บทที่ 4

ทฤษฎีและแนวคิดทางด้านกราฟิกในไมโครซอฟต์วินโดวส์

ชุดคำสั่งกราฟิกในระบบจีดีไอ

ภายในระบบไมโครซอฟต์วินโดวส์ได้มีการบรรจุชุดคำสั่งทางด้านกราฟิก (Graphic Library) ซึ่งสามารถเรียกใช้ได้ขณะที่โปรแกรมภายใต้ไมโครซอฟต์วินโดวส์กำลังทำงานอยู่ โดยชุดคำสั่งเหล่านี้ถูกเรียกโดยรวมว่า จีดีไอ (Graphic Device Interface: GDI) ซึ่งบรรจุอยู่ที่แฟ้ม GDI.EXE ในรูปแบบของ Dynamic-Link Librery (DLL)

จีดีไอนั้นประกอบด้วยชุดคำสั่งที่มีประสิทธิภาพและอรรถประโยชน์ ซึ่งถูกสร้างขึ้นเป็นพิเศษเพื่อใช้ในการทำงานภายใต้ระบบวินโดวส์ นอกเหนือจากความสามารถที่จะสร้างภาพกราฟิกตามมาตรฐาน จีดีไอยังอำนวยประโยชน์ในการทำงานแบบ Dethering, Bitblt และ Clipping function ได้อย่างดีเป็นพิเศษ ซึ่งระบบการทำงานทั้งอย่างนี้ เหมาะสมเป็นอย่างยิ่ง โดยเฉพาะงานกราฟิกที่ซับซ้อน รวมไปถึงการสร้างภาพ 3 มิติ การให้แสงเงา การสร้างภาพเคลื่อนไหวที่มีประสิทธิภาพ และงานมัลติมีเดีย

จีดีไอเป็นกลไกการทำงานทางด้านกราฟิกของวินโดวส์ ซึ่งสามารถเรียกขึ้นมาใช้งานได้จากทุกโปรแกรมที่ทำงานภายใต้วินโดวส์ ภาพที่สร้างขึ้นโดยจีดีไอ สามารถแสดงผลได้ทั้งที่จอภาพ หน่วยความจำ เครื่องพิมพ์ แฟ้มข้อมูล และอุปกรณ์อื่นๆ สืบเนื่องจากความยืดหยุ่นและหลายหลายในการทำงาน ในการใช้งานจีดีไอแต่ละครั้งต้องกำหนดรายละเอียดเกี่ยวกับอุปกรณ์ (Device Context) ที่ใช้ในการแสดงผลลัพธ์ให้ชัดเจน

อุปกรณ์ดังกล่าวถูกแบ่งออกเป็น 2 ประเภทหลักคือ อุปกรณ์ที่ใช้ในการนำข้อมูลเข้า (Input Device) และอุปกรณ์ที่ใช้แสดงผล (Output Device) โดยอุปกรณ์ที่ใช้ในการนำข้อมูลเข้าได้แก่ แผงแป้นอักขระ เมาส์ และนาฬิกา ในขณะที่จอภาพ เครื่องพิมพ์ เครื่องวาด หน่วยขับจานบันทึกข้อมูล และโมเด็มเป็นอุปกรณ์ในการแสดงผลลัพธ์ นอกจากนี้ยังอาจรวมอุปกรณ์ในหน่วยความจำอย่างเป็นแผนที่บิต (Bitmap) และเมตาไฟล์ (Metafile) เป็นอุปกรณ์ในการแสดงผลลัพธ์อีกด้วย

ก่อนการเรียกใช้งานกราฟิกทุกครั้งจะต้องกำหนดรายละเอียดเกี่ยวกับอุปกรณ์ในการแสดงผลลัพธ์ เพื่อที่จะบอกจีดีไอ ทราบว่า ควรจะบันทึกผลงานกราฟิกลงไปที่ไหนและอย่างไร รายละเอียดเกี่ยวกับอุปกรณ์ในการแสดงผลนั้นจะประกอบไปด้วยชนิดของอุปกรณ์แสดงผลและชนิดของไดร์เวอร์ (Driver) ของอุปกรณ์นั้นๆ ไดร์เวอร์คือชุดคำสั่งที่สามารถประมวลผลได้ที่ถูกรวบรวมไว้ด้วยกัน ซึ่งจะสามารถควบคุมการทำงานของฮาร์ดแวร์ที่ต่อเชื่อมอยู่ได้ อุปกรณ์ต่างๆ อย่างเช่น เครื่องพิมพ์ แผงวงจรกราฟิก (Graphic adapters cad) เครื่องวาด (Plotter) หรืออุปกรณ์ในการป้อนข้อมูลอย่างเมาส์ และเครื่องกราดตรวจ (Scaner) ล้วนแล้วแต่มีตัวขับเคลื่อนอุปกรณ์ด้วยกันทั้งสิ้น

ในกรณีที่ใช้งานระบบวินโดวส์บนเครื่องคอมพิวเตอร์ระบบวีจีเอ (Video Graphic Arrey: VGA) รหัสที่จะระบุลงไปในหน่วยความจำแสดงภาพ (Display Memory) บนแผงวงจรกราฟิก จะปรากฏอยู่ในไดร์เวอร์ที่ชื่อว่า VGA.DRV ซึ่งปกติจะอยู่ในไดเร็กตอรี C:\WINDOWS\SYSTEM ภายในจานบันทึกแบบแข็ง เมื่อเริ่มการใช้งานระบบวินโดวส์ จะอ่านไดร์เวอร์นี้มาเก็บไว้ที่หน่วยความจำ ดังนั้นเมื่อใดก็ตามที่จีดีไอถูกเรียกขึ้นมาใช้ในการสร้างงานกราฟิกบนจอภาพ จีดีไอก็จะเรียกไดร์เวอร์ VGA.DRV ขึ้นมาควบคุมการทำงานบนแผงวงจรกราฟิกวีจีเอ

รายละเอียดเกี่ยวกับอุปกรณ์แสดงผลลัพธ์นั้นสามารถระบุอุปกรณ์แสดงผลลัพธ์ประเภทต่างๆ ได้หลายประเภท รายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ (Dispiay Context) ก็เป็นหนึ่งในนั้น รายละเอียดเกี่ยวกับการแสดงผลทางจอภาพจะหมายถึงระบบวินโดวส์บนจอภาพ ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ จะอธิบายถึงการบันทึกผลงานกราฟิกลงบนพื้นที่ใช้งาน (Client Area) ของแต่ละวินโดวส์บนจอภาพ พื้นที่ใช้งาน ได้แก่ พื้นที่ภายในของวินโดวส์ซึ่งไม่รวมถึงขอบของวินโดวส์ แผงรายการเลือก (Menu bar) และแผงแสดงชื่อวินโดวส์ (Caption Bar) อาจกล่าวได้ว่าพื้นที่ใช้งานก็คือพื้นที่ที่จัดไว้สำหรับการวาดภาพ

ชุดคำสั่งต่างๆ ที่มีอยู่ในระบบของจีดีไอภายใต้สภาพแวดล้อมของไมโครซอฟต์วินโดวส์นั้นมีจำนวนมาก ตัวอย่างของชุดคำสั่งจีดีไอได้แสดงไว้ในภาคผนวก ก

<u>รายละเอียดการแสดงผลทางจอภาพ</u>

เมื่อรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพได้ถูกกำหนดขึ้นสำหรับแต่ละวินโดวส์ โดยเมื่อเริ่มทำงาน วินโดวส์จะเลื่อนจุดกำเนิดภาพกราฟิก (Graphic Origin) ไปไว้ยังมุมบนซ้ายของพื้นที่ใช้งานและกำหนดกรอบสี่เหลี่ยมขึ้นใหม่เพื่อให้เหมาะสมกับพื้นที่ใช้งานนั้น โดยวินโดวส์จะทำการกำหนดคุณสมบัติทั้งหมดเกี่ยวกับรายละเอียดการแสดงผลบนจอภาพโดยอัตโนมัติ

คุณสมบัติพื้นฐานเกี่ยวกับการแสดงผลทางจอภาพที่วินโดวส์กำหนดให้โดยสรุป ได้แสดงอยู่ในตาราง 4.1 แม้ว่าโปรแกรมบนวินโดวส์ต่างๆ จะทำการเปลี่ยนแปลงคุณสมบัติเหล่านี้ แต่รายละเอียดเกี่ยวกับการแสดงผลทางจอภาพส่วนใหญ่มักจะถูกใช้งานภายใต้สภาพนี้

ตารางที่ 4.1 แสดงรายละเอียดคุณสมบัติพื้นฐานเกี่ยวกับการแสดงผลทางจอภาพ

รายละเอียดการแสดงผล	คุณสมบัติที่ได้รับการกำหนด
สีฉากหลัง (Background color)	สีขาว
ภาวะของฉากหลัง (Background mode)	ทึบแสง (Opaque)
สีแปรง (Brush color)	สีขาว
สีปากกา (Pen color)	สีดำ
สีตัวอักษร (Text color)	สีดำ
รูปแบบตัวอักษร (Font)	System font, เปลี่ยนขนาดได้ (proportional)
ขอบเขตการตัดภาพ (Clipping region)	พื้นที่ใช้งาน (Client area)
จุดกำเนิดภาพของอุปกรณ์ (DC origin)	จุดพิกัด (0,0) ทางด้านบนซ้ายของพื้นที่ใช้งาน
ตำแหน่งปากกา (Pen position)	จุดพิกัด (0,0)
ตำแหน่งแปรง (Brush position)	จุดพิกัด (0,0)
รูปแบบลายเส้นของปากกา (Pen style)	Solid line
รูปแบบการระบายสีของแปรง (Brush style)	Solid brush

สีฉากหลังที่กำหนดไว้จะเป็นสีขาว ซึ่งหมายความว่า จีดีไอจะสมมติว่าสีฉากหลังของพื้นที่ใช้งานจะเป็นสีขาวด้วย สีแปรงที่กำหนดไว้ก็จะเป็นสีขาวซึ่งหมายความว่าสีภายในภาพที่จีดีไอสร้างขึ้นว่าจะเป็นสีเดียวกับสีฉากหลัง สีปากกาที่กำหนดไว้จะเป็นสีดำดังนั้น เส้นตรง สี่เหลี่ยม เส้นโค้ง และรูปหลายเหลี่ยมใดๆ จะถูกวาดขึ้นด้วยสีดำบนพื้นสีขาว สีตัวอักษรที่กำหนดไว้จะเป็นสีดำ จีดีไอจึงจะเขียนตัวอักษรสีดำบนพื้นสีขาว ภาวะการแสดงสีของฉากหลังจะถูกกำหนดเป็นทึบแสง ดังนั้นส่วนที่เป็นพื้นของแต่ละตัวอักษรจะถูกแสดงเป็นสีขาวขึ้นทับภาพอื่นๆ ที่ปรากฏอยู่ก่อนบนจอภาพ

ลายเส้นของปากกาเริ่มต้นจะเป็นแบบเส้นเต็ม แต่จีดีไอก็สามารถสร้างเส้นประประเภทต่างๆ ขึ้นด้วยสีดำแต่ในขณะเดียวกันช่องว่างระหว่างเส้นก็จะปรากฏขึ้นทับภาพกราฟิกที่ปรากฏอยู่เดิม ขอบเขตการตัดภาพที่ได้กำหนดไว้ก็คือพื้นที่ใช้งานของวินโดวส์ ดังนั้นการวาดภาพต่างๆ ที่ออกนอกพิกัด

บริเวณของพื้นที่ใช้งานจะไม่ปรากฏให้เห็น เส้น รูปเหลี่ยม แผนที่บิต และตัวอักษรทั้งหลายจะถูกตัดที่ขอบของพื้นที่ใช้งาน

เนื่องจากจุดกำเนิดของระบบพิกัดกราฟิก (0,0) อยู่ที่มุมบนซ้ายของพื้นที่ใช้งาน ดังนั้นตำแหน่งเริ่มต้นของปากกาและแปรง จึงถูกตั้งไว้ที่พิกัด 0,0 ด้วย ในการกำหนดรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพนั้น จะต้องเรียกใช้คำสั่ง GetDC() ของจีดีไอขึ้นมา เพื่อส่งผ่านการจัดการระบบวินโดวส์ที่ต้องการจะสร้างภาพกราฟิกไปยังจีดีไอ GetDC() จะคืนการจัดการไปสู่รายละเอียดเกี่ยวกับการแสดงผลทางจอภาพที่ถูกกำหนดขึ้น ซึ่งรายละเอียดเหล่านี้จะใช้ในการจัดการงานด้านกราฟิกภายใต้จีดีไอต่อไป

การลบรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ จะต้องดำเนินการหลังจากเลิกใช้งานรายละเอียดเหล่านั้นแล้ว เนื่องจากระบบวินโดวส์รองรับรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพได้เพียง 5 แบบเท่านั้น จึงควรที่จะลบรายละเอียดการแสดงผลทางจอภาพออกหลังจากเลิกการใช้งาน หากไม่ทำเช่นนี้โปรแกรมอื่นๆ อาจจะไม่สามารถกำหนดรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพและจะไม่สามารถสร้างภาพกราฟิกบนจอภาพได้

การลบรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพนั้นทำได้โดยการเรียกใช้คำสั่ง ReleaseDC() โปรแกรมที่ดีนั้นควรที่จะกำหนดรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพเมื่อเริ่มใช้สร้างผลงานทางด้านกราฟิกและจะลบรายละเอียดดังกล่าว เมื่อสิ้นสุดการใช้งาน โปรแกรมสร้างภาพเคลื่อนไหว (animation) บางโปรแกรมจะกำหนดรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพเมื่อโปรแกรมเริ่มการทำงาน และจะลบรายเอียดดังกล่าวออกเมื่อโปรแกรมหยุดการทำงาน การทำเช่นนี้จะเป็นการหลีกเลี่ยงการที่จะต้องกำหนดและลบรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพสำหรับแต่ละภาพเคลื่อนไหว (animation frame) นั้นๆ ได้

รายละเอียดเกี่ยวกับการแสดงผลทางจอภาพสามารถบันทึกและเรียกกลับมาใช้งานได้ด้วยคำสั่ง SaveDC() ซึ่งจะเป็นการสะดวกในกรณีที่คุณสมบัติหลายลักษณะที่ถูกกำหนดไว้เดิมนั้น อย่างเช่น สีฉากหลัง หรือขอบเขตการตัดภาพได้ถูกเปลี่ยนแปลงไป คำสั่ง SaveDC() จะบันทึกรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพไว้ในคลังรายละเอียด (context stock) สำหรับการเรียกรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพที่ได้ถูกบันทึกไว้ในคลังรายละเอียดกลับมาใช้งานนั้นให้ใช้คำสั่ง RestoreDC()

คำสั่ง SaveDC() และ RestoreDC() จะเป็นประโยชน์อย่างยิ่งโดยเฉพาะในกรณีที่โปรแกรมทางด้านกราฟิกนั้นต้องใช้วินโดวส์มากกว่าหนึ่งวินโดวส์ในการแสดงภาพ หรือสำหรับโปรแกรมที่ต้องใช้ฉากในการแสดงภาพเป็นพิเศษ การเรียกรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ ขึ้นมาหรือการเก็บ

คืนกลับไปในคลัง นั้นทำได้รวดเร็วและง่ายกว่าการกำหนดคุณลักษณะของรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ ในแต่ละครั้งมากนัก

ในการสร้างโปรแกรมกราฟิกภายใต้ระบบวินโดวส์นั้น การกำหนดรายละเอียดเกี่ยวกับอุปกรณ์ที่เข้ากันได้ (compatible device context) ก็คือรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพที่ถูกจำลองขึ้น ซึ่งจะเป็นส่วนหนึ่งของหน่วยความจำที่กันไว้เสมือนเป็นแผนที่บิต โดยแผนที่บิตก็คือ พื้นที่สี่เหลี่ยมของบิต (metrix of bit) ที่สามารถใช้วาดหรือทำงานทางด้านกราฟิกได้ แผนที่บิตจะคล้ายกับโปรแกรมภายใต้วินโดวส์ที่แสดงภาพทางจอเป็นอย่างยิ่ง ซึ่งจอภาพก็คือแมทริกซ์ของจุดภาพ (metrix of pixel) ที่แสดงถึงพื้นที่สี่เหลี่ยมของบิตในหน่วยความจำของแผงวงจรกราฟิก

ในการที่จะคัดลอกแผนที่บิตจากวินโดวส์หนึ่ง ไปไว้ในอีกวินโดวส์หนึ่งของโปรแกรมภายใต้วินโดวส์ทางจอภาพ จะต้องมีคุณลักษณะของรายละเอียดเกี่ยวกับการแสดงผลบนจอภาพหลายประการที่เหมือนกับวินโดวส์นั้น โดยทำการกำหนดไว้ในรายละเอียดของอุปกรณ์ในหน่วยความจำที่ช่วยให้มีคุณลักษณะของรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพเข้ากันได้กับจอภาพนั้น

<u>อุปกรณ์วาดภาพ</u>

อุปกรณ์ในการสร้างภาพของจีดีไอ นั้นประกอบด้วยปากกา แปรง และฟอนต์ (font) ปากกาใช้สำหรับสร้างเส้นและรูปทรง แปรงจะใช้ในการระบายสี และฟอนต์จะใช้ในการสร้างตัวอักษรชนิดต่างๆ

การใช้งานของอุปกรณ์ในการสร้างภาพนั้นเน้นการทำงานเป็นพิเศษในเรื่องของการสร้าง การกำหนดและการยกเลิก ในขณะที่การสร้างภาพ โดยการกำหนดคุณสมบัติของการวาด (Drawing attributes) จะมุ่งเน้นถึงการประยุกต์ใช้อุปกรณ์สร้างภาพในการใช้งาน

การเข้าใจถึงความแตกต่างระหว่างอุปกรณ์สร้างภาพและการสร้างภาพนั้นเป็นสิ่งจำเป็น อุปกรณ์สร้างภาพอันได้แก่ ปากกา แปรง และ ฟอนต์ จะใช้ในการสร้างผลงานภาพ ซึ่งได้แก่ เส้น รูปหลายเหลี่ยม รูปสี่เหลี่ยม เส้นโค้ง ตัวอักษร แผนที่บิต เมตาไฟล์ และอื่นๆ

1. การใช้อุปกรณ์สร้างภาพ

การใช้งานอุปกรณ์สร้างภาพนั้นคือ การสร้าง กำหนดเลือก และยกเลิก อุปกรณ์สร้างภาพต่างๆ เช่น ก่อนที่จะใช้ปากกาสีแดงนั้นจำเป็นต้องสร้าง และกำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพก่อน เมื่อใช้ปากการสีแดงเสร็จแล้ว ก็จะต้องถอนออกจากรายละเอียดเกี่ยวกับการ

แสดงผลทางจอภาพโดยการเรียกปากกาที่ได้กำหนดไว้เดิมขึ้นมาไว้ในรายละเอียดดังกล่าว หลังจากนั้นจึงยกเลิกปากกาสีแดงนั้นเพื่อยกเลิกข้อมูลในส่วนที่มีอยู่ในหน่วยความจำ

1.1 การสร้างอุปกรณ์วาดภาพ เป็นเรื่องของการสร้างปากกา และแปรงประเภทต่างๆ ในการสร้างปากกาลักษณะใหม่ให้ใช้คำสั่ง CreatePen() ในการสร้างแปรงระบายสีแบบสีเดี่ยว (solid brush) ใช้คำสั่ง CreateSolidBrush() ส่วนในการสร้างแปรงระบายสีแบบลวดลาย (pattern brush) ให้ใช้คำสั่ง CreateHatchBrush() หรือ CreatePatternBrush() อุปกรณ์สร้างภาพที่มีบรรจุอยู่ในวินโดวส์ โดยเก็บอยู่ในคลังวัตถุ (stock object) ของวินโดวส์ เช่น แปรงระบายสีแบบโปร่งแสง (transparent brush) หรือ ปากกาแบบเส้นประ (dashed pen) ถูกใช้เป็นประโยชน์ได้โดยการใช้คำสั่ง GetStockObject()

1.2 การกำหนดเลือกอุปกรณ์สร้างภาพ เพื่อที่จะใช้งานอุปกรณ์สร้างภาพที่ถูกสร้างขึ้นใหม่จะต้องกำหนดไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพด้วยคำสั่ง SelectObject()

1.3 การยกเลิกอุปกรณ์สร้างภาพ หลังจากใช้อุปกรณ์ดังกล่าวเสร็จแล้ว ควรที่จะยกเลิกการใช้งานอุปกรณ์นั้น เพื่อที่จะทำให้หน่วยความจำว่างด้วยการกำจัดข้อมูลเกี่ยวกับอุปกรณ์นั้น ซึ่งโปรแกรมที่ดีจะจัดการหน่วยความจำด้วยความระมัดระวังเนื่องจากหน่วยความจำเป็นอุปกรณ์ที่ต้องใช้ร่วมกับส่วนอื่นๆ นอกจากนี้การยกเลิกการใช้งานอุปกรณ์นั้น ยังช่วยให้สามารถลบรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพได้อย่างสมบูรณ์ ซึ่งจะไม่สามารถเกิดขึ้นได้ หากยังไม่ถอนอุปกรณ์ที่ได้เลือกไว้ออก ซึ่งโปรแกรมที่ดีก็มักจะลบรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพออกทันทีหลังจากเลิกใช้งานส่วนนั้น

การถอนอุปกรณ์สร้างภาพออกนั้นให้ใช้คำสั่ง SelectObject() เพื่อเลือกอุปกรณ์สร้างภาพที่ได้กำหนดไว้เดิมกลับขึ้นมาในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ และใช้คำสั่ง DeleteObject() เพื่อถอนอุปกรณ์ที่เลิกใช้งานออก

2. การสร้างภาพ

การใช้งานทางด้านลักษณะการสร้างภาพนั้นเป็นการประยุกต์ใช้อุปกรณ์สร้างภาพในการทำงาน การใช้งานด้านนี้มักจะเป็นการเปลี่ยนคุณสมบัติของฉากหลัง ลักษณะของการวาด สีตัวอักษร และคุณสมบัติของแผนที่บิต

2.1 คุณสมบัติของฉากหลัง ในการกำหนดสีของฉากหลังใช้คำสั่ง SetBkColor() ส่วนการกำหนดภาวะของฉากหลังใช้คำสั่ง SetBkMode() ถ้าฉากหลังเป็นภาวะทึบแสง จีดีไอจะกำหนดช่องว่างระหว่างเส้นประและพื้นที่ภายในรูปทรงให้เป็นสีเดียวกับฉากหลัง โดยการเขียนทับลงไปบนภาพที่มีอยู่

เดิม กรณีที่ลักษณะของฉากหลังเป็นแบบโปร่งแสง จีดีไอจะปล่อยช่องว่างไว้เมื่อสร้างรูปแบบของเส้นและลวดลาย ภาวะของฉากหลังยังมีผลถึงการเขียนตัวอักษร ส่วนที่เป็นช่องว่างในบริเวณที่ใช้เขียนตัวอักษรนั้นจะเป็นทึบแสง หรือโปร่งแสงขึ้นอยู่กับภาวะของฉากหลังขณะที่เขียน

2.2 คุณสมบัติของการวาด คุณสมบัติของการวาดที่ได้กำหนดไว้เดิมคือการเขียนทับ ซึ่งก็คือภาพที่เขียนขึ้นจะถูกเขียนทับภาพที่มีอยู่เดิมในบริเวณนั้น ในการเปลี่ยน boolean logic ซึ่งจีดีไอใช้สำหรับปากกาและการระบายสีวัตถุนั้นใช้คำสั่ง SetROP2() ในการใช้ boolean operators อย่างเช่น OR, XOR, AND, NOT ในลักษณะการวาดภาพ ระบบวินโดวส์จะเรียกใช้ boolean operator ใน raster operation code โดยกำหนดชื่อไว้เช่น R2_COPYPEN, R2_MERGEPEN, R2_XORPEN และ R2_NOT raster operation code เช่น exclusive_or (XOR) ซึ่งเป็นประโยชน์สำหรับการสร้างตัวชี้ของเมาส์ (mouse pointer)

2.3 คุณสมบัติการยืดขยายแผนที่บิต จีดีไอสามารถยืดขยายหรือหดแผนที่บิต ที่ได้คัดลอกมาด้วย ฟังก์ชัน bitblt ซึ่งจะสะดวกอย่างยิ่งถ้าเป็นการปะภาพจากคลิปบอร์ด ถ้าภาพนั้นใหญ่กว่าวินโดวส์หลักของโปรแกรมอยู่เล็กน้อย ภาวะการยืดขยาย (stretch mode) จะสามารถเรียกขึ้นมาใช้งานด้วยคำสั่ง SetStretchBltMode() โดยภาวะการยืดขยายนี้จะทำงานโดยจะลบการกราดวาดเส้น (scan line) หรือจุดภาพ เพื่อที่จะลดขนาดของแผนที่บิตต้นแบบ (source bitmap) ให้เข้ากับพื้นที่ที่มี เส้นที่ถูกกำจัดจะสามารถลบได้โดยตรงหรือจะเลือกใช้งานจาก boolean operator OR และ AND OR operator จะคง จุดภาพสีขาวไว้ขณะที่ลบจุดภาพสีดำ และ AND operator จะเก็บจุดภาพสีดำไว้ขณะที่ลบจุดภาพสีขาว

คำสั่ง SetStretchBltMode() จะมีผลกระทบต่อการทำงานคัดลอกของคำสั่ง StretchBlt() ในขณะที่ไม่ส่งผลกระทบกับคำสั่ง BitBlt() ที่จะคัดลอกแผนที่บิตมาในขนาดเดียวกับการขยายหรือย่อ ดังนั้นจึงควรใช้คำสั่ง BitBlt() ในการคัดลอกแผนที่ขนาดเดียวกัน และใช้คำสั่ง StretchBlt() ในกรณีที่แผนที่บิตต้องถูกย่อหรือขยายให้เหมาะสม

2.4 คุณสมบัติของปากกา การสร้างปากกาในการเขียนภาพนั้นจะใช้คำสั่ง CreatePen() โดยสามารถกำหนดรูปแบบ ความกว้างของเส้น และสีได้ตามต้องการ รูปแบบมีได้ 6 แบบตามที่ระบบวินโดวส์มีให้ ซึ่งรวมถึงเส้นทึบ เส้นประ และโปร่งใส ความกว้างของเส้นที่กำหนดไว้เดิมคือ 1 หน่วย โดยที่สามารถกำหนดให้เส้นปากกาหนาขึ้นกว่านี้ได้ตามต้องการ คำสั่ง CreatePen() สามารถเรียกขึ้นใช้ได้ไม่ว่าจะเป็นสีแบบใด แต่ระบบวินโดวส์จะใช้สีทึบเท่านั้นในการวาดโดยปากกา ซึ่งหมายความว่า ปากกาจะมีสีได้เพียง 16 สีเท่านั้น หากใช้งานระบบวินโดวส์กับการแสดงผลแบบวีจีเอ ระบบวินโดวส์สามารถ

สร้างสีอื่นๆ ได้อีกโดยการ dithering แต่จะไม่ใช้กับสีของปากกา ลักษณะของปากกาที่กำหนดไว้เดิมจะมีสีดำ เส้นทึบ และหนา 1 จุดภาพ

2.5 คุณสมบัติของแปรง การสร้างแปรงต้องใช้คำสั่ง CreateSolidBrush() ซึ่งจะสามารถเลือกใช้สีใดก็ได้ โดยสามารถยืนยันผลลัพธ์ได้ตามนั้น ขณะใช้งานระบบวินโดวส์ในการแสดงผลแบบวีจีเอ จีดีไอจะใช้ dithering ในการสร้างสีที่ต้องการ หากสีนั้นไม่ใช่สี 16 สีที่มีให้ dithering หมายถึงการผสมผสานจุดภาพที่มีสีแตกต่างกัน ซึ่งผลที่ได้จะเปลี่ยนเป็นสีใหม่ หากใช้งานระบบวินโดวส์ในการแสดงผลแบบ Super VGA หรือ 8514/A หรือ XGA จีดีไอจะสามารถใช้แถบสีเพิ่มมากขึ้นตามระบบของเครื่องในการสร้างสีที่แท้จริงตามต้องการ ลักษณะของแปรงที่กำหนดไว้เดิมคือสีขาวทึบ

2.6 รูปแบบตัวอักษร คำสั่ง TextOut() จะใช้ในการสร้างตัวอักษรตามรูปแบบที่กำหนดไว้ในขณะนั้น รูปแบบตัวอักษรที่กำหนดไว้เดิมโดยวินโดวส์จะเป็น Sans-serif รูปแบบช่องไฟที่กำหนดตามสัดส่วน เรียกว่า System รูปแบบตัวอักษรอื่นๆ ก็มีให้มากมาย รวมถึง Serif, San-serif, fixed-spacing และ propotional spaced typeface การเรียกใช้รูปแบบตัวอักษรใหม่ให้ใช้คำสั่ง GetStockObject() กำหนดเลือกรูปแบบตัวอักษรไว้ในรายละเอียดเกี่ยวกับการการแสดงผลทางจอภาพด้วยคำสั่ง SelectObject()

เพื่อที่จะดัดแปลงรูปแบบการแสดงตัวอักษรของจีดีไอ เช่น เจาะจงขนาด Styles Weight และอื่นๆ สามารถใช้คำสั่ง CreateFont() สีตัวอักษรที่กำหนดไว้เดิมนั้นคือสีดำ ในการเปลี่ยนสีตัวอักษรสามารถเรียกใช้คำสั่ง SetTextColor() ตัวอักษรทมี่ปรากฏตามมาจะมีสีใหม่ตามที่เปลี่ยนแม้ว่าจะเปลี่ยนรูปแบบตัวอักษรไปแล้วก็ตาม

2.7 คุณสมบัติของสี เมื่อเลือกใช้ปากกาหรือแปรงจะต้องระบุสีที่จะใช้ วิธีที่ง่ายที่สุดคือเรียก RGB() มาใช้เสมือนเป็นตัวส่งค่าของสีไปให้ RGB gun คำสั่ง RGB() จะให้ค่า COLORREF ที่เหมาะสมสำหรับการใช้งานโดยจีดีไอ

คำสั่ง RGB() เป็นการกำหนดค่าของสีแดง เขียว และน้ำเงินของจอภาพระบบ CRT แต่ละค่าสีจะใช้โครงสร้างข้อมูลแบบ BYTE และสามารถแปรค่า 0 ถึง 255 ค่า 0 จะเปิด RGB gun ในขณะที่ค่า 255 จะเปิด RGB gun ที่ความเข้มสูงสุด RGB(0,0,0) จะให้สีดำ ส่วน RGB(255,255,255) จะให้สีขาว RGB(0,0,255) จะสร้างสีน้ำเงินและ RGB(127,127,127) จะสร้างสีเทา

ถ้าสีที่ต้องการมีอยู่ในแผงวงจรกราฟิก จีดีไอก็จะแสดงสีที่แท้จริงออกมา แต่ถ้าหากไม่มี จีดีไอจะจำลองสีที่ต้องการนั้นจากการ dithering โดยใช้คำสั่ง RGB() ในการเลือกสีใดสีหนึ่ง จาก color palette ที่มีอยู่

Color palette เป็นกลุ่มของสีที่โปรแกรมสามารถแยกแยะและใช้ได้ palette ที่ถูกกำหนดไว้เดิมเป็น system palette หรือ hardware palette โดยระบบวินโดวส์ ซึ่งจะถูกกำหนดไว้ตั้งแต่เริ่มใช้งานระบบวินโดวส์ และจะไม่สามารถเปลี่ยนได้ด้วยการเรียกใช้งานของจีดีไอ คำสั่ง SetSystemPaletteUse() จะสามารถเปลี่ยนบางส่วนของ palette เท่านั้น

Logical Palette เป็นชุดของสีที่โปรแกรมกำหนดขึ้นไว้เพื่อใช้เอง ซึ่งใช้จีดีไอในการสร้าง ระบบวินโดวส์จะใช้วิธีที่แตกต่างกันในการสร้าง palette ซึ่งขึ้นกับ graphic adapter ที่ใช้ กรณีที่เป็นวีจีเอ และ palette color ไม่ใช่สี 16 สี ที่มีให้บนวีจีเอแล้วระบบวินโดวส์จะใช้งาน dithering ในการจำลองสี

ในการที่จะแบ่งพื้นที่ในหน่วยความจำไว้สำหรับข้อมูลของ palette ให้ใช้คำสั่ง LocalAlloc() หากต้องการระบุสีใน palette ใช้คำสั่ง CreatePalette() ก่อนที่จะใช้สีใน palette ได้นั้นจะต้องกำหนดเลือก palette นั้นไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพด้วยคำสั่ง SelectPalette() และคำสั่ง RealizePalette() จะใช้ในการเพิ่มสีให้กับ RGB gun โดยตั้งให้ไปค้นจากตารางสีของ super VGA, 8514/A และ XGA

<u>การสร้างรูปกราฟิกพื้นฐาน</u>

จีดีไอของระบบวินโดวส์จะแสดงผลลัพธ์ต่างๆ อันได้แก่ เส้น รูปสี่เหลี่ยม รูปหลายเหลี่ยม วงรี แผนที่บิต ขอบเขตการตัดภาพ เมตาไฟล์ ฟอนต์ และความสามารถในการพิมพ์ ในลักษณะของ cross section ซึ่งผลลัพธ์เหล่านี้บางอย่างจะถูกใช้ในบางโปรแกรมเป็นพิเศษเท่านั้น ในขณะที่ผลลัพธ์อื่นๆ จะถูกใช้โดยทั่วไปในโปรแกรมกราฟิก

เส้นตรง รูปสี่เหลี่ยม รูปหลายเหลี่ยมและวงรี การทำงานวาดภาพระดับพื้นฐานของจีดีไอ ได้แก่ การวาดเส้นตรง รูปสี่เหลี่ยม รูปหลายเหลี่ยม และวงรี

คำสั่ง MoveTo() ใช้ในการกำหนดตำแหน่งของปากกา แม้ว่าคำสั่งนี้จะไม่ได้เป็นการวาดภาพกราฟิกและเส้นใดๆ ก็ตามที่จะถูกวาดต่อจากนี้จะใช้ตำแหน่งปากกาที่กำหนดไว้แล้วนี้เป็นจุดเริ่มต้น

คำสั่ง LineTo() ไว้ในการวาดเส้นตรงจากตำแหน่งของปากกาที่ได้กำหนดไว้ ไปยังตำแหน่งที่ระบุไว้ขณะที่เรียกใช้คำสั่ง LineTo() ตำแหน่งปัจจุบันของปากกาจะถูกกำหนดขึ้นใหม่ ในจุดที่เรียกใช้คำสั่ง ดังนั้น เส้นต่อไปที่จะถูกลากขึ้นก็จะเริ่มต้นจากจุดสิ้นสุดของเส้นก่อนหน้าเสมอ เส้นตรงนั้นจะถูกลากด้วยสี รูปแบบ และความหนาของปากกา ที่ได้กำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ

คำสั่ง PolyLine() สามารถใช้ในการวาดชุดของเส้นต่อเนื่อง (set of connected line segment) ในการใช้คำสั่งนี้จำเป็นต้องกำหนดชุดของพิกัด (array of coordinates) ที่บอกถึงจุดต่างๆ ที่เป็นมุมของ Polyline คำสั่ง Polyline() จะถูกวาดขึ้นด้วยสี รูปแบบ และความหนาของปากกา ที่ได้ถูกกำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ แต่คำสั่ง PolyLine() นี้จะไม่กำหนดตำแหน่งของปากกาใหม่

คำสั่ง LineDDA() ของจีดีไอจะใช้เพื่อหาจุดพิกัด X,Y ของแต่ละจุดภาพบนเส้นแต่ละเส้น คำสั่ง LineDDA() จะเรียก core function ขึ้นมาทุกครั้งที่จะคำนวนจุด การทำงานนี้มีประโยชน์เป็นอย่างยิ่งสำหรับการทำ shading และ tracing ของ 3d ray การทำ advance hiden surface removal ใน complex 3d scenes และสำหรับการ Scan-Line Algorithm อื่นๆ

ในการวาดรูปสี่เหลี่ยมจะใช้คำสั่ง Rectangle() โดยต้องกำหนดจุด 4 จุดให้ด้วย รูปสี่เหลี่ยมที่ถูกวาดขึ้นนี้จะมีสี รูปแบบ และความหนาของเส้นปากกาที่ได้กำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพในขณะนั้น พื้นที่ภายในรูปสี่เหลี่ยมจะถูกระบายสีด้วยแปรงที่กำหนดไว้ ถ้าใช้คุณลักษณะตามที่กำหนดไว้เดิมของรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ รูปสี่เหลี่ยมจะถูกวาดขึ้นด้วยเส้นสีดำ และพื้นที่ภายในจะเป็นสีขาวเหมือนสีพื้นของระบบวินโดวส์ โดยพื้นที่ภายในสี่เหลี่ยมนี้จะลบภาพกราฟิกที่มีอยู่เดิม ในบริเวณนั้นออกไปโดยสิ้นเชิง คำสั่ง Rectangle() มักจะระบายสีพื้นที่ภายในด้วยลักษณะของแปรงที่กำหนดไว้ในขณะนั้น หากต้องการให้พื้นที่ภายในเป็นเหมือนฉากหลังของภาพกราฟิกแล้ว ควรเลือกแปรงแบบโปร่งแสง

คำสั่ง RoundRect() จะใช้ในการวาดรูปสี่เหลี่ยมมุมโค้งตามจุด 4 จุดที่ได้กำหนดให้รูปสี่เหลี่ยมนี้ก็จะใช้สี รูปแบบ และความหนาของปากกาที่กำหนดเลือกไว้ในขณะนั้น ในรายละเอียดเกี่ยวกับการแสดงผลของจอภาพ พื้นที่ภายในของรูปสี่เหลี่ยมสามารถระบายสีได้ด้วยแปรง ลักษณะของมุมโค้งจะเป็นไปตามค่าความกว้างและความลึกของวงรีที่กำหนด

คำสั่ง Polygon() ใช้ในการสร้างรูปหลายเหลี่ยมจากชุดของจุด ที่กำหนดให้หากจุดสุดท้ายที่กำหนดไม่บรรจบ รูปหลายเหลี่ยมที่จะสร้าง คำสั่งนี้จะบรรจบรูปดังกล่าวโดยการต่อเชื่อมเส้นระหว่างจุดสุดท้ายที่กำหนดกับจุดแรก รูปหลายเหลี่ยมนี้จะมีสี รูปแบบ และความหนาของเส้นตามปากกาที่ได้กำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพในขณะนั้น พื้นที่ภายในรูปหลายเหลี่ยมนี้จะถูกระบายสีด้วยแปรงที่กำหนดไว้ในขณะนั้น โดยใช้ขั้นตอนวิธีในการระบายสี (filling algorithm) ที่มีอยู่ ซึ่งสามารถระบุขั้นตอนวิธีในการระบายสีด้วยคำสั่ง SetPolyFillMode() การระบายสีแบบ winding ควรจะได้ถูกกำหนดขึ้นเพื่อให้แน่ใจได้ว่าพื้นที่ภายในทั้งหมดของรูปหลายเหลี่ยมที่ซับซ้อนจะถูกระบายในรูปหลายเหลี่ยมที่เรียบง่ายสามารถใช้ขั้นตอนวิธีแบบ alternate

หากต้องการสร้างรูปหลายเหลี่ยมมากกว่า 1 รูป สามารถเรียกใช้คำสั่ง PolyPolygon() ในการวาดชุดของรูปหลายเหลี่ยมตามจุดที่กำหนด ในการใช้คำสั่งนี้จะไม่เหมือนคำสั่ง Polygon โดยคำสั่งนี้จะไม่บรรจบรูปหลายเหลี่ยมที่สร้างขึ้นให้

วงกลมและวงรีสามารถวาดขึ้นได้ด้วยคำสั่ง Ellipse() พิกัด X,Y 4 ชุดที่กำหนดให้จะถูกจีดีไอใช้ในการกำหนด boundary box ซึ่งจะเป็นกรอบกำหนดรูปวงกลมหรือวงรีที่ต้องการวาด หาก boundary box เป็นรูปสี่เหลี่ยมจัตุรัส ผลลัพธ์ที่ได้จะเป็นรูปวงกลม และหากเป็นรูปสี่เหลี่ยมผืนผ้าผลลัพธ์ที่ได้จะเป็นรูปวงรี รูปที่วาดขึ้นนี้จะมีสี รูปแบบ และความหนาของเส้นตามปากกาที่ในขณะนั้นได้กำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ พื้นที่ภายในรูปก็จะถูกระบายสีด้วยแปรงที่กำหนดไว้ในขณะนั้น

<u>การกำหนดขอบเขตการสร้างภาพ</u>

จีดีไอของระบบวินโดวส์มีคำสั่งเกี่ยวกับการจัดการขอบเขตการสร้างภาพ (region) ที่มีประสิทธิภาพไว้ด้วยกันสองส่วนคือ ขอบเขตของการระบายสี (fill region) และขอบเขตของการตัดภาพ (clipping region) คำสั่งประเภทขอบเขตของการระบายสีมีไว้เพื่อสร้างพื้นที่ของรูปหลายเหลี่ยมที่สามารถระบายสีพื้นที่ส่วนในด้วยแปรง ส่วนขอบเขตของการตัดภาพนั้นคือกรอบสี่เหลี่ยมที่ใช้ในการตัดภาพกราฟิก

1. ขอบเขตของการระบายสี

ขอบเขตของการระบายสีรูปสี่เหลี่ยมผืนผ้าสามารถถูกสร้างด้วยคำสั่ง CreateRectRgn() คำสั่ง CreateRoundRectRgn() ใช้ในการสร้างขอบเขตของการระบายสีรูปสี่เหลี่ยมผืนผ้ามุมโค้ง และคำสั่ง CreatEllipseRgn() จะใช้ในการสร้างขอบเขตของการระบายสีรูปวงรี

คำสั่ง FillRgn() ใช้สำหรับการระบายสีลงในขอบเขตของการระบายด้วยแปรงที่กำหนด ส่วนคำสั่ง PaintRgn() นั้นจะระบายสีขอบเขตของการระบายด้วยแปรงที่กำหนดเลือกไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพในขณะนั้น

การ invert สีในขอบเขตการระบายนั้นสามารถใช้คสั่ง InvertRgn() ในการทดสอบว่าจุดที่กำหนดไว้นั้นอยู่ภายในขอบเขตการระบายหรือไม่นั้นจะใช้คำสั่ง PtInRegion() การสร้างขอบเขตใหม่จากขอบเขตที่ปรากฏอยู่เดิม 2 ขอบเขตจะใช้คำสั่ง CombineRgn() นอกจากนี้ยังสามารถจัดการกับขอบเขตการระบายได้อีกหลายอย่าง ควรใช้คำสั่งเกี่ยวกับขอบเขตการตัดภาพในจีดีไอในการกำหนดขอบเขตในการตัดภาพที่อยู่นอกเหนือไปจากพื้นที่ใช้งาน

2. ขอบเขตการตัดภาพ

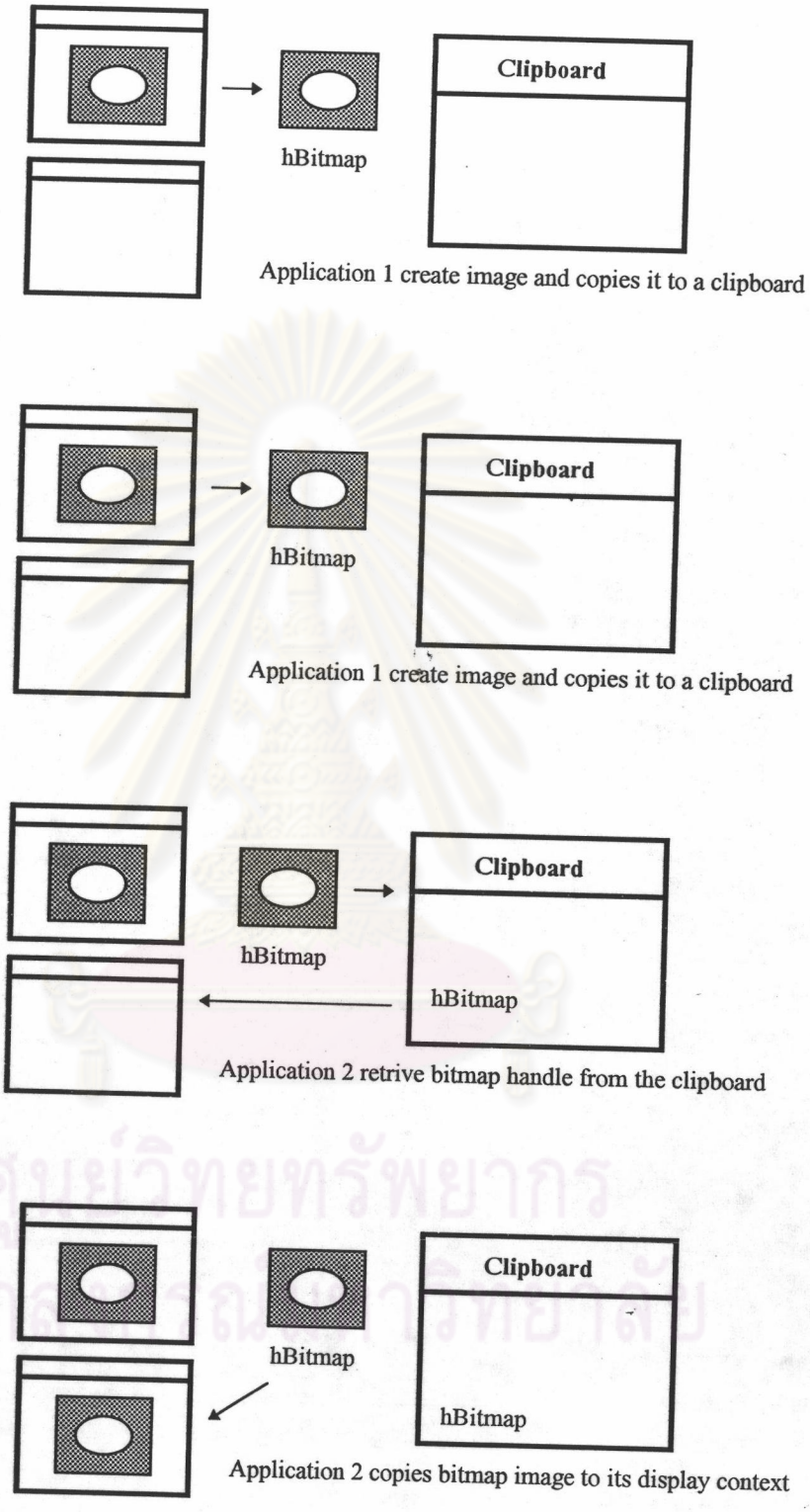
ในการสร้างขอบเขตการตัดภาพนั้น ขั้นตอนแรกต้องใช้คำสั่งอย่างเช่น CreateRectRgn() เพื่อที่จะสร้างขอบเขต ต่อจากนั้นจะใช้คำสั่ง SelectClipRgn() ในการเลือกขอบเขตนั้นให้เป็นขอบเขตการตัดภาพในขณะนั้น คำสั่งใดๆ ก็ตามของจีดีไอที่ถูกเรียกขึ้นมาใช้ในการสร้างภาพกราฟิกต่อจากนี้จะถูกตัดภาพที่ขอบของขอบเขตที่กำหนดไว้

เพื่อเป็นการทดสอบว่าจุดใดจุดหนึ่งนั้นจะอยู่ในขอบเขตการตัดภาพหรือไม่นั้นจะใช้คำสั่ง PtVisible() เพื่อตรวจสอบดูว่ารูปสี่เหลี่ยมใดๆ นั้นจะอยู่ภายในขอบเขตการตัดภาพหรือมีเพียงบางส่วนที่ทับกับขอบเขตการตัดภาพเท่านั้นหรือไม่ สามารถใช้คำสั่ง RectVisible() และเพื่อเคลื่อนย้ายขอบเขตการตัดภาพจะต้องใช้คำสั่ง OffsetClipRgn()

<u>การทำงานของคลิปบอร์ด</u>

คลิปบอร์ดจะเป็น common buffer ที่ปรากฏอยู่ขณะที่ระบบวินโดวส์กำลังทำงานที่จะมีประโยชน์ในการแลกเปลี่ยนข้อมูลระหว่างโปรแกรม ชุดคำสั่งของจีดีไอที่ใช้ในการแลกเปลี่ยนภาพจากโปรแกรมที่ใช้งานกับโปรแกรมอื่นของระบบวินโดวส์

ลักษณะการทำงานของคลิปบอร์ด เมื่อโปรแกรมที่ใช้งานจะร่วมใช้ภาพกับโปรแกรมอื่นด้วยคลิปบอร์ดนั้น โปรแกรมที่ใช้งานจะส่งการจัดการเรื่องนี้ไปให้กับคลิปบอร์ดดังปรากฏในรูป 4.1 การจัดการนั้นมักจะเป็นการจัดการกับแผนที่บิตที่มีภาพนั้นบรรจุอยู่ โปรแกรมที่จะรับภาพนั้นจะปะภาพนั้นลงบนพื้นที่ใช้งาน โดยการดึงภาพนั้นจากที่ถูกบันทึกไว้ในคลิปบอร์ด



รูปที่ 4.1 แสดงการส่งข้อมูลภาพผ่านคลิปบอร์ด

การคัดลอกภาพลงในคลิปบอร์ด ก่อนที่จะคัดลอกภาพใดๆ ลงสู่คลิปบอร์ดนั้นจำเป็นต้องสร้างแผนที่บิตเพื่อรองรับภาพนั้นขึ้นก่อน โดยใช้คำสั่ง CreateCompatibleDC() ในการกำหนดรายละเอียดเกี่ยวกับการแสดงผลบนหน่วยความจำ และใช้คำสั่ง BitBlt() ในการคัดลอกภาพจากจอภาพหลักของโปรแกรมไปยังแผนที่บิต ต่อไปจึงใช้คำสั่ง OpenClipboard() เพื่อเปิดการใช้งานของคลิปบอร์ดของระบบวินโดวส์ที่มีอยู่ ขั้นต่อไปควรที่จะเรียกคำสั่ง EmptyClipboard() เพื่อทำให้คลิปบอร์ดว่างลงเสียก่อน ต่อจากนั้นด้วยการเรียกคำสั่ง SetClipboardData() บิตแม็พที่ต้องการจะถูกส่งต่อไปยังคลิปบอร์ด สุดท้ายจบการทำงานในขั้นตอนนี้ด้วยคำสั่ง CloseClipboard() ควรปิดการทำงานของคลิปบอร์ดทุกครั้งหลังจากบันทึกภาพลงไปแล้ว เพราะโปรแกรมอื่นจะไม่สามารถใช้งานได้หากไม่ทำเช่นนั้น

การปะภาพจากคลิปบอร์ดใช้คำสั่ง GetClipboardData() เพื่อเรียกแผนที่บิตที่ต้องการกลับขึ้นมาจากคลิปบอร์ด ต่อจากนั้นก็จะสามารถคัดลอกแผนที่บิตนั้นลงสู่พื้นที่ใช้งานของโปรแกรมได้

<u>โครงสร้างภาพแผนที่บิต</u>

ภายในระบบวินโดวส์ แผนที่บิตมักจะหมายถึงชุดหรือแมทริกซ์ของบิต (array or metrix of bits) ใน RAM ซึ่งเป็นตัวแทนของภาพ แผนที่บิตสามารถสร้างขึ้นได้ดังที่สร้างภาพขึ้นด้วยโปรแกรมที่วินโดวส์หลักของจอภาพ อีกทั้งยังสามารถคัดลอกแผนที่บิตจากหน่วยความจำไปแสดงไว้บนจอภาพ จากจอภาพกลับมาไว้ในหน่วยความจำ และจากแผนที่บิตหนึ่งไปยังอีกแผนที่บิตหนึ่งบนหน่วยความจำเดียวกัน

ในการสร้างแผนที่บิตนั้นจำเป็นต้องกำหนดรายละเอียดของอุปกรณ์ ในหน่วยความจำให้เข้ากันได้กับรายละเอียดเกี่ยวกับอุปกรณ์ของจอภาพก่อน โดยใช้คำสั่ง CreateCompatibleDC() รายละเอียดเกี่ยวกับอุปกรณ์ที่กำหนดขึ้นใหม่บนหน่วยความจำนั้น เรียกว่า รายละเอียดการแสดงผลของหน่วยความจำ (memory display context) ต่อจากนั้นจึงใช้รายละเอียดเกี่ยวกับอุปกรณ์นี้สร้างแผนที่บิตที่เข้ากันได้กับทั้งรายละเอียดเกี่ยวกับการแสดงผลบนหน่วยความจำและรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพของโปรแกรมที่วินโดวส์หลักบนจอภาพด้วยคำสั่ง CreateCompatibleBitmap() ก่อนที่จะวาดภาพใดๆ บนบิตแม็พนั้น จะต้องกำหนดเลือกบิตแม็พนั้นไว้ในรายละเอียดเกี่ยวกับการแสดงผลบนหน่วยความจำด้วยคำสั่ง SelectObject() เสียก่อน

การสร้างภาพกราฟิกบนแผนที่บิตนั้นสามารถทำได้ด้วยวิธีเดียวกับการสร้างภาพด้วยโปรแกรมในวินโดวส์หลักบนจอภาพ โดยใช้การจัดการของรายละเอียดเกี่ยวกับการแสดงผลบนหน่วยความจำของแผนที่บิตแทนรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพ เมื่อเรียกใช้คำสั่งการวาดภาพของจีดีไอ หรือ

สามารถที่จะวาดภาพบนวินโดวส์หลักของโปรแกรมก่อน แล้วจึงคัดลอกภาพนั้นๆ ลงไปไว้ในแผนที่บิต ก็ได้

การเคลื่อนย้ายแผนที่บิต แผนที่บิตสามารถถูกคัดลอกจากที่หนึ่งไปไว้ในอีกที่หนึ่งได้ด้วยคำสั่ง BitBlt() ในการที่จะคัดลอกแผนที่บิตไปไว้บนภาพที่ปรากฏอยู่แล้ว และให้แสดงผลลัพธ์ที่แตกต่างกัน นั้นสามารถทำได้โดยการใช้ raster operation ที่แตกต่างกัน ดังตัวอย่างที่ปรากฏอยู่ในรูป 4.2 คำสั่ง BitBlt() นั้นจะเป็นประโยชน์อย่างยิ่งสำหรับ partial screen animation และสำหรับการย้ายแผนที่บิตที่ฝังตัวอยู่ไปแสดงให้เห็นบนวินโดวส์

ในกรณีที่แผนที่บิตจำเป็นต้องถูกย่อหรือขยายเพื่อให้เหมาะสมกับที่หมายที่จะแสดงสามารถทำได้ด้วยคำสั่ง StretchBlt() ทั้งคำสั่ง StretchBlt() และ BitBlt() นั้นจะใช้ raster operation ในการควบคุมผลของการวางภาพแผนที่บิตลงบนภาพที่ได้ปรากฏอยู่แล้วในบริเวณนั้น และคำสั่งทั้งสองยังสามารถย้ายส่วนของแผนที่บิตต้นแบบที่อยู่ภายใต้กรอบสี่เหลี่ยมที่เลือกไว้ ไปไว้ยังแผนที่บิตเป้าหมายได้ด้วย ซึ่งวิธีนี้จะเป็นการอำนวยความสะดวกในการใช้ฉากหลัง สำหรับการเคลื่อนย้ายบนหน้าจอหลักของโปรแกรมประยุกต์ด้วยวิธีง่ายๆ โดยการคัดลอกรูปสี่เหลี่ยม ซึ่งมีขนาดใหญ่กว่าหน้าจอบนจอภาพที่ได้เลือกไว้จากแผนที่บิตที่ฝังตัวอยู่

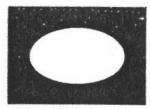
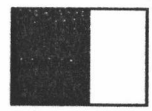
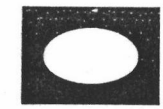
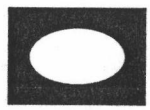
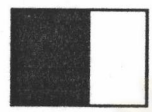
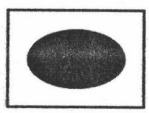
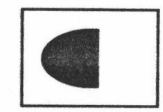
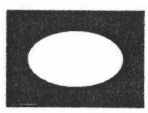
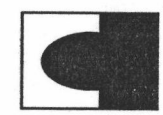
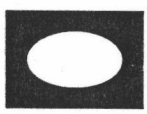
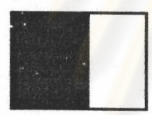
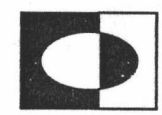
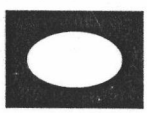
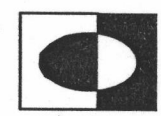
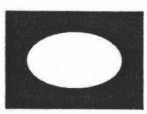
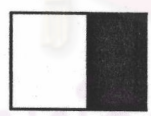
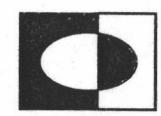
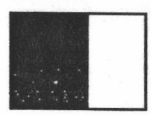
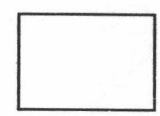
การลบแผนที่บิตหลังจากเลิกใช้งานแผนที่บิตนั้นๆ สามารถใช้คำสั่ง DeleteObject() ในการลบแผนที่บิตนั้นเพื่อเป็นการทำหน่วยความจำที่เคยใช้บันทึกแผนที่บิตนั้นว่างลง Microsoft (1992) ได้กล่าวถึงโครงสร้างของแผนที่บิตไว้ โดยแบ่งโครงสร้างหลักของแผนที่บิตออกเป็น

1. bmfh เป็นส่วนที่แสดงโครงสร้างของแฟ้มข้อมูลที่เก็บแผนที่บิต ใช้โครงสร้างข้อมูลแบบ BITMAPFILEHEADER

2. bmih เป็นส่วนที่แสดงโครงสร้างของแผนที่บิต ใช้โครงสร้างข้อมูลแบบ BITMAPINFOHEADER

3. aColos เป็นส่วนที่แสดงสีที่ใช้ในแผนที่บิต ใช้โครงสร้างข้อมูลแบบ RGBQUAD ซึ่งเป็น array ขนาด 16 elements ของ long integer

4. aBitmapBits เป็นส่วนที่แสดงภาพของแผนที่บิต ใช้โครงสร้างข้อมูลแบบ BYTE

Source Bitmap	Target Bitmap	Raster Operation	Boolean Logic	Resulting Image
		SRCCOPY	Overwrite	
		SRCPAINT	OR	
		MERGEPAINT	Invert source, then OR	
		NOTSRCERASE	Invert result of OR	
		SRCINVERT	XOR	
		SRCAND	AND	
		SRCERASE	Invert target, then AND	
Null		BLACKNESS	pattern	
Null		WHITENESS	pattern	

รูปที่ 4.2 แสดง Raster Operation ของภาพแผนที่บิต

รายละเอียดโครงสร้างส่วนหัวของแผนที่บิต ได้แก่ bmfh และ bmih ได้แสดงไว้ในตารางที่ 4.2 และ 4.3 ตามลำดับ

ตารางที่ 4.2 รายละเอียดโครงสร้าง bmfh ของแผนที่บิต

ชื่อข้อมูล	ความหมาย
Type	เป็นไบนารีคีย์ (Binary key) ที่บอกประเภทของไฟล์ มักถูกกำหนดค่าเป็นเลขฐานสิบหกเท่ากับ 4D42
Size	บอกขนาดของแฟ้มข้อมูลที่เก็บแผนที่บิต
Reserve1	สงวนไว้ ไม่ได้ใช้
Reserve2	สงวนไว้ ไม่ได้ใช้
OffsetBits	เป็นเลขฐานสิบหก เท่ากับ 76

ตารางที่ 4.3 รายละเอียดโครงสร้าง bmih ของแผนที่บิต

ชื่อข้อมูล	ความหมาย
Size	จำนวน ไบท์ที่เก็บ Bitmap Info Header
Width	บอกความกว้างของภาพแผนที่บิต
Height	บอกความสูงของภาพแผนที่บิต
Planes	บอกจำนวนบของภาพแผนที่บิต ปกติมีค่าเท่ากับ 1
Bit/pixel	บอกจำนวนบิตต่อจุดภาพ ปกติมีค่าเท่ากับ 4
Compression	บอกถึงรูปแบบของการอัดข้อมูลภาพแผนที่บิต ปกติเท่ากับ 0
SizeImage	ขนาดของภาพที่ไม่รวมข้อมูลส่วนหัวอื่นๆ
XPelsPerMeter	ปกติมีค่าเท่ากับ 0
YPelsPerMeter	ปกติมีค่าเท่ากับ 0
ColorsUsed	จำนวนสีที่ใช้ในภาพแผนที่บิต ปกติมีค่าเท่ากับ 16
ColorsImpotant	จำนวนสีหลักที่ใช้ในภาพแผนที่บิต ปกติมีค่าเท่ากับ ColorsUsed

โครงสร้างของภาพเมตาไฟล์

เมตาไฟล์คือชุดของคำสั่งในจีดีไอที่ถูกเรียกขึ้นมาใช้แล้วจะถูกเก็บไว้สำหรับการใช้งานอีกในภายหลังใน interactive graphic application เมตาไฟล์จะถูกใช้ในการเก็บการกระทำต่างๆ ของผู้ใช้งาน จีดีไอยังมีคำสั่งเกี่ยวกับการบันทึกเมตาไฟล์ลงไปยังหน่วยเก็บข้อมูล และการดึงเมตาไฟล์ที่ถูกบันทึกไว้ก่อนหน้ากลับขึ้นมาใช้งานจากหน่วยเก็บข้อมูลด้วย

การสร้างเมตาไฟล์ เมตาไฟล์จะถูกสร้างขึ้นบนหน่วยความจำด้วยคำสั่ง CreatMetaFile() การใส่ข้อมูลในเมตาไฟล์ คำสั่งทางด้านกราฟิกของจีดีไอตามปกติสามารถใช้ในการใส่ผลงานกราฟิกลงบนเมตาไฟล์ได้ เพียงแต่จะต้องสั่งผ่านการทำงานไปสู่เมตาไฟล์ โดยระบุในรายละเอียดเกี่ยวกับอุปกรณ์แทนที่จะเป็นการทำงานในรายละเอียดเกี่ยวกับการแสดงผลบนจอภาพ บนเมตาไฟล์สามารถวาดเส้น รูปสี่เหลี่ยม วงรี และรูปอื่นๆ ทั้งยังสามารถระบายสีขอบเขตและเคลื่อนย้ายแผนที่บิตได้ด้วย

การปิดเมตาไฟล์ หลังจากสิ้นสุดการบรรจุคำสั่งทางด้านกราฟิกของจีดีไอ แล้วควรที่จะปิดเมตาไฟล์นั้นก่อนที่จะเรียกใช้งานซ้ำ อีกทั้งยังต้องปิดก่อนทำการบันทึกลงบนหน่วยเก็บข้อมูลด้วย การปิดเมตาไฟล์นั้นทำได้โดยการใช้คำสั่ง CloseMetaFile()

การเรียกใช้งานเมตาไฟล์ การเรียกใช้งานชุดคำสั่งทางด้านกราฟิกของจีดีไอที่บรรจุไว้ในเมตาไฟล์จะต้องใช้คำสั่ง PlayMetaFile() ระบบวินโดวส์จะอ่านข้อมูลในเมตาไฟล์ขึ้นมาและจะทำงานตามคำสั่งของจีดีไอที่ได้พบ เมตาไฟล์จำเป็นต้องถูกปิดก่อนที่จะถูกเรียกใช้งานหรือถูกบันทึกลงบนหน่วยเก็บข้อมูลเสมอ

การบันทึกเมตาไฟล์ลงบนจานบันทึก คำสั่ง CopyMetaFile() จะใช้ในการบันทึกเมตาไฟล์ลงบนหน่วยเก็บข้อมูล การเรียกเมตาไฟล์กลับขึ้นมาจากหน่วยเก็บข้อมูล การที่จะเรียกเมตาไฟล์ซึ่งก่อนหน้านี้ได้ถูกบันทึกลงบนจานบันทึกขึ้นมาใช้อีกนั้น จะต้องใช้คำสั่ง GetMetaFile() ซึ่งจะย้ายเมตาไฟล์ลงสู่หน่วยความจำซึ่งสามารถใช้คำสั่ง PlayMetaFile() เรียกใช้งานได้

การลบเมตาไฟล์หลังจากสิ้นสุดการเรียกใช้งานเมตาไฟล์แล้วควรใช้คำสั่ง DeleteMetaFile() เพื่อลบเมตาไฟล์ที่อยู่บนหน่วยความจำเสียโดยที่เมตาไฟล์ที่ถูกบันทึกลงบนจานบันทึกแล้ว จะไม่ได้รับผลการะทบจากคำสั่งนี้

ข้อแตกต่างของเมตาไฟล์กับแผนที่บิตมีอยู่หลายประการ แม้ว่าทั้งสองจะเป็นโครงสร้างมาตรฐานในการเก็บภาพในวินโดวส์เหมือนกันก็ตาม ข้อแตกต่างที่สำคัญก็คือ แผนที่บิตจะเก็บภาพที่วาดด้วยฟังก์ชันวาดภาพ ส่วนเมตาไฟล์เก็บคำสั่งที่ใช้ในการวาดภาพ บุญเลิศ เอี่ยมทัศนา (2537) ได้สรุปข้อแตกต่างไว้ดังนี้

1. ไฟล์ของเมตาไฟล์จะมีขนาดเล็กกว่าแผนที่บิตมาก

2. เมตาไฟล์สามารถแสดงภาพซ้อนทับกันโดยมองทะลุช่องว่างได้ แต่แผนที่บิตทำไม่ได้ คือจะซ้อนทับกันสนิท

3. เมตาไฟล์สามารถเลื่อนภาพเล็กในภาพใหญ่ได้ ขณะที่แผนที่บิตทำเช่นนี้ไม่ได้

4. เมตาไฟล์สามารถย่อและขยายภาพ โดยคงภาพเดิมได้ดีกว่าแผนที่บิต

5. การแสดงภาพของเมตาไฟล์ จะช้ากว่าแผนที่บิต เพราะต้องทำการแปลและดำเนินการทีละคำสั่ง

6. หากต้องการให้ใช้งานได้ทั่วไป จะต้องใช้สีเพียง 16 สี ต่างกับบิตแม็พที่อาจถือได้ว่าไม่จำกัด

แม้จะมีความแตกต่างระหว่างรูปแบบการเก็บภาพทั้งสองของวินโดวส์ แต่เมตาไฟล์ก็ยังมีคุณสมบัติที่เหนือกว่า คือสามารถรวมภาพแผนที่บิต เข้าไว้ในภาพของเมตาไฟล์ได้ จึงเป็นการรวมคุณสมบัติของทั้งสองรูปแบบเข้าด้วยกัน

บุญเลิศ เอี่ยมทัศนา (2537) ได้กล่าวถึงประเภทของเมตาไฟล์ที่ใช้งานอยู่ในวินโดวส์ โดยแบ่งออกเป็น 2 ชนิดคือ เมตาไฟล์ธรรมดา และเพลซเอเบิลเมตาไฟล์ (Placeable Meta File) ซึ่งจะต่างจากเมตาไฟล์ธรรมดา โดยมีข้อมูลเพิ่มอีก 22 ไบท์ ที่ส่วนหัว (Header) ของไฟล์เป็นข้อมูลเกี่ยวกับสัดส่วน (aspect ratio) และขนาดเดิม (original size) ของรูปภาพในเมตาไฟล์นั้น ทั้งนี้เพื่อเป็นการแก้ไขข้อบกพร่องเดิมของเมตาไฟล์แบบธรรมดา ที่ไม่มีข้อมูลส่วนนี้บันทึกไว้ เป็นผลให้การแสดงภาพเมตาไฟล์อาจบิดเบือนไปจากเดิมที่ผู้สร้างได้กำหนดไว้

Microsoft (1992) ได้กล่าวถึงโครงสร้างของเมตาไฟล์ไว้ โดยแบ่งโครงสร้างหลักของเมตาไฟล์ออกเป็นส่วนหัว (Header) และระเบียน (Record) ย่อย ซึ่งเก็บชุดคำสั่งประเภทต่างๆ ของจีดีไอไว้ รายละเอียดโครงสร้างส่วนหัวของเมตาไฟล์ ได้แสดงไว้ในตารางที่ 4.4 ส่วนระเบียนย่อยต่างๆ ของเมตาไฟล์ได้แสดงไว้ในภาคผนวก ข.

ตารางที่ 4.4 รายละเอียดโครงสร้างส่วนหัวของเมตาไฟล์

ชื่อข้อมูล	ความหมาย
Key	เป็นไบนารีคีย์ (Binary key) ที่บอกประเภทของไฟล์ มักถูกกำหนดค่าเป็นเลขฐานสิบหกเท่ากับ 9AC6CDD7
hmf	ไม่ได้ใช้
bbox	บอกถึงพิกัดของจุดที่เป็นขอบของสี่เหลี่ยมที่เล็กสุด และสามารถครอบคลุมภาพของเมตาไฟล์นี้ได้
inch	บอกหน่วยที่ใช้ในเมตาไฟล์เป็นนิ้ว
reserved	ไม่ได้ใช้
checksum	เป็นตัวตรวจสอบ checksum โดยจะใช้การ XOR ด้วย 10 ไบท์แรกของ header
mtType	บอกประเภทของเมตาไฟล์ คือ เป็นแบบที่จัดเก็บในหน่วยความจำ หรือในแฟ้มข้อมูล
mtHeaderSize	บอกขนาดของ header
mtVersion	บอกรุ่นของเมตาไฟล์
mtSize	บอกขนาดของข้อมูลที่เก็บเมตาไฟล์
mtNoObject	บอกจำนวนสูงสุดของวัตถุที่สามารถเก็บไว้ในเมตาไฟล์เดียวกัน
mtMaxRecord	บอกขนาดของระเบียนที่ใหญ่สุดในเมตาไฟล์
mtNoParameter	ไม่ได้ใช้งาน

การแสดงตัวอักษร

คำสั่งเกี่ยวกับรูปแบบตัวอักษรที่มีอยู่ในจีดีไอ แบ่งเป็น 2 ประเภทคือ รูปแบบตัวอักษรและตัวอักษร คำสั่งเกี่ยวกับรูปแบบตัวอักษรจะใช้ในการสร้างและคัดเลือกรูปแบบตัวอักษร ส่วนคำสั่งเกี่ยวกับตัวอักษรนั้นจะใช้ในการแสดงตัวอักษรบนจอภาพ ระบบวินโดวส์จะมีรูปแบบตัวอักษรที่บรรจุอยู่ภายในซึ่งโปรแกรมต่างๆ สามารถเรียกใช้ได้ขณะทำงาน

การแสดงตัวอักษร คำสั่ง TextOut() จะใช้ในการแสดงตัวอักษร โดยจะแสดงรูปแบบตัวอักษรและสีของตัวอักษรตามที่ขณะนั้นได้กำหนดให้ไว้ ระบบวินโดวส์จะมีรูปแบบตัวอักษรที่บรรจุอยู่ภายในซึ่งโปรแกรมกราฟิกสามารถเรียกขึ้นมาและใช้งานได้

การกำหนดสีของตัวอักษร การกำหนดสีเพื่อใช้ในการเขียนตัวอักษรนั้นจะต้องใช้คำสั่ง SetTextColor() ส่วนการกำหนดสีเพื่อใช้เป็นฉากหลังของส่วนที่เป็นตัวอักษรนั้นใช้คำสั่ง SetBkMode() และคำสั่ง SetBkMode() จะใช้ในการกำหนดว่าฉากหลังในส่วนที่เป็นตัวหนังสือนั้นจะทึบแสงหรือโปร่งแสง

การเลือกรูปแบบตัวอักษร ในการที่จะเลือกรูปแบบตัวอักษรที่มีไว้ให้ในระบบวินโดวส์นั้นต้องใช้คำสั่ง GetStockObject() และต้องกำหนดเลือกรูปแบบตัวอักษรไว้ในรายละเอียดเกี่ยวกับการแสดงผลทางจอภาพด้วยคำสั่ง SelectObject() โดยการเรียกใช้คำสั่ง TextOut() ต่อจากนี้ไปจะใช้รูปแบบตัวอักษรที่ได้เลือกไว้แล้วนี้

การเลือกรูปแบบและขนาดที่แตกต่างกันไป คำสั่ง CreatFont() จะใช้ในการสร้างรูปแบบตัวอักษรที่มีคุณลักษณะสมเหตุสมผลกับขนาดและรูปแบบที่ต้องการ ส่วนใหญ่ระบบวินโดวส์จะพยายามคัดเลือกรูปแบบตัวอักษรที่มีอยู่แล้วให้เข้ากันได้ดีที่สุดกับความต้องการ

การจัดวางตัวอักษร การแสดงตัวอักษรในพื้นที่รูปสี่เหลี่ยมผืนผ้านั้นใช้คำสั่ง ExtTextOut() การปรับขนาดของตัวอักษรในแต่ละบรรทัด (Justify) นั้นใช้คำสั่ง SetTextJustification() และการค้นหาที่ว่างสำหรับ text string นั้นใช้คำสั่ง GetTextMetrix()

การกำหนดระบบพิกัด

จีดีไอของระบบวินโดวส์ จะมีคำสั่งเกี่ยวกับระบบพิกัดที่จะใช้กำหนดแผนที่จุดของภาพกราฟิกจากพิกัดหนึ่งไปสู่อีกพิกัดหนึ่ง ในระบบพิกัดที่ได้กำหนดไว้เดิมนั้น จีดีไอจะจัดพิกัด X,Y ด้วยระบบ 1 ต่อ 1 กับจุดภาพ ในหน้าจอหลักของโปรแกรมบนจอภาพ จุดกำเนิด (0,0) จะอยู่ที่มุมบนซ้ายของพื้นผิวที่ใช้งานวาดภาพ พิกัด X จะเพิ่มขึ้นไปทางขวา ขณะที่พิกัด Y จะเพิ่มขึ้นเมื่อเคลื่อนลงข้างล่างของหน้าจอ ระบบพิพัดแบบนี้เรียกว่า MM_TEXT

ระบบพิกัดแบบอื่นก็มีให้เลือกใช้ได้โดยเฉพาะกับโปรแกรมกราฟิกอย่าง drafting, page layout และอื่นๆ ระบบ MM_LOENGLISH จะจัดแต่ละหน่วยเป็น 0.01 นิ้วบนพื้นผิวที่ใช้ในการแสดงผล ขณะที่ระบบ MM_LOMETRIC จะจัดแต่ละ logical unit เป็น 0.1 มิลลิเมตร บนพื้นผิวที่ใช้ในการแสดงผลนั้น และระบบ MM_AMSOTROPIC จะจัดแต่ละ logical unit ตามขนาดที่ผู้เขียนโปรแกรมกำหนดไว้บนพื้นผิวในการแสดงผล ระบบพิกัดที่มีให้ใช้ในระบบพิกัดของวินโดวส์ ได้แสดงไว้ในตารางที่ 4.5

ตารางที่ 4.5 แสดงระบบพิกัดที่มีใช้ใน Mapping mode ของวินโดวส์

ชื่อระบบพิกัด	ความหมาย
MM_ANISOTROPIC	เป็นระบบที่เปลี่ยนจาก logocal unit ไปเป็น arbitary unit หรือหน่วยพิกัดที่ผู้ใช้ต้องการ โดยการเปลี่ยนมาสู่ระบบพิกัดนี้ จะไม่ส่งผลกระทบต่อการปรับค่า window และ viewport ที่ผ่านมา หากผู้ใช้ต้องการเปลี่ยน window และ viewport ให้เป็นตามระบบพิกัดใหม่นี้ จะต้องทำการปรับค่าต่างๆ ด้วยคำสั่ง SetWindowExt() และ SetViewportExt()
MM_HIENGLISH	เป็นระบบที่เปลี่ยนทุกค่าของ logical unit ให้เท่ากับ 0.001 นิ้ว โดยค่าในแกน X มีค่าเพิ่มขึ้นไปทางขวา และค่าในแนวแกน Y มีค่าเพิ่มขึ้นทางด้านบน
MM_HIMETRIC	เป็นระบบที่เปลี่ยนทุกค่าของ logical unit ให้เท่ากับ 0.01 มิลลิเมตร โดยค่าในแกน X มีค่าเพิ่มขึ้นไปทางขวา และค่าในแนวแกน Y มีค่าเพิ่มขึ้นทางด้านบน

ตารางที่ 4.5 แสดงระบบพิกัดที่มีใช้ใน Mapping mode ของวินโดวส์ (ต่อ)

ชื่อระบบพิกัด	ความหมาย
MM_ISOTROPIC	เป็นระบบที่เปลี่ยนจาก logocal unit ไปเป็น arbitary unit หรือหน่วยพิกัดที่ผู้ใช้ต้องการ โดยกำหนดให้แต่ละหน่วยในแนวแกน X จะต้องเท่ากันในแนวแกน Y ด้วย การปรับค่า window และ viewport ด้วยคำสั่ง SetWindowExt() และ SetViewportExt() ยังคงใช้ได้ เพียงแต่ จีดีไอจะทำการปรับค่าทั้ง 2 แกนให้เท่ากันตลอดเวลา
MM_LOENGLISH	เป็นระบบที่เปลี่ยนทุกค่าของ logical unit ให้เท่ากับ 0.01 นิ้ว โดยค่าในแกน X มีค่าเพิ่มขึ้นไปทางขวา และค่าในแนวแกน Y มีค่าเพิ่มขึ้นทางด้านบน
MM_LOMETRIC	เป็นระบบที่เปลี่ยนทุกค่าของ logical unit ให้เท่ากับ 0.1 มิลลิเมตร โดยค่าในแกน X มีค่าเพิ่มขึ้นไปทางขวา และค่าในแนวแกน Y มีค่าเพิ่มขึ้นทางด้านบน
MM_TEXT	เป็นระบบที่เปลี่ยนทุกค่าของ logical unit ให้เท่ากับ 1 device pixel โดยค่าในแกน X มีค่าเพิ่มขึ้นไปทางขวา และค่าในแนวแกน Y มีค่าเพิ่มขึ้นทางด้านล่าง
MM_TWIPS	เป็นระบบที่เปลี่ยนทุกค่าของ logical unit ให้เท่ากับ 1/20 ของ point (1 point เท่ากับ 1/72 นิ้ว ดังนั้น 1 twip จึงเท่ากับ 1/1440 นิ้ว) โดยค่าในแกน X มีค่าเพิ่มขึ้นไปทางขวา และค่าในแนวแกน Y มีค่าเพิ่มขึ้นทางด้านบน

การใช้ระบบพิกัดจะทำงานโดยใช้ viewport และ window ในการอธิบายถึงระบบพิกัดนั้น คำว่า window จะมีความหมายในทางกราฟิกแบบดั้งเดิม ไม่ได้หมายความถึงหน้าจอของโปรแกรมที่ใช้กันโดยทั่วไป แต่จะหมายถึง abstract scaling rectangle

viewport จะถูกระบุด้วยพิกัดทางกายภาพ viewport จะหมายถึง physical coordinate space ส่วน window จะเป็น logical coordinates space เมื่อเรียกใช้คำสั่งทางด้านกราฟิกของจีดีไอ logical coordinate จะถูกกำหนดให้ไว้ด้วยระบบพิกัดที่ถูกเลือกใช้จะพิจารณาถึงลักษณะที่จีดีไอจัด logical coordinate บนจอภาพโดยใช้ physical coordinate ซึ่งตามที่กำหนดไว้เดิมจะเป็น 1:1 ของ logical coordinate ที่กำหนด

ให้ไว้จะถูกจัดให้ต่างกับ physical coordinate space ซึ่งหน้าจอหลักของโปรแกรมก็จะอยู่ใน physical coordinate space นี้ด้วย

คำสั่งเกี่ยวกับระบบพิกัดสามารถใช้ในการจัดวางภาพได้โดยอัตโนมัติ หรือจะใช้คำสั่งเกี่ยวกับพิกัดของจีดีไอ จัดวางภาพแต่ละพิกัดในแต่ละครั้งได้ ตัวอย่างเช่นคำสั่ง LPtoDP() ของจีดีไอจะเปลี่ยน logical coordinate ไปเป็น window coordinate และคำสั่ง DPtoLP() จะเปลี่ยน window coordinate กลับเป็น logical coordinate

<u>การแสดงผลทางเครื่องพิมพ์</u>

โปรแกรมภายใต้ระบบวินโดวส์เรียกใช้ระบบ window print manager ในการพิมพ์ภาพออกมาตาม parameter ของเครื่องพิมพ์ที่ปรากฏอยู่ได้ ไดร์เวอร์ของอุปกรณ์ในการพิมพ์สามารถเรียกขึ้นมาเพื่อแก้ไขได้ การเริ่มและการหยุดพิมพ์ คำสั่ง Escape() ใช้สำหรับการเริ่มพิมพ์และหยุดการพิมพ์ คำสั่ง Escape() ยังใช้ในการควบคุมการแบ่งหน้ากระดาษอีกด้วย

การปรับแก้งานพิมพ์ ในการปรับขนาดหรือการจัดวางภาพเพื่อให้ได้ตามต้องการ โดยเฉพาะอย่างยิ่งบนเครื่องพิมพ์แบบเลเซอร์นั้น ขั้นตอนแรกต้องเรียกใช้คำสั่ง GetProfileString() เพื่อที่จะอ่านข้อมูลเกี่ยวกับเครื่องพิมพ์ ที่อยู่ในแฟ้มข้อมูล WIN.INI ขึ้นมา ต่อจากนั้นจึงใช้คำสั่ง LoadLibrary() เรียก printer device ขึ้นมาใช้ หลังจากใช้ตัวขับเคลื่อนในการแก้ไขเครื่องพิมพ์แล้ว จึงเริ่มการพิมพ์ คำสั่ง GetDeviceCaps() จะเป็นประโยชน์อย่างยิ่งในการขอดูข้อมูลเกี่ยวกับความสามารถในการผลิตงานกราฟิกของเครื่องพิมพ์ที่ติดตั้งอยู่ รวมไปถึงข้อมูลที่ว่าเครื่องพิมพ์สามารถรองรับคุณลักษณะของแผนที่บิต dithering และอื่นๆ ได้หรือไม่