนคลาสเสร็จสิ้น ส่วนต่อไปคือการเขียนส่วนติดต่อกับโปรแกรม หรืออินเตอร์เฟสของโปรแกรมโดยเป็นการรับค่าจากเมาส์และคีย์บอร์ดมาเพื่อติดต่อกับคลาสที่เราสร้างขึ้น

เมื่อสร้างส่วนติดต่อเสร็จ จึงนำโมเดลที่สร้างไว้ในขั้นตอนที่ 2 มาใส่ลงไปในเกม เพื่อทดลองรันเกมและดีบักโปรแกรมเพื่อหาข้อผิดพลาดต่างๆ และแก้ไขต่อไป

บทที่ 13
บทวิจารณ์และสรุป

13.1 สรุปผลการทำโครงงานและความสำคัญของเอนจิน

การพัฒนาเกม 3 มิติ บนเครื่องคอมพิวเตอร์ในปัจจุบันนั้น สามารถพัฒนาได้โดยไม่ยากเย็นมากนัก โดยการพัฒนานั้น ระดับล่างสุด จะเป็นการเรียกใช้ฮาร์ดแวร์ของไดเร็กเอ็กซ์ซึ่งการเรียกใช้ไดเร็กเอ็กซ์ นั้น มีวิธีการเรียกใช้ , รายละเอียดในการเรียกใช้ , ค่าต่างๆที่จะกำหนดให้กับไดเร็กเอ็กซ์ และค่าต่างๆที่จะรีเทริน (Return) กลับมาจากไดเร็กเอ็กซ์อยู่หลายแบบ การพัฒนาเกม 1 เกมโดยใช้โค้ดไดเร็กเอ็กซ์ การเขียนจึงจำเป็นต้องมีความเข้าใจในส่วนนี้อยู่สูงพอสมควร และเนื่องจากว่าทางไมโครซอฟต์ได้พัฒนาไดเร็กเอ็กซ์ขึ้นมาเพื่อครอบคลุมการเขียนเกมทุกส่วน ไม่ว่าจะเป็นในด้านรายละเอียดเล็ก ๆ น้อย ๆ ซึ่งบางที เราสามารถที่จะไม่ใช้ความสามารถในส่วนนั้นของไดเร็กเอ็กซ์ได้ และเพื่อง่ายแก่การพัฒนาเกมจริงๆเราจึงควรที่จะเขียนเอนจินขึ้นมาเพื่อครอบคลุมความสามารถทั้งหมดของไดเร็กเอ็กซ์ แล้วดึงเอาเฉพาะส่วนที่เราต้องการเขียนเกมมาทำเป็นเอนจินซึ่งในโครงงานที่กลุ่มเกมในปีการศึกษา 2545 ได้จัดทำนั้น จะมีเอนจิน หลักที่ทุกคนใช้ร่วมกัน และแต่ละกลุ่ม ก็จะมีเอนจินของแต่ละกลุ่มเพื่อเขียนโปรแกรมเกมของตัวเองให้ง่ายขึ้น ดังนั้นเอนจินจึงมีความสำคัญต่อการพัฒนาเกมมาก

13.2 แนวทางในการพัฒนาต่อ

การพัฒนาโครงงานเกมที่เกิดขึ้นในมหาวิทยาลัยนั้น โดยเฉพาะสาขาวิศวกรรมคอมพิวเตอร์ การเขียนโค้ดและการออกแบบเกมนั้น สามารถที่จะพัฒนาไปได้เรื่อย ๆ โดยที่ตอนนี้ ยังดึงเอาความสามารถของ ไดเร็กเอ็กซ์ ออกมาใช้ได้ไม่หมด แต่เป็นเพียงแค่การดึงบางส่วนเอามาใช้ ซึ่งบางส่วนที่ว่านี้ เป็นส่วนที่ทำ แล้วออกมาดูเป็นเกม สามารถเล่นได้ ดังนั้น แนวทางในการพัฒนาต่อ อาจจะเป็นการพัฒนาในจุดย่อยๆ เช่น การพัฒนาในส่วนของแสง การสะท้อนของแสง พื้นผิวต่างๆ การเคลื่อนไหวโดยมีลมพัด โดยจะต้องพัฒนาเอนจินขึ้นมา เพื่อช่วยในส่วนนี้ ซึ่งการทำตรงนี้นั้น ต้องเขียนโปรแกรมอย่างมากมาย แต่ผลที่ได้ อาจเป็นเพียงแค่ส่วนนิดเดียวในเกมเท่านั้น จึงไม่ค่อยมีคนให้ความสนใจมากเท่าไร

อีกส่วนหนึ่งคือในส่วนของกราฟิก ซึ่งในจุดนี้ในฐานะที่คนพัฒนาเกมและศึกษาอยู่ในสาขาวิศวกรรมคอมพิวเตอร์นั้น ความรู้ในด้านการสร้างโมเดล การทำกราฟิก อาจรู้ไม่เพียงพอ หรืออาจต้องไปหาโมเดล หรือเท็กซ์เจอร์ (Texture) มาจากต่างเว็ปไซต์ต่างประเทศ ดังนั้น ถ้ามีการร่วมมือกันระหว่างคณะวิศวกรรมศาสตร์กับ คณะสถาปัตยกรรมศาสตร์ เพื่อคณะสถาปัตยกรรมศาสตร์พัฒนาด้านกราฟิก ส่วนทางคณะวิศวกรรมศาสตร์ก็พัฒนาทางด้านโปรแกรม เมื่อเอาผลงานมารวมกัน ก็อาจจะทำให้โครงงานที่ออกมามีความสวยงามและมีคุณภาพอย่างมากก็ได้

ภาคผนวก

ภาคผนวก ก

ชนิดของไฟล์โมเดล 3 มิติ

เนื่องจากปัจจุบันนี้ มีซอฟแวร์ช่วยในการทำโมเดลอยู่หลายแบบ แต่ละแบบ ก็จัดเก็บกันตามมาตรฐานของโปรแกรมนั้นๆ แต่เราสามารถเอาโมเดล 3 มิติ ที่ได้จากการทำในซอฟแวร์ต่างๆ นำมาลงในเกม 3 มิติ ที่เราสร้างขึ้นได้ โดยการแปลงไฟล์เหล่านั้นให้อยู่ในรูปแบบ .x (Direct X)

1. แนะนำโปรแกรมที่ใช้แปลงรูปแบบไฟล์

โปรแกรมที่ช่วยแปลงรูปแบบมีมากมาย แต่โปรแกรมที่ใช้ในโครงงานนี้คือ 3D Exploration ซึ่งเป็นโปรแกรมที่ช่วยในการจัดการแปลงรูปแบบไฟล์ชนิดต่างๆ ให้อยู่ในรูปแบบไฟล์ในแบบที่เราต้องการ (รวมถึงของไดเร็กเอ็กซ์) เพื่อที่จะนำเอาไปใช้เช่น เราสร้างไฟล์ 3 มิติด้วย 3D Studio Max จะได้ไฟล์ออกมาในรูปแบบ .3ds แล้วจึงนำมาเข้าสู่ 3D Exploration เพื่อแปลงให้เป็นรูปแบบ .x แล้วนำมาลงในเกมของเราต่อไป ในส่วนของโปรแกรมนี้เป็นแชร์แวร์ให้ทดลองใช้ได้ 30 วัน สามารถหาข้อมูลเพิ่มเติมและดาวน์โหลดตัวโปรแกรมมาทดลองใช้ได้ที่ http://www.righthemisphere.com/3dexploration/

2. ชนิดของไฟล์ที่โปรแกรมสนับสนุน

หมายถึงประเภทของ ไฟล์ ที่สามารถเอาเข้ามาแปลงรูปแบบได้ ซึ่งโปรแกรม 3D Exploration ได้สนับสนุน ไฟล์ หลายรูปแบบ โปรแกรมจะสนับสนุนการนำเข้าและส่งออกดังนี้

2.1 การนำเข้าไฟล์ (Import File)

การนำเข้าไฟล์หน้าที่หลักของโปรแกรมก็คือทำการเปิดไฟล์ในรูปแบบอื่น ๆ ได้ เพื่อนำมาปรับปรุงแก้ไขได้ ซึ่งรูปแบบที่นำเข้าได้มีดังนี้

2.1.1 รูปแบบการนำเข้าไฟล์ 3 มิติ (3D Import)

นามสกุลของไฟล์	ชนิดของไฟล์	Note
3dm	Rhinoceros	all NURBS and mesh objects
3dm,3dmf	3D Metafile	binary files only, partially supported
3ds	3D Studio Binary	
3ds max	3ds max import/export	
Asc	3D Studio ASCII	

Ase	3D Studio MAX ASCII Scene Export	with animation support
c4d	Cinema 4D	version 6.0 not supported
Cob	Caligari objects	binary and ASCII, with animation
Dwg	AutoCAD DWG	ACIS SAT not supported
dxf,dxb	AutoCAD DXF	
Flt	OpenFlight Scene Description Database	Multigen API DLLs not required
Geo	AOFF File Format	
Iob	Imagine	
Iso	ISO G Code	
Iv	Open Inventor files	ASCII only, partially supported
lwo,lw	LightWave objects Lightwave 3D layered objects Lightwave 6.0 objects	Lightwave files + UView files supported
Lws	Lightwave scenes	with animation support
Map	Quake map files	textures are not supported yet
md2	Quake II model	with animation support
md3	Quake III model	
Mdl	Half Life model	with animation support
Mdl	Quake I model	
Mts	Metastream file	
mts,mtx,mtz	VET file	
Ndo	Nendo v1.0/v1.1	
Obj	Alias WaveFront objects	
Objf	Stripe objf file format	
Off	Object file format	
peo,geo	Homeworld geometry	with LiF textures
Pgm	16 bit PGM file format	
Prj	3D Studio Project	
Pro	Power Render	
Raw	RAW File Format	
Rax	RAX File Format	

Scn	Caligari scenes	binary and ASCII, with animation
Spx	SPX file format	
Stl	Stereo Litho	
Wrl	Virtual Reality Markup Language VRML 1.0, VRML 2.0	ASCII only
X	DirectX	binary and ASCII
Xsi	SOFTIMAGE XSI	with animation support

ตารางที่ ก-1 แสดงรูปแบบของไฟล์ที่นำเข้าได้แบบ 3 มิติของโปรแกรม 3D Exploration

2.1.2 รูปแบบการนำเข้าไฟล์ 2 มิติ (2D Import)

นามสกุลของไฟล์	ชนิดของไฟล์	Note
Bmp	Windows Bitmap	Windows and OS/2. 1/8/16/24/32 bpp. RLE and uncompressed
Cel	Autodesk CEL file	
fli,flc	Autodesk FLI/FLC file	first frame only
Gif	Graphics Interchange Format	
Iff	EA/Amiga Interchange File Format	1-24 bpp
Jpg	JPEG JFIF	
Pcx	ZSoft Publisher's Paintbrush	
Pic	SOFTIMAGE pic	
Png	Portable Network Graphics	all sub-types supported
Ppm	PPM Image	
Psd	Photoshop document	
Rgb	SGI Image Format	8-32 bpp; no colormaps; RLE or uncompressed
Tga	Targa TGA	all sub-types supported
Tiff	Tag Image File Format	
Tim	Playstation TIM picture	

ตารางที่ ก-2 แสดงรูปแบบของไฟล์ที่นำเข้าได้แบบ 2 มิติของโปรแกรม 3D Exploration

2.2 การส่งออกไฟล์ (Export File)

หน้าที่ของโปรแกรมส่วนนี้ คือการจัดเก็บไฟล์ให้อยู่ในรูปแบบอื่นๆได้ (และตัวโปรแกรม ยังมีความสามารถในการปรับแต่งโมเดลบางส่วนอีกด้วย) โดยที่รูปแบบของ ไฟล์ ที่สามารถจะจัดเก็บได้นั้น แน่นอนว่า ต้องมี ไฟล์ .x รวมอยู่ด้วย นอกจากนั้น ยังสามารถแปลงเป็นรูปแบบอื่นๆได้ซึ่งมีดังนี้

2.2.1 รูปแบบการนำส่งออกไฟล์ 3 มิติ (3D Export)

นามสกุลของไฟล์	ชนิดของไฟล์	Note
3ds	3D Studio binary	with animation export
Asc	3D Studio ASCII	
Cob	Caligari objects	binary and ASCII, with animation export
Cpp	Open GL cpp code	
Dxf	AutoCAD DXF	
Lwo	Lightwave 6.0 objects / Lightwave 5.6 +.uv files	
Obj	Alias WaveFront objects	
Off	Object file format	
Peo,geo	Homeworld geometry	with LiF/BMP textures
Raw	RAW File Format	
Stl	StereoLitho	binary and ASCII
Wrl	VRML 2.0	
X	Direct X	with animation export
Xsi	SOFTIMAGE XSI	with animation export

ตารางที่ ก-3 แสดงรูปแบบของไฟล์ที่ส่งออกได้แบบ 3 มิติของโปรแกรม 3D Exploration

2.2.2 รูปแบบการนำส่งออกไฟล์ 3 มิติ (3D Export)

นามสกุลของไฟล์	ชนิดของไฟล์	Note
Avi	AVI file	
Bmp	Windows Bitmap	
Jpg	JPEG JFIF	
Ppm	PPM Image	
Tga	Targa TGA	

ตารางที่ ก-4 แสดงรูปแบบของไฟล์ที่ส่งออกได้แบบ 2 มิติของโปรแกรม 3D Exploration

บรรณานุกรม

[1] Jim Adams (2002): "Programming Role Playing Games with DirectX", Premire Press, 2002.

[2] Mason McCuskey (2002): "Special Effect Game Programming with DirectX", Premire Press, 2002.

[3] Peter J.Kovach (2000): "Inside Direct3D", Microsoft Press, 2000.

[4] พีรภัทร์ สว่างเพียร (2002): "เทคนิคการเขียนโปรแกรมและเกมด้วย Visual C++", บริษัทซีเอ็ด-ยูเคชั่นจำกัด, 2002

[5] อนุศาสน์ สุวรรณพรหม,พีรพล อริยรัตนา (2001): "3ds max 4 Over all", บริษัทซีเอ็ดยูเคชั่น จำกัด, 2001

[6] พูนศักดิ์ ธนพันธ์พาณิช (2002): "เทคนิคการสร้าง character modeling บน 3ds max 4.2", สำนักพิมพ์เอส.พี.ซี.บุ๊คส์, 2002

[7] ปิยะบุตร สิทธิคารา (2001): "discreet 3ds max 4", สำนักพิมพ์อินโฟเพรส, 2001