บทที่ 4

การออกแบบระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูล

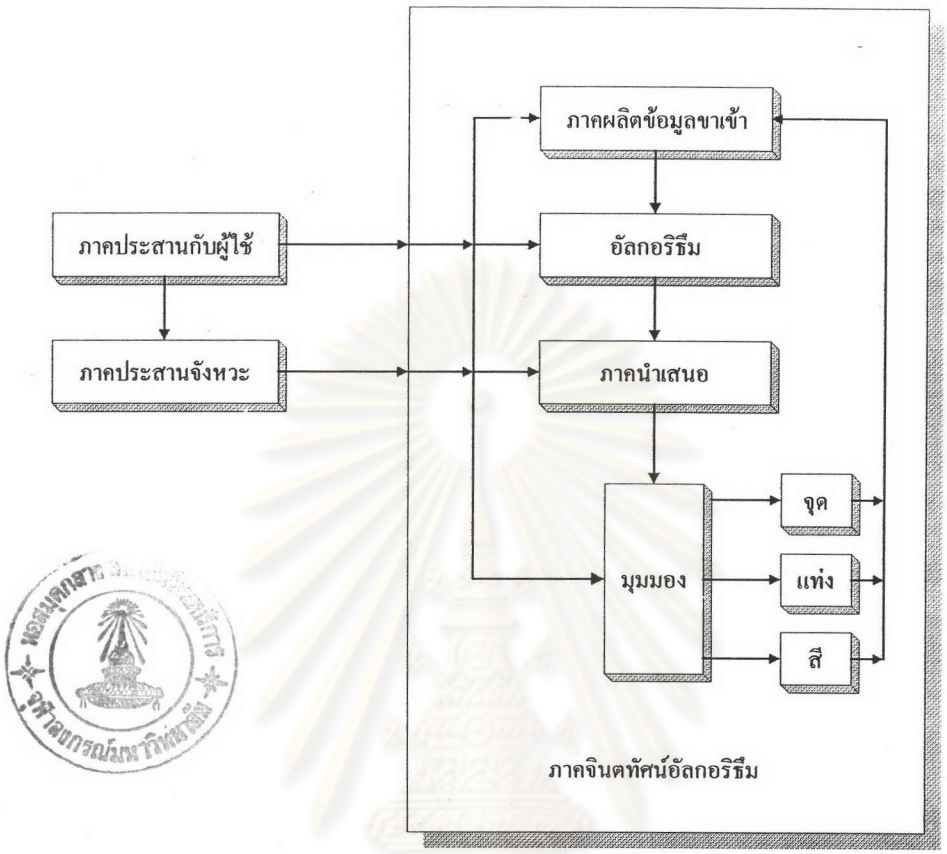
โครงสร้างหลักของระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูล

การจินตทัศน์อัลกอริทึมจะได้ผลก็ต่อเมื่อมีระบบที่เอื้ออำนวยในการสร้างการจินตทัศน์และการใช้งานให้เกิดความเข้าใจง่ายน่าติดตาม ในแง่ของผู้สร้างการจินตทัศน์ระบบจะต้องเอื้ออำนวยชุดหรือคลังคำสั่งในการแสดงภาพการเคลื่อนไหวของภาพ การจัดการเกี่ยวกับมุมมองในการนำเสนอแบบต่างๆ การรับหรือสร้างข้อมูล การโต้ตอบกับผู้ใช้ กลไกการประสานจังหวะการทำงานระหว่างอัลกอริทึมและการเปลี่ยนแปลงภาพในมุมมองต่าง ๆ

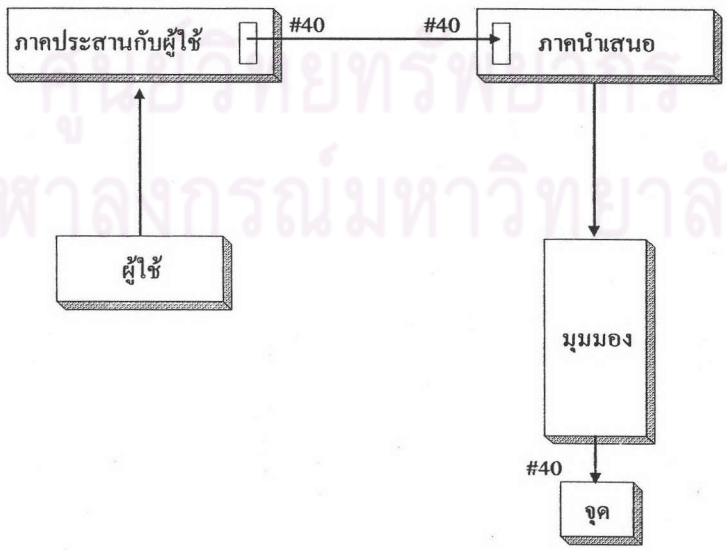
ส่วนในแง่ของผู้ใช้ ปัจจัยที่สำคัญที่สุดคือความสะดวกในการใช้งานระบบจินตทัศน์อัลกอริทึมจะต้องมีลักษณะการติดต่อประสานกับผู้ใช้อย่างเป็นเอกภาพตรงตามสามัญสำนึก สามารถเปิดโอกาสให้ผู้ใช้ขัดจังหวะ เร่ง หน่วง หยุด หรือเลิกการทำงาน ระหว่างการจินตทัศน์ได้ตามต้องการ

โครงสร้างหลักของระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลนี้ประกอบด้วยภาคต่าง ๆ ดังแสดงในรูปที่ 4.1 ซึ่งภาคต่างๆ เหล่านี้ จะมีการถ่ายทอดส่งสัญญาณหรือข้อความซึ่งกันและกัน ณ จุดนี้เองที่เราได้นำคุณลักษณะสำคัญที่มีในวินโดวส์มาใช้งานนั่นคือ การแลกเปลี่ยนข้อมูลแบบพลวัตนั่นเอง รูปที่ 4.2 เป็นตัวอย่างอันหนึ่งของการใช้การแลกเปลี่ยนข้อมูลแบบพลวัตซึ่งเกิดขึ้นเมื่อผู้ใช้สั่งงานให้โปรแกรมแสดงมุมมองแบบจุด

จากรูปที่ 4.2 เมื่อผู้ใช้ติดต่อผ่านทางภาคประสานกับผู้ใช้ว่าต้องการดูมุมมองในการนำเสนอแบบจุด (ค่า #40) ภาคประสานผู้ใช้ก็จะทำการใส่ค่าสตริง (string) #40 ลงในชิ้นข้อมูลของตนเองและด้วยคุณสมบัติของการแลกเปลี่ยนข้อมูลแบบพลวัตก็จะทำให้ภาคนำเสนอมองเห็นชิ้นข้อมูลนั้นด้วยซึ่งแปลความหมายได้ว่าให้ทำการสร้างมุมมองการนำเสนอแบบจุด ในการทำงานลักษณะเดียวกันนี้ยังปรากฏอยู่อีกในส่วนของภาคประสานจังหวะกับอัลกอริทึมเมื่อผู้ใช้มีการขัดจังหวะ ปรับเร่ง ความเร็วหรือหน่วง การทำงานของอัลกอริทึม ดังนั้นเราจึงต้องมีการกำหนดรูปแบบการสนทนาเพื่อให้ภาคต่างๆสามารถแปลความหมายในทางเดียวกัน



รูปที่ 4.1 โครงสร้างหลักของระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูล



รูปที่ 4.2 การแลกเปลี่ยนข้อมูลแบบพลวัต

การกำหนดรูปแบบการสนทนาเพื่อให้ภาคต่างๆสามารถแปลความหมายในทางเดียวกันมีดังนี้คือ

ค่าของสตริง	ความหมาย
"#10"	สั่งการให้ภาคนำเสนอทราบว่าผู้ใช้ได้ทำการสร้างข้อมูลขาเข้าใหม่จะต้องอ่านข้อมูลเหล่านั้นขึ้นมาแล้วจึงทำการปรับปรุงภาพในมุมมองต่างๆให้สอดคล้องกับข้อมูลชุดใหม่
"#11"	สั่งการให้ภาคนำเสนอย้อนสถานะกลับเป็นสถานะเริ่มต้นก่อนมีการเรียงลำดับข้อมูล
"#12"	สั่งการให้ภาคนำเสนอทำการวาดภาพในมุมมองใหม่
"#20"	สั่งการให้ภาคนำเสนอทำการจัดระเบียบมุมมองต่างๆให้ซ้อนกัน
"#21"	สั่งการให้ภาคนำเสนอทำการจัดระเบียบมุมมองต่างๆโดยให้เห็นมุมมองทั้งหมด
"#34"	สั่งการให้ภาคอัลกอริทึมขอบริการการประสานจังหวะ
"#30"	สั่งการให้ภาคประสานจังหวะหยุดการให้จังหวะชั่วคราว
"#31"	สั่งการให้ภาคประสานจังหวะให้จังหวะทีละหนึ่งจังหวะ
"#32"	สั่งการให้ภาคประสานจังหวะทำงานต่อไป
"#33"	สั่งการให้ภาคประสานจังหวะหยุดการให้จังหวะ
"#40"	สั่งการให้ภาคนำเสนอทำการสร้างมุมมองแบบจุด
"#41"	สั่งการให้ภาคนำเสนอทำการสร้างมุมมองแบบแท่ง
"#42"	สั่งการให้ภาคนำเสนอทำการสร้างมุมมองแบบแถบสี
"#80"	สั่งการให้ภาคอัลกอริทึมทำการเรียงลำดับจากน้อยไปมาก
"#81"	สั่งการให้ภาคอัลกอริทึมทำการเรียงลำดับจากมากไปน้อย

จากการกำหนดความหมายในการสนทนาข้างต้นสามารถนำมานิยามเป็นค่าคงที่โดยการใช้ภาษาวิชวลเบสิกได้ดังนี้

```
'Constant For DDE With Client

Global Const REGENERATE        = "#10"
Global Const RESET_VIEW        = "#11"
Global Const REPAINT_VIEW      = "#12"
Global Const WINDOW_CASCADE    = "#20"
Global Const WINDOW_TILE       = "#21"
Global Const WINDOW_ARRANGE    = "#22"
Global Const RUN_START         = "#34"
Global Const RUN_PLAY          = "#30"
Global Const RUN_PAUSE         = "#31"
Global Const RUN_NEXT          = "#32"
Global Const RUN_STOP          = "#33"
Global Const VIEW_POINTS       = "#40"
Global Const VIEW_STICK_BARS   = "#41"
Global Const VIEW_COLOR        = "#42"
Global Const SORT_ASCENDING    = "#80"
Global Const SORT_DESCENDING   = "#81"
```

ค่าคงที่เหล่านี้จะถูกนำไปใช้ในการพัฒนาระบบเพื่อทำให้การติดต่อสื่อความหมายระหว่างภาคต่างๆที่มีการใช้การแลกเปลี่ยนแบบพลวัตเป็นไปอย่างมีประสิทธิภาพ

ภาคจินตทัศน์อัลกอริทึม

ภาคจินตทัศน์อัลกอริทึมจะประกอบไปด้วยส่วนจำเพาะย่อย ๆ ดังนี้คือ

1. อัลกอริทึม

สำหรับในโครงงานวิจัยนี้อัลกอริทึมคือขั้นตอนวิธีการเรียงลำดับข้อมูล ในการพัฒนาระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลนี้นอกจากการที่เราจะต้องศึกษาการทำงานของอัลกอริทึมแต่ละแบบอย่างละเอียดแล้วยังจะต้องพิจารณาด้วยว่า ณ จุดใดในขั้นตอนการทำงานของอัลกอริทึมที่สามารถแสดงให้เห็นถึงการเปลี่ยนแปลงของภาพการแสดงผลได้อย่างชัดเจนและผู้ใช้เกิดความเข้าใจ จากการศึกษาพบว่าจุดที่มีการสลับตำแหน่งของข้อมูลจะสามารถแสดงคุณสมบัติดังกล่าวได้ชัดเจนที่สุด

รูปที่ 4.3 แสดงอัลกอริทึมการเรียงลำดับข้อมูลแบบเลือกโดยใช้ภาษาวิชวลเบสิก ในบรรทัดที่ 15,16 และ 17 คือจุดที่ข้อมูลถูกสลับตำแหน่งและเป็นจุดที่สามารถนำเสนอการเปลี่ยนแปลงของรูปภาพได้อย่างชัดเจน ฟังก์ชัน OutputViewEvent จึงถูกนำมาใช้เพื่อทำการเปลี่ยนแปลงภาพในมุมมอง อีกสิ่งหนึ่งซึ่งมีความสำคัญเช่นกันก็คือการวัดประสิทธิภาพการทำงานของอัลกอริทึมแต่ละแบบโดยการใช้จำนวนครั้งของการเปรียบเทียบค่าของข้อมูล อัลกอริทึมใดมีการเปรียบเทียบมากครั้งกว่าอีกอัลกอริทึมหนึ่งก็ย่อมแสดงว่ามีประสิทธิภาพด้อยกว่า ในโปรแกรมย่อย

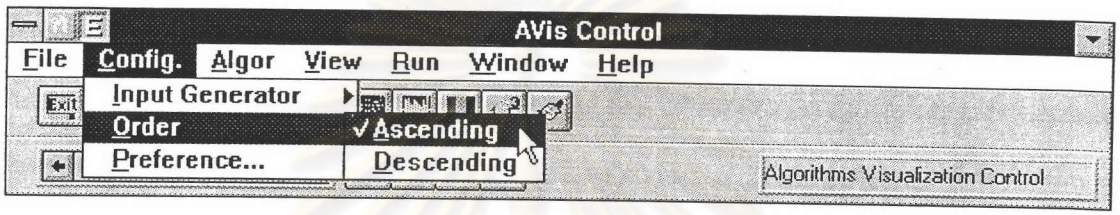
นี้มีตัวแปรชนิดเลขจำนวนเต็มชื่อ nCompare (ในบรรทัดที่ 7 และ 10) ตัวแปรนี้จะถูกเพิ่มค่าทีละหนึ่งทุกครั้งที่มีการเปรียบเทียบ นอกจากนั้นระบบจินตทัศน์อัลกอริทึมควรจะอนุญาตให้ผู้ใช้ทำการจินตทัศน์ได้หลายๆอัลกอริทึมพร้อมกัน ผู้พัฒนาจะต้องคำนึงถึงจุดนี้เนื่องจากแต่ละอัลกอริทึมจะต้องได้โอกาสในการทำงานเท่าเทียมกันจึงจะทำให้การเปรียบประสิทธิภาพเป็นไปอย่างอย่างถูกต้อง

จากข้อมูลข้างต้นทำให้เราได้ทราบแล้วว่าประสิทธิภาพของอัลกอริทึมอยู่ที่จำนวนครั้งของการเปรียบเทียบดังนั้นเมื่ออัลกอริทึมหนึ่งเสร็จสิ้นจากเปรียบเทียบหนึ่งครั้งก็ควรจะหยุดรอให้อัลกอริทึมอื่นๆทำงานจนเสร็จการเปรียบเทียบหนึ่งครั้งเช่นกัน ดังนั้นตัวอย่างในรูปที่ 4.3 จะเห็นได้ว่าเมื่ออัลกอริทึมการเรียงลำดับข้อมูลแบบเลือกได้ทำงานมาจนเสร็จการเปรียบเทียบหนึ่งครั้ง (บรรทัดที่ 6) ก็จะมีการเรียกใช้ฟังก์ชันชื่อ NotifySync( ) ผลของการเรียกใช้ฟังก์ชั่นนี้จะทำให้อัลกอริทึมนี้หยุดการทำงานชั่วคราวเพื่อเปิดโอกาสให้กับอัลกอริทึมอื่นทำงานจนเสร็จขั้นตอนการเปรียบเทียบหนึ่งครั้งครบทุกอัลกอริทึม (ในอัลกอริทึมอื่นก็จะมีการเรียกใช้ฟังก์ชัน NotifySync( ) เช่นกัน) จึงจะกลับมาทำงานต่อไป การทำงานลักษณะนี้จึงทำให้เราสามารถประกันได้ว่าทุกๆอัลกอริทึมที่กำลังทำงานไปพร้อมๆกันจะได้รับโอกาสในการทำงานเท่าเทียมกัน สำหรับรายละเอียดของฟังก์ชัน NotifySync( ) นี้จะกล่าวถึงโดยละเอียดในภาคประสานจังหวะ

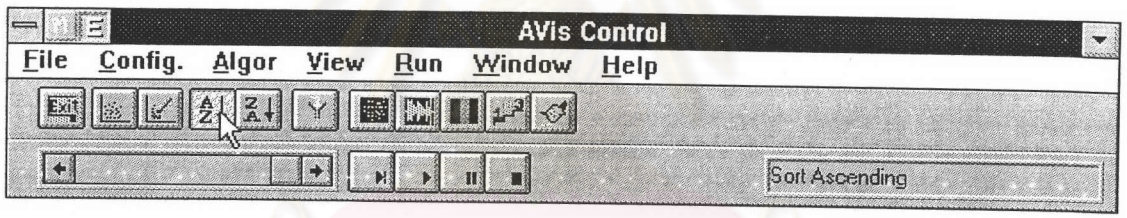
```
01)     Sub SelectionSort_ASC( )
02)       n= NumData - 1
03)       For idx = n TO 0 Step - 1
04)            max = idx
05)            For jdx = idx - 1 TO 0 Step - 1
06)                 If item(jdx) > item(max) Then
07)                      nCompare = nCompare + 1
08)                      i = NotifySync( )
09)                      max = jdx
10)                 Else
11)                      nCompare = nCompare + 1
12)                      i = NotifySync( )
13)                 End If
14)            Next jdx
15)            tmp = item(max)
16)            item(max) = item(idx)
17)            item(idx) = tmp
18)            OutputViewEvent
19)       Next idx
20)     End Sub
```

รูปที่ 4.3 ตัวอย่างอัลกอริทึมการเรียงลำดับข้อมูลแบบเลือก

เนื่องจากระบบที่ผู้วิจัยได้ออกแบบไว้นั้นผู้ใช้สามารถดูการทำงานของอัลกอริทึมการเรียงลำดับข้อมูลทั้งจากทิศทางจากน้อยไปหามากหรือจากมากไปหาน้อยได้โดยการเลือกจากเมนูของโปรแกรมหรือแถบเครื่องมือ (tools bar) ได้ดังแสดงในรูปที่ 4.5 และ 4.6 ตามลำดับ เพื่อเป็นการเพิ่มประสิทธิภาพในด้านความเร็ว ในการทำงานของโปรแกรม ผู้วิจัยจึงได้แยกอัลกอริทึมการเรียงลำดับข้อมูล แต่ละแบบออกเป็น 2 ชุด เพื่อทำงานในลักษณะจากน้อยไปมาก และจากมากไปน้อย ตามลำดับ



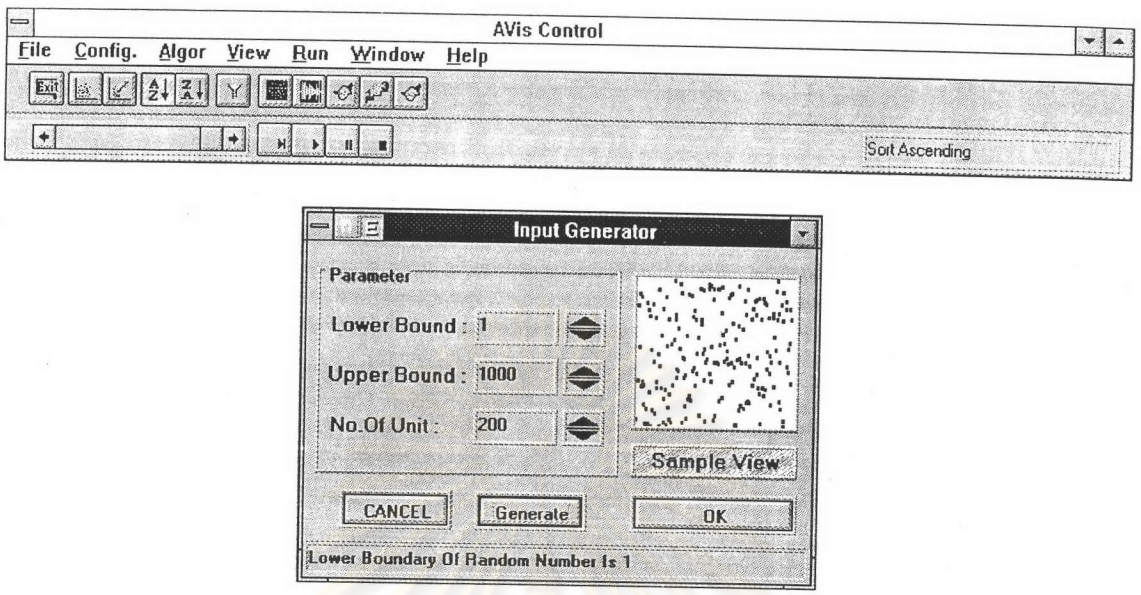
รูปที่ 4.5 การเลือกทิศทางการเรียงลำดับโดยใช้เมนู



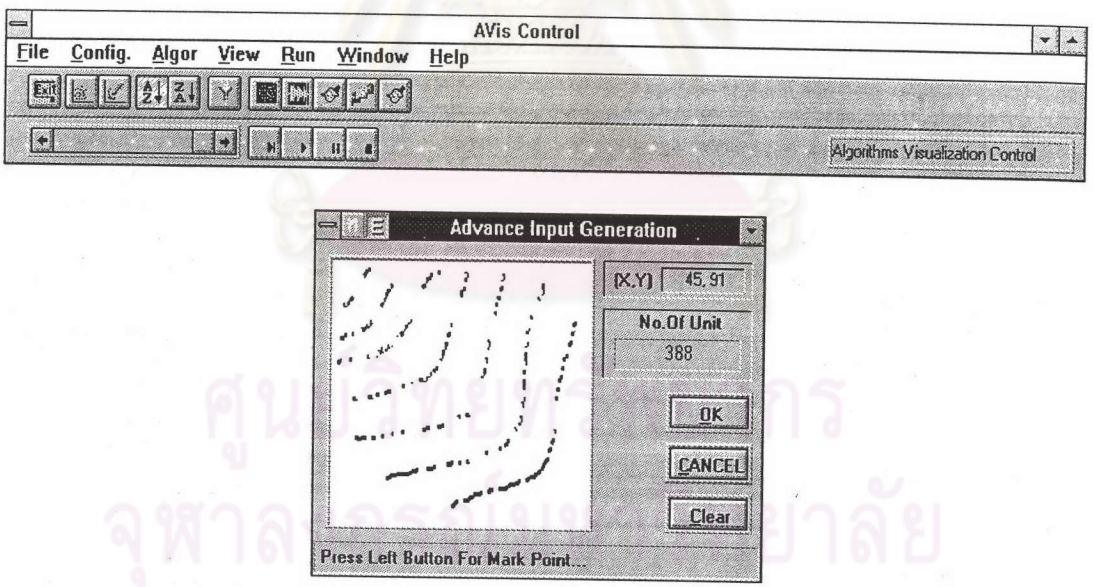
รูปที่ 4.6 การเลือกทิศทางการเรียงลำดับโดยใช้แถบเครื่องมือ

2. ภาคผลิตข้อมูลขาเข้า

หลังจากที่เลือกอัลกอริทึมแล้วผู้ใช้สามารถเลือกภาคผลิตข้อมูลขาเข้าเพื่อทำการระบุลักษณะของข้อมูลขาเข้าที่จะให้กับอัลกอริทึมระหว่างการทำงาน สำหรับโครงงานวิจัยนี้ผู้วิจัยได้ออกแบบให้ผู้ใช้สามารถเลือกลักษณะของข้อมูลได้ 2 แบบ คือ ลักษณะเป็นข้อมูลแบบสุ่มโดยสามารถกำหนด จุดสูงสุด ต่ำสุดของค่าข้อมูลและจำนวนข้อมูลที่จะนำมาใช้ทดสอบได้ ดังแสดงในรูปที่ 4.7 ส่วนในรูปที่ 4.8 คืออีกวิธีหนึ่งที่ผู้ใช้สามารถทำการสร้างรูปแบบของข้อมูลได้อย่างอิสระโดยกดปุ่มด้านซ้ายมือบนตัวเมาส์แล้วทำการลากลงในพื้นที่ของข้อมูล ในรูปที่ 4.8 จะเห็นว่าได้แสดงค่าของคู่ลำดับ (X,Y) ซึ่ง X หมายถึงตำแหน่งของข้อมูล และ Y หมายถึงค่าของข้อมูล จากคุณสมบัตินี้ทำให้ผู้ใช้สามารถสร้างรูปแบบของข้อมูลได้อย่างหลากหลายเพื่อทำการทดสอบการทำงานของแต่ละอัลกอริทึม



รูปที่ 4.7 การผลิตข้อมูลขาเข้าแบบสุ่ม



รูปที่ 4.8 การผลิตข้อมูลขาเข้าโดยผู้ใช้

หากผู้ใช้ยืนยันการสร้างข้อมูล ระบบจะทำการบันทึกข้อมูลลงในแฟ้มข้อมูลโดยผู้วิจัยได้กำหนดให้ชื่อของแฟ้มข้อมูลที่เก็บข้อมูลขาเข้านี้คือ "default.alg" สำหรับระบบการจัดเก็บแฟ้มข้อมูลขาเข้านี้ผู้วิจัยได้เลือกเป็นแฟ้มข้อมูลลำดับ (sequential file) เพื่อให้ง่ายต่อการใช้งานและผู้ใช้

สามารถเรียกแฟ้มข้อมูลนี้ขึ้นมาดูได้โดยโปรแกรมบรรณาธิการที่มีอยู่ทั่วไป ตัวอย่างของข้อมูลที่จัดเก็บในในแฟ้มข้อมูลขาเข้าดังแสดงได้ดังนี้คือ

1,1000,200	(ค่าข้อมูลต่ำสุด, ค่าข้อมูลสูงสุด, จำนวนชิ้นข้อมูล)
577	(ค่าของข้อมูลดัชนีที่ 0)
131	(„ „ „ 1)
159	(„ „ „ 2)
442	(„ „ „ 3)
178	(„ „ „ 4)
....	
....	
252	(„ „ „ 199)

หลังจากที่ผู้ใช้ได้ทำการสร้างข้อมูลขาเข้าแล้ว ภาคผลิตข้อมูลขาเข้านี้จะต้องแจ้งให้ภาคนำเสนอทราบ วิธีการนี้สามารถนำหลักการแลกเปลี่ยนข้อมูลแบบพลวัตระหว่างภาคผลิตข้อมูลขาเข้ากับภาคการนำเสนอซึ่งจะได้กล่าวในหัวข้อถัดไป

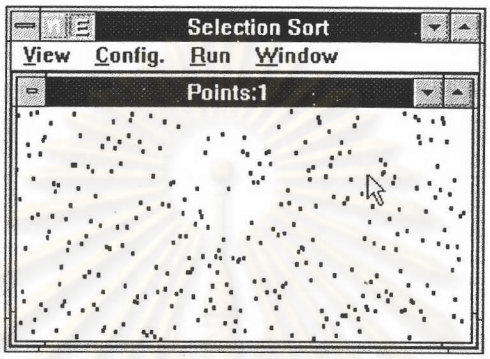
3. ภาคนำเสนอ

หลังจากที่ผู้ใช้ได้เลือกอัลกอริทึมที่ต้องการศึกษาและทำการระบุลักษณะของข้อมูลขาเข้าแล้ว ก็จะมาถึงสิ่งที่เป็นหัวใจของระบบจินตทัศน์อัลกอริทึมนั่นคือ การเลือกว่าจะให้ระบบแสดงภาพการทำงานของอัลกอริทึมในลักษณะใดในระบบจินตทัศน์อัลกอริทึมเราจะเรียกภาพการแสดงผลการทำงานนี้ว่า มุมมอง (view) โดยทั่วไปการทำงานของอัลกอริทึมหนึ่งๆจะสามารถแสดงได้หลายมุมมองแต่ผู้สร้างการจินตทัศน์จะต้องทำการวิเคราะห์ว่ามุมมองใดจะสามารถอธิบายการทำงานของอัลกอริทึมให้เกิดความเข้าใจได้ดีที่สุด สำหรับในระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลนี้ ผู้วิจัยได้เลือกที่จะทำการนำเสนอมุมมองสามชนิดด้วยกันคือ มุมมองแบบจุด มุมมองแบบแท่งและมุมมองแบบแถบสี โดยมีรายละเอียดดังนี้

3.1 มุมมองแบบจุด (Points View)

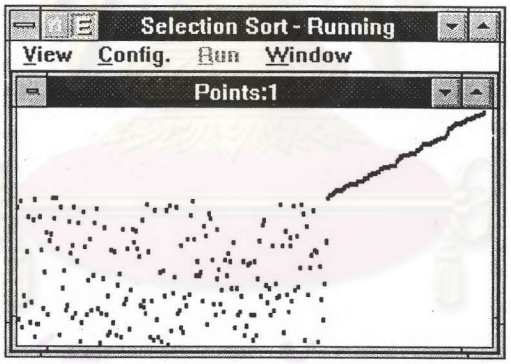
ในการจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลเราจะได้เห็นมุมมองประเภทนี้บ่อยครั้งที่สุดเนื่องจากสามารถอธิบายการทำงานของอัลกอริทึมให้เกิดความเข้าใจได้ง่ายประกอบกับนำมาใช้ได้กับทุกๆอัลกอริทึม ในการนำเสนอมุมมองแบบจุดนี้มักจะใช้เมื่อมีการสร้างข้อมูลขาเข้าเป็นจำนวนมาก ตัวอย่างการนำเสนอมุมมองแบบจุดได้แสดงไว้ในรูปที่ 4.9 จุดแต่ละจุดในมุม

มองจะเกิดจากคู่ลำดับ (x,y) โดยค่า x จะแทนตำแหน่งและ y จะแทนค่าของข้อมูลดังนั้นเมื่อมีการสลับตำแหน่งของข้อมูลก็จะมีผลทำให้จุดต่างๆเหล่านี้ในมุมมองมีการเปลี่ยนแปลงไปด้วย ในรูปที่ 4.10 แสดงการเปลี่ยนแปลงของมุมมองในขณะที่อัลกอริทึมการเรียงลำดับข้อมูลแบบเลือกกำลังทำงาน



หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

รูปที่ 4.9 มุมมองแบบจุดของข้อมูลที่มีลักษณะสุ่มก่อนการเรียงลำดับข้อมูล



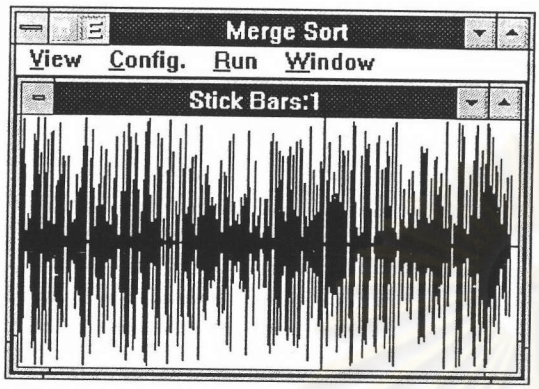
รูปที่ 4.10 มุมมองแสดงการเปลี่ยนแปลงขณะทำการเรียงลำดับข้อมูลแบบเลือก

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

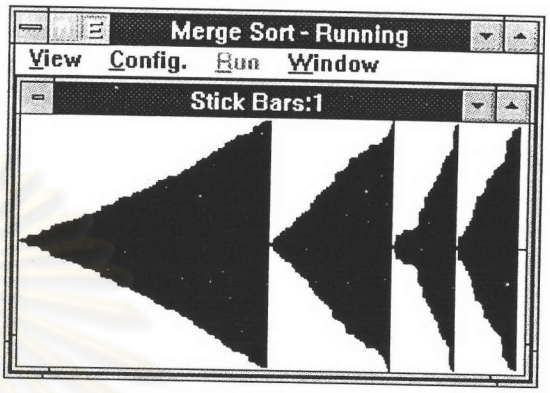
3.2 มุมมองแบบแท่ง (Stick Bars View)

มุมมองแบบแท่งเป็นอีกมุมมองหนึ่งที่นิยมใช้ในการนำเสนอรองลงมาจากมุมมองแบบจุดแต่มีข้อที่เด่นกว่าคือในการเรียงลำดับข้อมูลบางชนิดหากใช้มุมมองแบบแท่งจะทำให้การนำเสนอเข้าใจได้ง่ายกว่ามุมมองแบบจุดเช่น อัลกอริทึมการเรียงลำดับข้อมูลแบบผสาน ดังแสดงในรูปที่ 4.11 เป็นสถานะก่อนที่ระบบจะเริ่มทำการเรียงข้อมูล และขณะที่ระบบกำลังทำงานจะเกิดการเปลี่ยนแปลงภายในมุมมองดังแสดงในรูปที่ 4.12 ในมุมมองแบบแท่ง ความสูงของแต่ละแท่งจะแทนค่าของข้อมูลแท่งที่มีความสูงมากจะแทนข้อมูลที่มีค่ามากกว่าแท่งที่มีความสูงน้อย แท่งหนึ่ง

แท่งจะแทนค่าของข้อมูลหนึ่งค่าและตำแหน่งของแท่งในมุมมองก็จะแทนตำแหน่งของข้อมูลนั่นเอง



รูปที่ 4.11 มุมมองแบบแท่งของข้อมูลที่มีลักษณะสุ่มก่อนการเรียงลำดับ

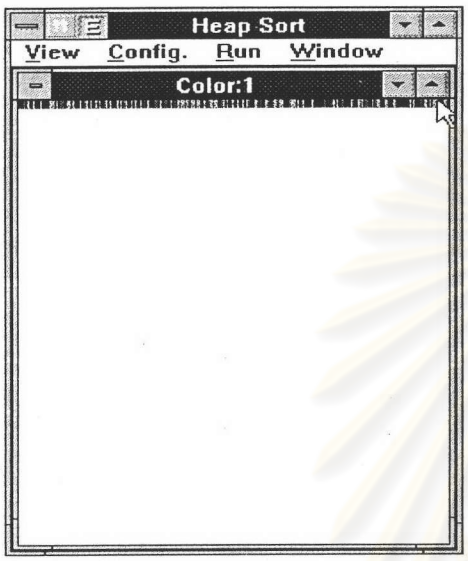


รูปที่ 4.12 มุมมองแสดงการเปลี่ยนแปลงขณะทำการเรียงลำดับข้อมูลแบบผสาน

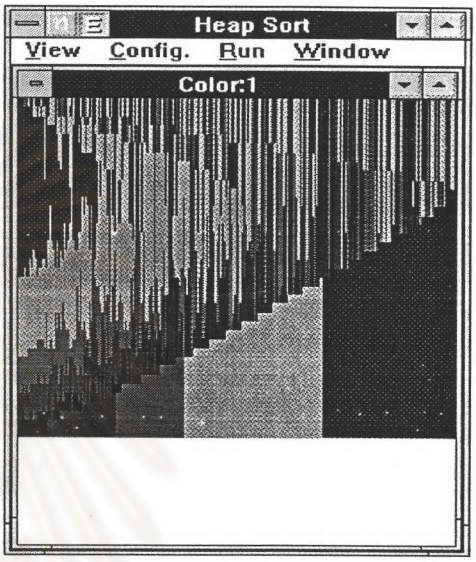
3.3 มุมมองแบบแถบสี (Color Band View)

จากการนำเสนอมุมมองสองวิธีที่ผ่านมาจะมีสิ่งหนึ่งที่เหมือนกันคือ เมื่อมีการสลับตำแหน่งของข้อมูลในอัลกอริทึมก็จะส่งผลให้มีการเปลี่ยนแปลงภาพในมุมมองแบบทันทีทันใดโดยผ่านโปรแกรมย่อยชื่อ OutputViewEvent (ดังที่กล่าวไว้ในภาคอัลกอริทึม) ซึ่งมีข้อดีคือจะเกิดเป็นภาพเคลื่อนไหว ผู้ศึกษาสามารถติดตามได้จนจบการทำงาน แต่หากต้องการดูการเปลี่ยนแปลงที่ผ่านมาขณะที่อัลกอริทึมกำลังทำงานจะไม่สามารถดูได้เลย จึงเกิดเป็นการนำเสนอมุมมองแบบใหม่ขึ้นมาโดยให้แสดงค่าของข้อมูลในแต่ละช่วงเวลาที่เราสนใจลงบนมุมมอง แต่ปัญหาคือทำอย่างไรจึงจะทำให้ผู้ศึกษาได้ทราบความแตกต่างระหว่างค่าข้อมูลได้ ในจุดนี้ Brown และ Hershberger (1992) ได้ร่วมกันทำวิจัยถึงเรื่องการนำสเปคตรัมของสีมาใช้แสดงภาพการเปลี่ยนแปลงในระบบจินตทัศน์อัลกอริทึม แต่สำหรับในโครงงานวิจัยนี้ผู้วิจัยได้จำกัดการแสดงสีเพียงแค่สามโทนเท่านั้นคือ โทนสีแดงแทนกลุ่มข้อมูลที่มีค่าน้อย โทนสีเขียวแทนกลุ่มข้อมูลที่มีค่าอยู่ในช่วงกลาง และโทนสีน้ำเงินแทนกลุ่มข้อมูลที่มีค่ามาก ในตอนแรกก่อนการทำงาน หากลักษณะของข้อมูลขาเข้าเป็นเลขสุ่ม เมื่อนำมาแสดงในมุมมองแบบแถบสีจะเห็นได้ว่าแถบสีจะออกมาในลักษณะกระจาย ดังแสดงในรูปที่ 4.13 และเมื่อข้อมูลผ่านการเรียงลำดับในแต่ละรอบภาพในมุมมองก็จะถูกระบายด้วยแถบสีใหม่โดยใช้ข้อมูลที่ได้มีการสลับตำแหน่งไปบ้างบางส่วน จนกระทั่งการเรียงลำดับข้อมูลเสร็จก็จะได้ผลการแสดงมุมมองดังรูปที่ 4.14 ซึ่งเป็นอัลกอริทึมการเรียงลำดับข้อมูลแบบฮีบ จะเห็นว่าแกนตั้งจะแทนเวลาที่เปลี่ยนแปลงไปและแกนนอนชุดของข้อมูลขาเข้าที่

ผ่านการเรียงลำดับในแต่ละช่วงเวลา ดังนั้นจะเห็นว่าความเปลี่ยนแปลงของข้อมูลที่เกิดขึ้นในระหว่างที่อัลกอริทึมทำการเรียงลำดับข้อมูลตั้งแต่ต้นจนจบได้ถูกบันทึกลงในมุมมองทั้งหมดซึ่งเป็นหัวใจหลักของการนำเสนอมุมมองแบบนี้นั่นเอง



รูปที่ 4.13 แสดงมุมมองแบบแถบสีก่อนการทำงาน



รูปที่ 4.14 แสดงมุมมองหลังจากจบการทำงาน

ในการนำเสนอมุมมองแบบแถบสีนี้จะมีความซับซ้อนกว่าการนำเสนอมุมมองแบบจุดและมุมมองแบบแท่ง คือ จากชุดของข้อมูลขาเข้าที่มีอยู่จะต้องนำมาผ่านการคำนวนและเปรียบเทียบช่วงของค่าข้อมูลว่าควรจะตกอยู่ในโทนของสีใด การแก้ไขปัญหานี้ผู้วิจัยได้อาศัยหลักการดังนี้คือ

ขั้นที่ 1 นำค่าสูงสุดของข้อมูลที่ผู้ใช้ได้ทำการตั้งค่าไว้มาแบ่งออกเป็น 3 ช่วง

ขั้นที่ 2 กำหนดให้ช่วงที่ 1 แทนโทนสีแดง ข้อมูลใดตกในช่วงนี้ก็จะให้แสดงสีแดง ช่วงที่ 2 แทนโทนสีเขียว และช่วงที่ 3 แทนโทนสีน้ำเงินตามลำดับ

ขั้นที่ 3 อ่านข้อมูลจากชุดข้อมูลขาเข้าทีละหนึ่งค่าแล้วทำการเปรียบเทียบว่าตกอยู่ในช่วงของโทนสีใดก็ให้ทำการวาดเส้นตั้งขนาดความสูง 20 หน่วย ความละเอียดของจอภาพลงในมุมมองด้วยโทนสีนั้น

จากขั้นตอนข้างต้นสามารถนำมาเขียนเป็นโปรแกรมย่อยภาษาวิชวลเบสิกเพื่อทำการวาดแถบสีลงในมุมมองได้ดังนี้คือ

```
Sub DrawColor(frmView As Form)
  frmView.Cls
  frmView.DrawWidth = dWidth
  frmView.BackColor = QBColor(15)
  frmView.ForeColor = QBColor(15)
1)  cFraction = Int(UpBound / 3)
  maxRED = cFraction
2)  maxGREEN = (cFraction * 2)
  maxBLUE = (cFraction * 3)
  frmView.Scale (0, UpBound) - ((NumData - 1), 0)
  For i = 0 To (NumData - 1)
    If item(i) <= maxRED Then
3)    frmView.Line(i,ColorLine)-
                  (i,ColorLine- 20),
                  RGB(item(i),0,0)
    ElseIf item(i) <= maxGREEN Then
      frmView.Line(i,ColorLine)-
                  (i,ColorLine- 20),
                  RGB(0,item(i),0)
    ElseIf item(i) <= maxBLUE Then
      frmView.Line(i,ColorLine)-
                  (i,ColorLine- 20),
                  RGB(0,0,item(i))
    EndIf
    DoEvents
  Next i
End Sub
```

ผลการทำงานของโปรแกรมย่อยข้างต้นก็คือแถบสีต่างๆที่ปรากฏในมุมมองดังที่ได้แสดงผ่านมาแล้วในรูปที่ 4.13 อย่างไรก็ตามผู้วิจัยยังพบอีกว่าในการแสดงมุมมองแบบแถบสีนี้ยังคงมีปัญหาที่สำคัญอีกประการหนึ่งคือ ทำอย่างไรจึงจะทำให้การวาดแถบสีลงในมุมมองขณะที่อัลกอริทึมกำลังทำงานอยู่โดยไม่ให้ล้นออกมานอกมุมมอง จากการศึกษาพบว่า เราสามารถหาค่าโดยประมาณของจำนวนครั้งของการเปรียบเทียบในแต่ละอัลกอริทึม (Sedgewick,1990) ตัวอย่างเช่น จำนวนครั้งของการเปรียบเทียบสำหรับการเรียงลำดับข้อมูลแบบเลือกจะมีประมาณ (N*N)/2 เมื่อ N แทนจำนวนข้อมูลที่จะทำการเรียงลำดับ (Sedgewick,1990) จากคุณสมบัตินี้เองที่สามารถนำมาใช้ในการควบคุมการวาดแถบสีลงในมุมมองเพื่อไม่ให้เกินขอบเขตโดยใช้หลักการดังนี้คือ

ขั้นที่ 1 ประมาณค่าจำนวนครั้งของการเปรียบเทียบของอัลกอริทึมที่จะทำการจินตทัศน์

ขั้นที่ 2 นำค่าความสูงของมุมมองตั้งแล้วหารด้วยค่าที่ได้จากขั้นที่ 1

ขั้นที่ 3 เมื่อมีการเปรียบเทียบค่าของข้อมูลก็ให้นับจำนวนครั้งเอาไว้

ขั้นที่ 4 เมื่อมีการสลับตำแหน่งของข้อมูลและจะต้องทำการเปลี่ยนแปลงภาพในมุมมองก็ให้ตรวจสอบก่อน ถ้าจำนวนครั้งของการเปรียบเทียบมีค่ามากกว่าหรือเท่ากับค่าที่ได้จากขั้นตอนที่ 2 ก็ให้ทำการเปลี่ยนแปลงภาพในมุมมองแบบแถบสี แล้วเริ่มต้นนับหนึ่งใหม่ ทำเช่นนี้เรื่อยไปจนจบการทำงานของอัลกอริทึมจึงทำการเปลี่ยนแปลงภาพในมุมมองอีกครั้งหนึ่ง

จากขั้นตอนข้างต้นสามารถนำมาเขียนเป็นโปรแกรมย่อยภาษาวิชวลเบสิกได้ดังนี้คือ

```
Sub mnuRunStart_Click ()
  Const E = 2.718282
  If SELECTION_SORT Then
ขั้นที่ 1        nTheory = (idx * idx) / 2
ขั้นที่ 2        nChangeColor = (nTheory) / (UpBound / 20)
   ElseIf INSERTION_SORT Then
       nTheory = (idx * idx) / 4
       nChangeColor = (nTheory) / (UpBound / 20)
            .........
            .........
            .........
   ElseIf MERGE_SORT Then
       nTheory = (idx) * (Log(idx))
       nChangeColor = (nTheory) / (UpBound / 20)
  EndIf
End Sub
```

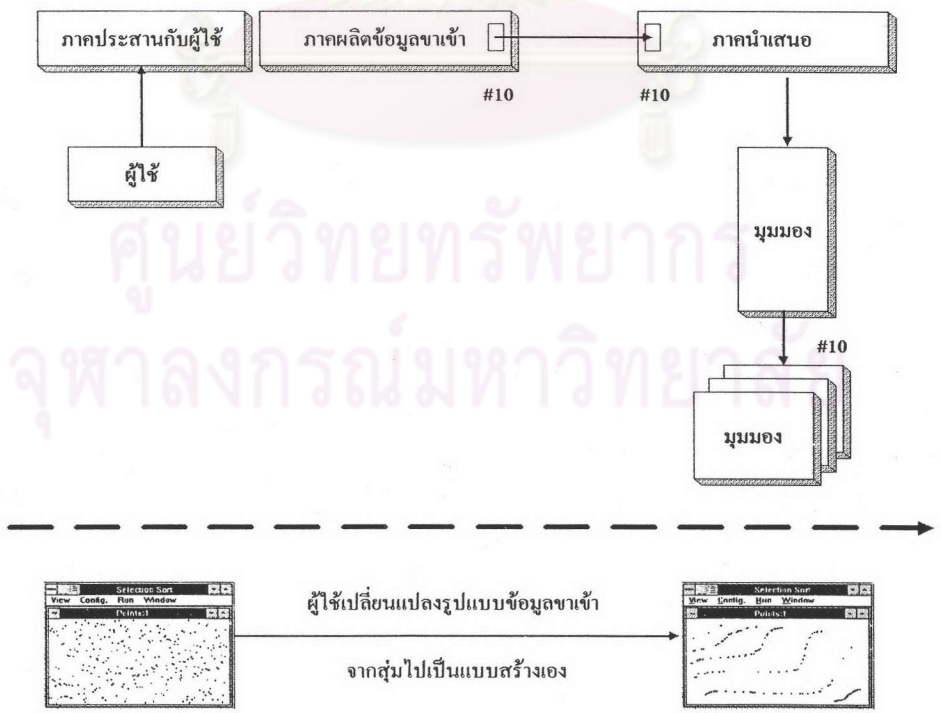
และในอัลกอริทึมการเรียงลำดับข้อมูลทุกๆอัลกอริทึมจะต้องมีการใส่ชุดคำสั่งเพื่อคอยตรวจการวาดแถบสีลงในมุมมอง ตัวอย่างอัลกอริทึมการเรียงลำดับข้อมูลแบบเลือกที่ได้รับการใส่ชุดคำสั่งลงไปแล้วจะมีลักษณะดังนี้

```
Sub SelectionSort_ASC( )
  ..........
  For idx = n TO 0 Step - 1
       ..........
       ..........
            If item(jdx) > item(max) Then
                 nCompare = nCompare + 1
ขั้นที่ 3            cChg = cChg + 1
                 i = NotifySync( )
                 max = jdx
            Else
                 nCompare = nCompare + 1
ขั้นที่ 3            Chg = cChg + 1
```

I10716450

```
                    i = NotifySync( )
                End If
            ..........
            ..........
ขั้นที่ 4       If cChg >= nChangeColor Then
                    ColorViewEvent
                    cChg = 0
            .   End If
        Next idx
ขั้นที่ 4   ColorViewEvent
    End Sub
```

ดังที่กล่าวไว้ในตอนท้ายของภาคผลิตข้อมูลขาเข้าว่า ภาคผลิตข้อมูลนำเข้านี้จะมีการสื่อสารกับภาคนำเสนอด้วยซึ่งจะขออธิบายดังนี้คือ เมื่อผู้ใช้ได้ทำการเลือกอัลกอริทึม สร้างข้อมูลขาเข้าและได้เลือกมุมมองที่จะนำเสนอแล้วหากผู้ใช้ต้องการเปลี่ยนรูปแบบของข้อมูลขาเข้าก็สามารถทำได้แต่ภาคผลิตข้อมูลขาเข้าจะต้องมีแจ้งให้ภาคนำเสนอได้ทราบว่าในขณะนี้ข้อมูลได้ถูกเปลี่ยนไป ภาคผลิตข้อมูลขาเข้าจะทำการใส่ค่าสตริง #10 ลงไปในชิ้นข้อมูล และด้วยคุณสมบัติการแลกเปลี่ยนข้อมูลแบบพลวัตภาคนำเสนอจะมองเห็นค่าสตริง #10 นั้นด้วยซึ่งจากการที่ได้กำหนดมาตรฐานในการสนทนาไว้แล้วจึงแปลความหมายได้ว่าให้ภาคนำเสนอทำการอ่านข้อมูลจากแฟ้มข้อมูลชื่อ "default.alg" ขึ้นมาและทำการสร้างภาพในมุมมองต่างๆด้วยข้อมูลชุดใหม่โดยอัตโนมัติ ขบวนการทำงานดังกล่าวสามารถแสดงได้ดังรูปที่ 4.15



รูปที่ 4.15 แสดงการเปลี่ยนแปลงมุมมองเมื่อเกิดการแลกเปลี่ยนข้อมูลแบบพลวัต

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ภาคประสานจังหวะ

หน้าที่ของภาคประสานจังหวะการทำงานในระบบจินตทัศน์อัลกอริทึมเปรียบได้กับไวทยากรที่มีหน้าที่ควบคุมการบรรเลงในวงดนตรีให้เป็นไปตามจังหวะและท่วงทำนองที่สอดคล้องกัน นักดนตรีจะคอยสังเกตการให้สัญญาณควบคุมวงจากไวทยากร เมื่อใดที่ได้รับสัญญาณให้ทำการบรรเลงในโนตของตนเองจึงจะทำการบรรเลงได้ หากเราจะเปรียบเทียบไวทยากรคือภาคประสานจังหวะ ดังนั้นนักดนตรีที่ทำการบรรเลงเครื่องดนตรีของตนก็คืออัลกอริทึมการเรียงลำดับข้อมูลที่ผู้ใช้ได้ทำการเรียกขึ้นมาจินตทัศน์ซึ่งจำเป็นต้องมีการประสานจังหวะการทำงานซึ่งกันและกันนั่นเอง สำหรับโครงงานวิจัยนี้มีการเรียกใช้ฟังก์ชันการประสานจังหวะโดยผ่านทางคลังโปรแกรมเชื่อมโยงแบบพลวัตเพื่อทำให้ระบบมีลักษณะเป็นโครงสร้างและสามารถจัดการหน่วยความจำได้อย่างมีประสิทธิภาพ คลังโปรแกรมที่เก็บฟังก์ชันต่างๆสำหรับการประสานจังหวะนี้มีชื่อว่า "SYNCLIB.DLL" ภายในจะประกอบด้วยฟังก์ชันต่างๆซึ่งสามารถจัดกลุ่มตามหน้าที่การทำงานได้เป็น 2 กลุ่มดังนี้คือ ฟังก์ชันสำหรับผู้ควบคุมหรือผู้ให้สัญญาณ และ ฟังก์ชันสำหรับผู้ถูกควบคุมหรือผู้รับสัญญาณ โดยมีรายละเอียดดังนี้

1. ผู้ควบคุมหรือผู้ให้สัญญาณจังหวะ ประกอบด้วยฟังก์ชันย่อยดังนี้

1.1 การแจ้งให้ผู้ถูกควบคุมทราบว่าจะมีการให้จังหวะเป็นอย่างไร วิธีการเรียกใช้ฟังก์ชันจะต้องมีการประกาศก่อนคือ

```
Declare Sub ConfigSync Lib "synclib.dll"
          (ByVal SyncSize As Integer,
          ByVal noDelay As Integer,
          ByVal runfree As Integer)
```

ความหมายของพารามิเตอร์ต่างๆ

SyncSize คือ จำนวนอัลกอริทึมที่ผู้ควบคุมสามารถส่งสัญญาณให้มีค่าได้ไม่เกิน 20 อัลกอริทึม

noDelay „ คือตัวแปรที่ใช้ระบุช่วงการหน่วงเวลาของอัลกอริทึม โดย

0 = ไม่มีการหน่วงเวลา

L = มีการหน่วงเวลาเท่ากับ L มิลลิวินาทีก่อนจะให้สัญญาณครั้งต่อไป

runfree „ 0 หมายถึง หยุดการให้สัญญาณชั่วคราว

1 หมายถึงการให้สัญญาณต่อ

1.2 การส่งสัญญาณการประสานจังหวะให้ผู้ถูกควบคุม มีการประกาศดังนี้

```
Declare Function NextSync Lib "synclib.dll" () As Integer
```

1.3 การบอกยกเลิกการให้บริการการประสานจังหวะ มีการประกาศดังนี้

```
Declare Sub DropSync Lib "synclib.dll" ()
```

พารามิเตอร์แต่ละตัวในฟังก์ชันต่างๆหากมีค่าเป็น -1 จะทำให้ฟังก์ชันไม่สนใจพารามิเตอร์นั้น การนำฟังก์ชันต่างๆของผู้ควบคุมมาใช้สามารถแสดงได้ดังนี้

```
Sub mnuRunStart_Click ( )
        Call ConfigSync(1, 0, -1)
        i = NextSync()
End Sub
```

2. ผู้ถูกควบคุมหรือผู้รับสัญญาณจังหวะ ประกอบด้วยฟังก์ชันย่อยดังนี้

2.1 การขอลงทะเบียนเพื่อรับสัญญาณการประสานจังหวะ มีการประกาศดังนี้

```
Declare Function RegisterSync Lib "synclib.dll" ()
                As Integer
```

ผลของการเรียกใช้ฟังก์ชันนี้จะได้ส่งตัวเลขจำนวนกลับมาเพื่อใช้ระบุเป็นเลขประจำตัวของโพรเซส

2.2 การรับสัญญาณการประสานจังหวะ ในส่วนนี้จะขออธิบายเพิ่มเติมดังนี้ ในการวัดประสิทธิภาพการทำงานของอัลกอริทึมแต่ละชนิดมักจะใช้จำนวนครั้งของการดำเนินการขั้นพื้นฐาน (สมชาย ประสิทธิ์จูตระกูล,1993) เป็นตัววัด ดังนั้นอัลกอริทึมใดที่มีการดำเนินการขั้นพื้นฐานน้อยกว่าอีกอัลกอริทึมหนึ่งย่อมแสดงได้ว่าอัลกอริทึมนั้นมีประสิทธิภาพมากกว่าสำหรับปัญหาในเรื่องการเรียงลำดับข้อมูลจะใช้การเปรียบเทียบค่าของข้อมูลเป็นการดำเนินการขั้นพื้นฐาน (Sedgewick,1990) ดังนั้นเพื่อให้เกิดความเสมอภาคในการวัดประสิทธิภาพ อัลกอริทึมใดได้ทำงานมาจนถึงการดำเนินการขั้นพื้นฐานจะต้องหยุดการทำงานชั่วคราวเพื่อให้โอกาสแก่อัลกอริทึมอื่นๆได้ทำงานจนถึงการดำเนินการขั้นพื้นฐานของตนเองจึงหยุดเช่นกันการทำงานจึงออกมาในลักษณะสลับไปมาจนเสร็จครบทุกอัลกอริทึม ในการเรียกใช้ฟังก์ชันการรอรับสัญญาณจะต้องมีการประกาศดังนี้

```
Declare Function NotifySync Lib "synclib.dll"
    ( Byval hWnd As Integer, ByVal procnum As Integer )
              As Integer
```

ความหมายของพารามิเตอร์ต่างๆ

hWnd เป็นค่าที่ใช้ระบุหมายเลขของโพรเซสในไมโครซอฟต์วินโดวส์

procnum เป็นเลขจำนวนเต็มที่ได้จากการส่งค่ากลับของฟังก์ชัน RegisterSync เพื่อใช้กำกับเลขประจำตังของโพรเซส

2..3 การบอกยกเลิกการรับบริการการประสานจังหวะ ฟังก์ชันนี้จะมีการเรียกใช้เมื่ออัลกอริทึมได้ทำงานเสร็จแล้วจึงต้องมีการขอยกเลิกการรับบริการ ซึ่งจะมีผลทำให้อัลกอริทึมที่เหลืออยู่ทำงานเร็วขึ้น เนื่องจากจำนวนสมาชิกที่คอยรอรับบริการการประสานจังหวะมีน้อยลง ในการใช้ฟังก์ชันนี้มีวิธีการประกาศดังนี้คือ

```
Declare Function DropSync Lib "synclib.dll"
        ( ByVal procnum As Integer ) As Integer
```

โดยพารามิเตอร์ procnum คือ หมายเลขประจำตัวโพรเซส ที่ได้จากขั้นตอนการเรียกใช้ฟังก์ชัน RegisterSync สำหรับตัวอย่างการใช้ฟังก์ชันของผู้ถูกควบคุมในอัลกอริทึมการเรียงลำดับข้อมูลแบบเลือกแสดงได้ดังรูปที่ 4.16

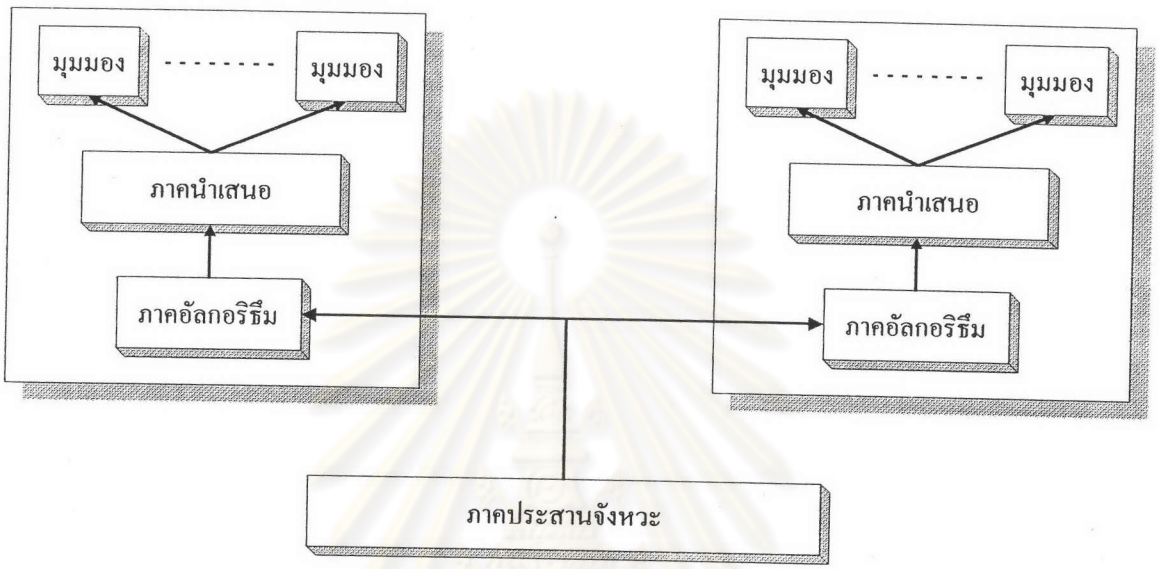
```
Sub Run_Start()
  procnum = RegisterSync()
  i = NotifySync(ViewMDI.hWnd, procnum)
  ...
  ...  'Start Selection Sort Process'
  ...
  i = DropSync(procnum)

End Sub

Sub SelectionSort_ASC( )
  ...
  ...
  ...
  If item(jdx) > item(max) Then
       nCompare = nCompare + 1
       i = NotifySync( )
  Else
       nCompare = nCompare + 1
       i = NotifySync( )
  End If
  ...
  ...
  ...
End Sub
```

รูปที่ 4.16 ตัวอย่างการใช้ฟังก์ชันการขอรับบริการการประสานจังหวะ

จากรายละเอียดต่างๆดังที่ได้กล่าวมาจะเห็นว่าการทำงานของภาคประสานจังหวะจะมีส่วนสัมพันธ์กับภาคอื่นๆในระบบจินตทัศน์อัลกอริทึม ซึ่งความสัมพันธ์นี้สามารถแสดงได้ดังรูปที่ 4.17



รูปที่ 4.17 ภาคประสานจังหวะกับการนำเสนอ

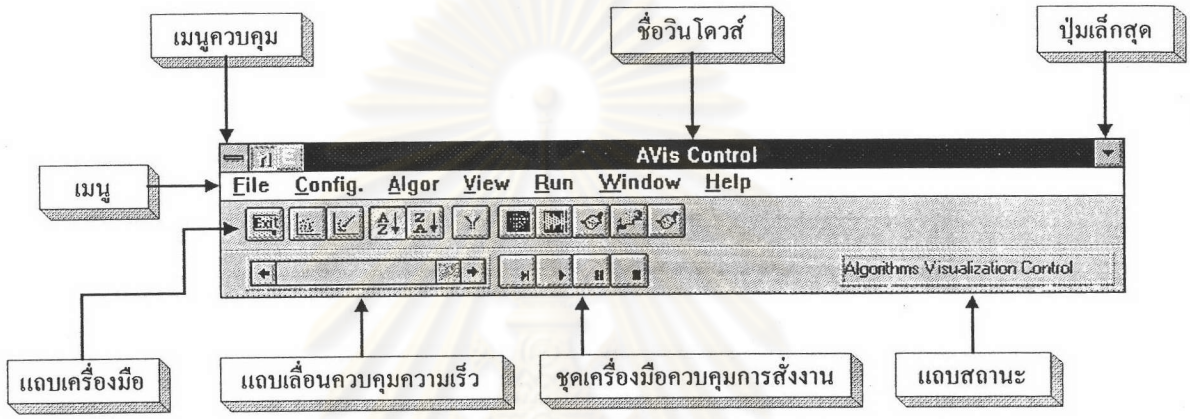
<u>ระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลภายใต้สภาพปฏิบัติการไมโครซอฟต์วินโดวส์</u>

จากที่ผ่านมาเราได้ทราบแล้วว่าไมโครซอฟต์วินโดวส์เป็นสภาพปฏิบัติการที่ทำให้ผู้ใช้เกิดความสะดวกและง่ายในการใช้งานเนื่องจากมีการติดต่อกับผู้ใช้แบบกราฟิก (Graphical User Interface : GUI) ผู้ใช้ไม่จำเป็นจะต้องจำคำสั่งที่มีไวยกรณ์เฉพาะเช่นเดียวกับระบบปฏิบัติการเอมเอสดอส (MS - DOS) และนอกจากนั้นโปรแกรมประยุกต์ที่พัฒนาภายใต้ไมโครซอฟต์วินโดวส์มักจะมีการติดต่อกับผู้ใช้เป็นมาตรฐานเดียวกันหากผู้ใช้ได้ทำการศึกษาโปรแกรมหนึ่งๆ ได้แล้วก็จะใช้เวลาศึกษาโปรแกรมต่อไปได้ไม่ยากนัก ในระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลนี้ผู้วิจัยก็ได้ทำการพัฒนาโดยให้มีรูปแบบการติดต่อกับผู้ใช้เป็นไปตามมาตรฐานเช่นกัน

นอกเหนือจากการติดต่อกับผู้ใช้แบบกราฟิกแล้วไมโครซอฟต์วินโดวส์ยังมีคุณสมบัติอื่นที่เอื้ออำนวยต่อการพัฒนาเช่น การทำงานแบบหลายภารกิจ การแสดงมุมมองแบบหลายเอกสาร และการใช้คลังโปรแกรมเชื่อมโยงแบบพลวัต เป็นต้น รายละเอียดในส่วนต่าง ๆ มีดังนี้คือ

1. การประสานกับผู้ใช้แบบกราฟิก (Graphical User Interface : GUI)

ดังที่ทราบแล้วว่าไมโครซอฟต์วินโดวส์ได้ใช้รูปภาพในการสื่อความหมาย เช่น สัญรูป (icon) แถบเลื่อน (scroll bar) และเมนู (menu) เป็นต้น ในโครงงานวิจัยก็ได้นำสิ่งเหล่านี้มาใช้ ลักษณะโปรแกรมระบบจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูลเมื่อถูกเรียกใช้งานจะแสดงดังรูปที่ 4.18



รูปที่ 4.18 โปรแกรมจินตทัศน์อัลกอริทึมการเรียงลำดับข้อมูล

โดยมีรายละเอียดดังนี้

1.1 เมนูควบคุม หากกดปุ่มนี้จะแสดงรายการคำสั่งสำหรับควบคุมวินโดวส์ซึ่งประกอบด้วยคำสั่ง ลดขนาดเป็นสัญรูป คำสั่งเคลื่อนย้ายวินโดวส์ คำสั่งปิดวินโดวส์ ซึ่งก็คือ คำสั่งเลิกการทำงานของโปรแกรมนั่นเองและคำสั่งสำหรับสวิช (switch to) ไปทำงานอื่น

1.2 ปุ่มเล็กสุด เมื่อกดปุ่มนี้จะลดขนาดของวินโดวส์ให้มีขนาดเล็กสุดโดยจะแสดงในรูปของสัญรูป

1.3 เมนู ประกอบด้วยรายการคำสั่งต่างๆโดยจะเรียงตามแนวนอนด้านบนของวินโดวส์ ในแต่ละหัวข้อของเมนูจะมีโครงสร้างดังต่อไปนี้

1.4 แถบเครื่องมือ เป็นปุ่มควบคุมที่ช่วยอำนวยความสะดวกในการใช้งาน แถบเครื่องมือจะมีฟังก์ชันการทำงานเหมือนกับรายการคำสั่งในเมนูแต่จะแสดงด้วยภาพแทน ผู้ใช้สามารถเลือกคำสั่งได้อย่างรวดเร็วโดยเลื่อนเมาส์ไปที่ปุ่มควบคุมที่มีรูปภาพกำกับอยู่และทำการกดปุ่มเมาส์ด้านซ้าย ปุ่มควบคุมต่างๆในแถบเครื่องมือมีความหมายดังนี้

หมายถึง การออกจากโปรแกรมระบบจินตทัศน์อัลกอริทึม

„ การสร้างข้อมูลขาเข้าแบบตัวเลขสุ่ม

„ การสร้างข้อมูลขาเข้าโดยผู้ใช้สร้างเอง

„ การเรียงลำดับข้อมูลจากน้อยไปมาก

„ การเรียงลำดับข้อมูลจากมากไปน้อย

„ การเลือกประเภทของอัลกอริทึมมาทำงาน

„ การเลือกนำเสนอมุมมองแบบจุด

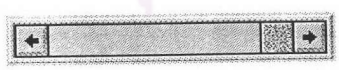
„ การเลือกนำเสนอมุมมองแบบแท่ง

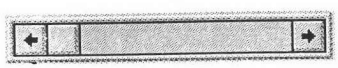
„ การเลือกนำเสนอมุมมองแบบแถบสี

„ การเลือกแสดงจำนวนครั้งของการเปรียบเทียบในแต่ละอัลกอริทึม

„ การย้อนสถานะของมุมมองแต่ละมุมมองกลับไปที่จุดตั้งต้น

1.5 แถบเลื่อนควบคุมความเร็ว เป็นส่วนที่ใช้ในการควบคุมความเร็วในการทำงานและการนำเสนอการเปลี่ยนแปลงของภาพในมุมมองต่างๆได้ตามที่ผู้ใช้ต้องการ เนื่องจากในบางอัลกอริทึมเช่นการเรียงลำดับข้อมูลแบบเร็ว ผู้ใช้อาจจะติดตามการเปลี่ยนแปลงของภาพในมุมมองไม่ทันก็สามารถทำการหน่วงการทำงานให้ช้าลงได้ ในแถบเลื่อนควบคุมความเร็วนี้จะมีตัวเลื่อนอยู่ภายในซึ่งผู้วิจัยได้ทำการออกแบบตามความรู้สึกของผู้ใช้คือ หากตัวเลื่อนอยู่ทางขวามือสุดแสดงว่าอัลกอริทึมจะทำงานในภาวะที่เร็วที่สุดตามความเร็วของซีพียูแต่หากตัวเลื่อนอยู่ทางซ้ายมือสุดแสดงว่าอัลกอริทึมได้รับการหน่วงให้ทำงานช้าลง วิธีการใช้งานแถบเลื่อนสามารถทำได้โดยการลากเมาส์มาที่ลูกศรทางซ้ายหรือทางขวาเพื่อบอกทิศทางในการเลื่อนจากนั้นก็กดปุ่มเมาส์ด้านซ้ายตัวเลื่อนจะเลื่อนไปตามทิศทางของลูกศรทันที


หมายถึง การทำงานของอัลกอริทึมในสถานะเร็วที่สุด


„ การทำงานของอัลกอริทึมในสถานะช้าที่สุด

รูปที่ 4.19 แสดงสถานะของแถบเลื่อนควบคุมความเร็ว

1.6 ชุดเครื่องมือควบคุมการสั่งงาน เป็นชุดที่ใช้ควบคุมการทำงานอัลกอริทึมโดยมีความหมายดังนี้

หมายถึง ควบคุมให้อัลกอริทึมทำงานทีละขั้นตอนเมื่อกดปุ่มควบคุมนี้หนึ่งครั้ง

„ การสั่งให้อัลกอริทึมเริ่มทำงาน

„ การสั่งให้อัลกอริทึมหยุดทำงานชั่วคราว

„ การยกเลิกการควบคุมการทำงาน

1.7 แถบสถานะ เป็นส่วนที่จะแสดงข้อความสั้นๆสำหรับอธิบายความหมายโดยย่อของแถบเครื่องมือต่างๆ ข้อมูลในแถบสถานะจะเปลี่ยนแปลงเมื่อผู้ใช้ลากเมาส์ไปอยู่เหนือปุ่มควบคุมต่างๆ

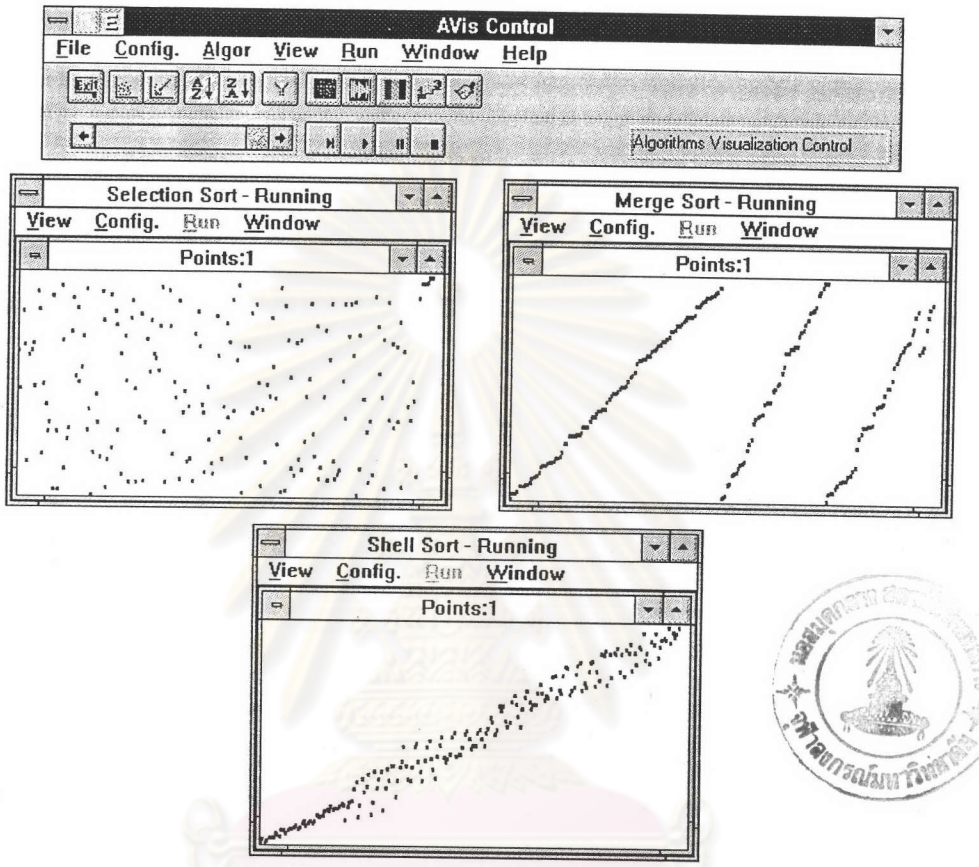
2. การทำงานแบบหลายภารกิจ (Multitasking)

ถึงแม้ไมโครซอฟต์วินโดวส์จะเป็นสภาพปฏิบัติการที่ทำงานได้หลายภารกิจแต่ในความเป็นจริงแล้ววินโดวส์มีลักษณะของ non - preemtive multitasking นั่นคือผู้พัฒนาโปรแกรมจะต้องทำการเขียนโปรแกรมในลักษณะที่ไม่ทำให้เกิดการจองทรัพยากรของเครื่องคอมพิวเตอร์ เช่น ซีพียู คีย์บอร์ด แต่เพียงผู้เดียวซึ่งในจุดนี้จะต้องมีความระมัดระวังเป็นอย่างยิ่ง มิฉะนั้นวินโดวส์ก็จะไม่ต่างอะไรกับระบบปฏิบัติการเอมเอสดอสที่ทำงานได้เพียงงานเดียว ณ เวลาใดเวลาหนึ่งนั่นเอง จากรูปที่ 4.20 แสดงอัลกอริทึมการวาดจุดลงบนมุมมองซึ่งจะเห็นว่ามีการทำงานแบบวงวนซึ่งเป็นช่วงที่โปรแกรมใช้ทรัพยากรของซีพียูตลอดเวลา ดังนั้นในตัวแปลภาษาวิชวลเบสิกได้ให้ฟังก์ชันพิเศษชื่อ DoEvents ซึ่งมีหน้าที่สำหรับหยุดการทำงานของโปรแกรมชั่วคราวและให้วินโดวส์กลับไปสนใจโปรแกรมอื่นๆที่ผู้ใช้ได้โหลด (load) เข้ามาในหน่วยความจำและทำงานกับโปรแกรมดังกล่าวจนหมดชิ้นเวลา (time slice) จากนั้นจึงกลับไปทำงานในคำสั่งถัดไปจากฟังก์ชัน DoEvents ซึ่งหากไม่ทำเช่นนี้จะทำให้วินโดวส์ไม่สามารถไปทำงานอื่นๆได้เลยจนกว่าจะทำการสร้างจุดลงในมุมมองจนเสร็จ

```
Sub DrawPoints(frmView As Form)
   ..........
   ..........
   idx= Ubound (unit)
   screen.MousePointer = 12
   for i = 0 To idx
        frmView.Pset (i, unit (i) )
        DoEvents
   Next i
End Sub
```

รูปที่ 4.20 แสดงการเขียนโปรแกรมแบบหลายภารกิจในวินโดวส์

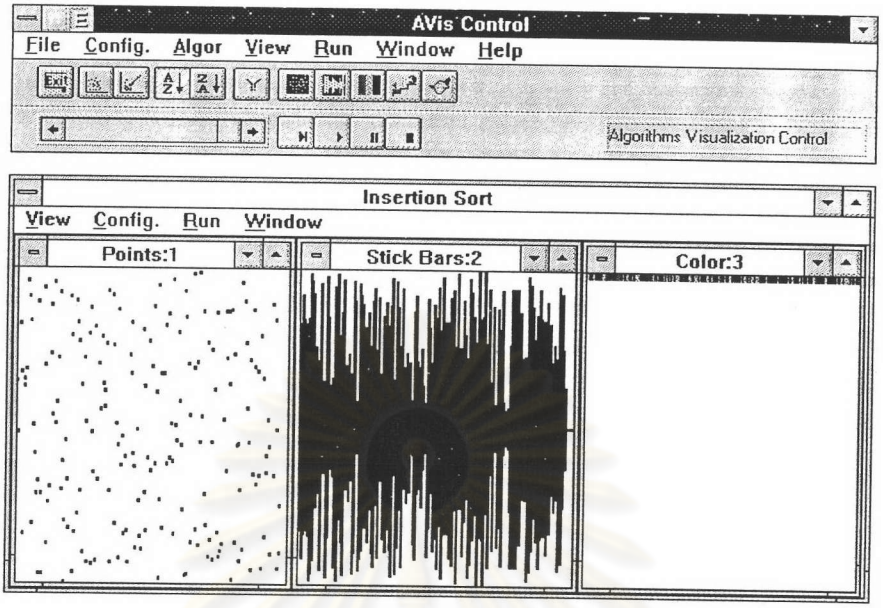
แต่อย่างไรก็ตามด้วยการทำงานแบบหลายภารกิจในวินโดวส์ทำให้ผู้ใช้สามารถดูการทำงานของหลายๆอัลกอริทึมได้ในเวลาเดียวกัน ดังแสดงในรูปที่ 4.21



รูปที่ 4.21 การทำงานแบบหลายภารกิจภายใต้สภาพปฏิบัติการไมโครซอฟต์วินโดวส์

3. การประสานแบบหลายเอกสาร (Multiple Document Interface :MDI)

เป็นคุณสมบัติอันหนึ่งที่เอื้ออำนวยต่อการแสดงและจัดการวินโดวส์ย่อยหลายๆ วินโดวส์ที่อยู่ภายใต้วินโดวส์หลัก ดังที่ได้กล่าวไว้ในขอบเขตของการวิจัยว่าระบบจินตทัศน์ที่จะพัฒนานี้จะต้องสามารถนำเสนอมุมมองได้หลายมุมมองพร้อมๆกัน ดังนั้นผู้วิจัยจึงได้นำคุณสมบัตินี้มาใช้ในการออกแบบดังแสดงในรูปที่ 4.22 จะเห็นว่าผู้ใช้สามารถเลือกการแสดงมุมได้ตั้งแต่หนึ่งถึงสามมุมมอง ไม่ซ้ำกันภายใต้ อัลกอริทึมเดียวกันได้



รูปที่ 4.22 การแสดงมุมมองแบบหลายเอกสาร

4. การใช้คลังโปรแกรมเชื่อมโยงแบบพลวัต (Dynamic Link Library : DLL)

คือคลังโปรแกรมที่เก็บไว้ในแฟ้มข้อมูลที่เป็นอิสระจากโปรแกรมที่เรียกใช้ เมื่อมีการเรียกใช้ฟังก์ชันในคลังโปรแกรมก็จะมีการนำคลังโปรแกรมดังกล่าวเข้าสู่หน่วยความจำและเชื่อมโยงกับโปรแกรมที่เรียกใช้ หากมีหลายๆโปรแกรมเรียกใช้ฟังก์ชันเดียวกันฟังก์ชันที่ถูกเรียกจะใช้เนื้อที่ในหน่วยความจำเพียงชุดเดียวอีกทั้งหากมีการเปลี่ยนคลังโปรแกรมก็ไม่จำเป็นต้องนำโปรแกรมที่เรียกใช้มาทำการแปล (compile) เพื่อให้ได้รหัสจุดหมาย (object code) ใหม่

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย