บทที่ 3

การออกแบบขั้นตอนวิธีและโครงสร้างข้อมูล

ในการวิเคราะห์และพัฒนาขั้นตอนวิธี (Algorithm) จะได้กล่าวถึงรายละเอียดต่าง ๆ ของการทำงาน รวมทั้งข้อจำกัดและเงื่อนไขที่จำเป็นต้องคำนึงถึง นอกจากนี้ก็จะพิจารณาถึงลักษณะข้อมูลที่เกี่ยวข้องในแต่ละขั้นตอน ซึ่งจะเป็นแนวทางไปสู่ข้อสรุปเกี่ยวกับการพัฒนาระบบข้อมูลในภายหลังด้วย การจัดลำดับหัวข้อในการวิเคราะห์แอลกอริทึมที่จะกล่าวถึงในบทนี้ ไม่ได้จัดตามลำดับการทำงานจริงทั้งหมด แต่จัดไว้ในลักษณะที่ง่ายต่อการทำความเข้าใจกับปัญหาต่าง ๆ ที่เกี่ยวข้องกันแต่ละขั้นตอน

สำหรับการออกแบบและพัฒนาขั้นตอนวิธีนั้น ได้พยายามให้มีรูปแบบทั่วไป (general) โดยไม่อิงกับอุปกรณ์ทางฮาร์ดแวร์ และซอฟท์แวร์ใด ๆ โดยเฉพาะ เพื่อให้เป็นพื้นฐานในการนำไปพัฒนาระบบงานจริงได้โดยสะดวก การแสดงข้อสรุปของขั้นตอนวิธีต่าง ๆ ในบทนี้นั้น นอกเหนือจากการใช้ลักษณะของการบรรยายแล้ว ยังได้แสดงไว้ในลักษณะของผังงาน (Flow Chart) และขั้นตอนวิธีในรูปแบบรหัสเทียม (Pseudo Code) ด้วย เนื่องจากการทำงานของขั้นตอนส่วนใหญ่นั้นวางรากฐานอยู่ที่สูตรการคำนวณต่าง ๆ ขั้นตอนวิธีในรูปแบบรหัสเทียมจะให้รายละเอียดของการทำงานได้ชัดเจน รหัสเทียมที่ใช้ในวิทยานิพนธ์ฉบับนี้ ได้ใช้รูปแบบไวยากรณ์และคำสำคัญ (Key word) ที่คล้ายคลึงกับภาษา PL/1

3.1 ขั้นตอนทั่วไปของระบบงาน

แม้ในทางปฏิบัติระบบงานสร้างทัศนียภาพจะประกอบด้วยขั้นตอนปลีกย่อยที่ซับซ้อน แต่สามารถจัดเป็นขั้นตอนใหญ่ ๆ ที่สำคัญ ซึ่งได้กล่าวถึงในบทที่แล้วดังนี้

1. จากข้อมูลเกี่ยวกับการมองข้อมูลจุดพิกัดของวัตถุในระบบพิกัดพื้นฐาน (World Coordinate System) ทุกจุดจะถูกแปลงไปเป็นข้อมูลในระบบพิกัดของการมอง

(Eye Coordinate System) โดยอาศัยการแปลงลักษณะในการมอง (Viewing Transformation)

2. ข้อมูลจุดพิกัดในระบบพิกัดของการมอง จะถูกแปลงไปเป็นข้อมูลจุดพิกัดพื้นฐานของภาพ (Screen Coordinate System) โดยการแปลงลักษณะทัศนียภาพ (Perspective Transformation) ข้อมูลเพิ่มเติมในขั้นตอนนี้ คือข้อมูลเกี่ยวกับอาณาเขตของการมองเห็น (Viewing Region หรือ Aperture) การทำงานช่วงนี้แบ่งเป็น 2 ขั้นตอนคือ

ก. ทำการชลิบขอบภาพจากข้อมูลในระบบพิกัดของการมอง และอาณาเขตของการมองเห็น

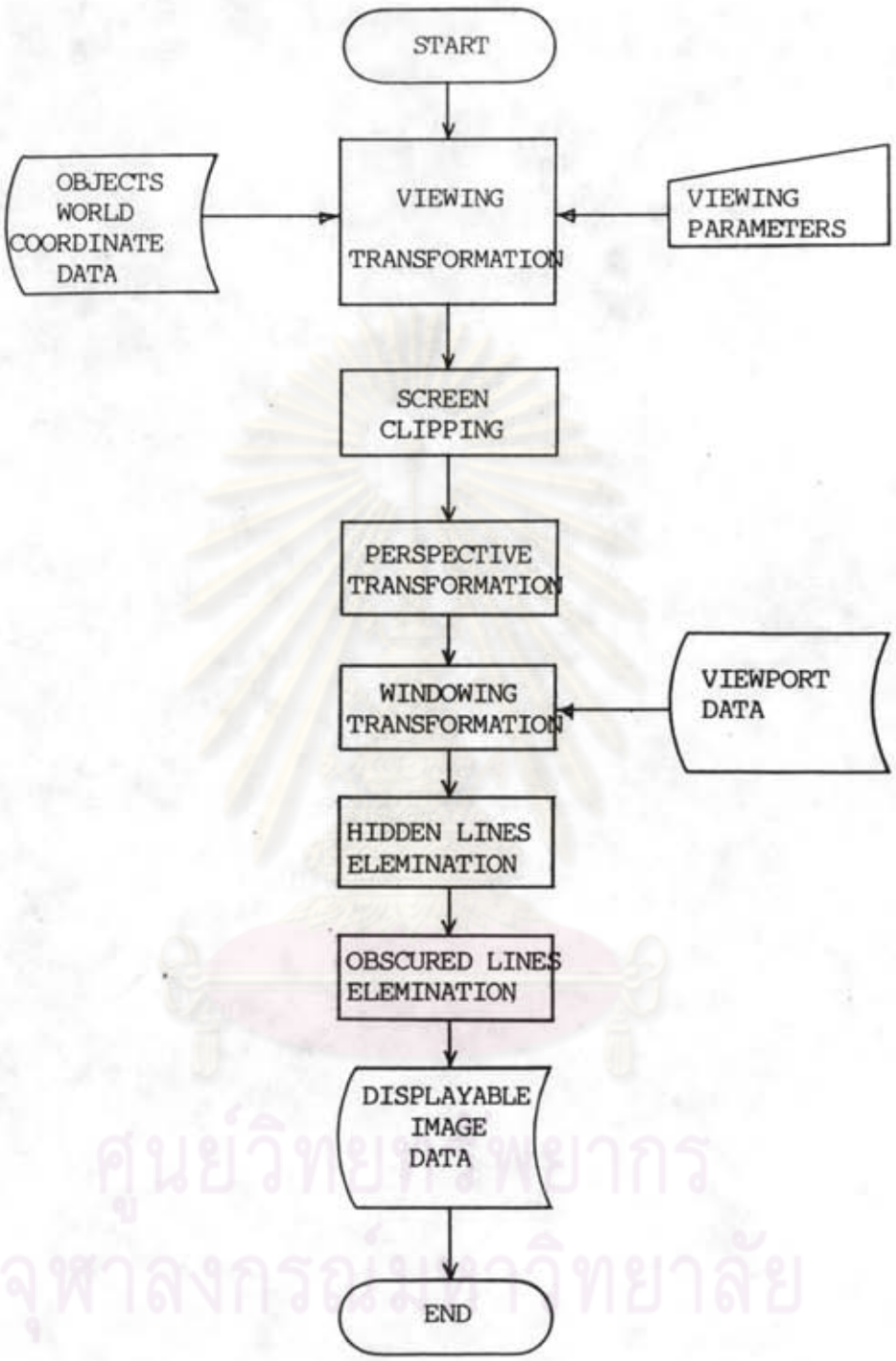
ข. แปลงข้อมูลส่วนที่เห็นได้ในส่วนที่เรียกว่า อาณาเขตภาพ (Window) ของวัตถุ จากระบบพิกัดของการมองไปเป็นข้อมูลในระบบพิกัดพื้นฐานของภาพ โดยการแปลงลักษณะทัศนียภาพ

3. แปลงข้อมูลพื้นฐานของภาพไปเป็นข้อมูลในระบบพิกัดของพื้นแสดงภาพ ซึ่งพิจารณาถึงช่องแสดงภาพ (Viewport) ด้วย กรรมวิธีขั้นตอนนี้เรียกว่า การแปลงลักษณะอาณาเขตภาพ (Windowing Transformation)

4. ลบส่วนของผิวหน้าที่มองไม่เห็น (Back Face) โดยพิจารณาถึงนอร์แมลเวคเตอร์ของแต่ละผิวหน้าของวัตถุ ดังรายละเอียด ในข้อ 2.8.1

5. ลบส่วนของวัตถุที่ถูกบังด้วยวัตถุอื่น ๆ หรือส่วนอื่นของวัตถุเดียวกัน (Obscured line Elimination)

ขั้นตอนต่าง ๆ มีกรรมวิธีปลีกย่อย ที่อาจจะคาบเกี่ยว หรือพัฒนาไปด้วยกันได้ และบางขั้นตอนก็สามารถทำงานได้ค่อนข้างอิสระ ลำดับขั้นตอนการทำงานตามแนวทางดังกล่าวข้างต้นแสดงไว้ในผังงานที่ 3.1 สำหรับขั้นตอนการทำงานที่แท้จริง อันเป็นข้อสรุปของการวิเคราะห์จะได้กล่าวถึงในภายหลัง



รูปที่ 3.1 ผังงานแสดงโครงสร้างทั่วไปของระบบงานสร้างทัศนียภาพ

3.2 การแปลงลักษณะในการมอง

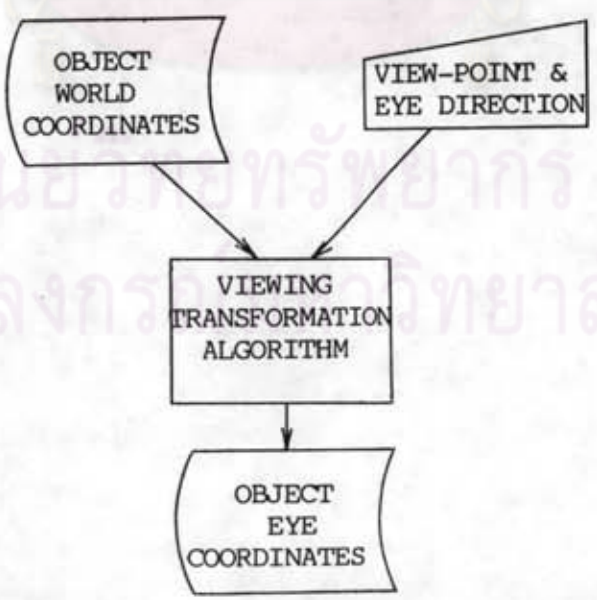
งานขั้นตอนแรกของระบบ คือ การแปลงข้อมูลจุดพิกัดพื้นฐานของวัตถุ ไปเป็นจุดพิกัดในการมอง โดยอาศัยการคำนวณตามสูตรในหัวข้อ 3.4 ระบบงานและข้อมูลที่เกี่ยวข้อง แสดงไว้ในผังงานที่ 3.2

ข้อมูลนำเข้ามี 2 ชุด คือ

1. ข้อมูลจุดพิกัดพื้นฐานของวัตถุ

2. ข้อมูลเกี่ยวกับการมอง ได้แก่ ค่าพิกัดของตำแหน่งจุดมอง และทิศทางของการมอง ซึ่งอยู่ในรูปของค่าพิกัดของจุดใด ๆ บนเส้น ซึ่งแสดงทิศทางการมอง

ข้อมูลทั้ง 2 ชุด จะถูกนำมาทำการคำนวณตามสูตรที่ (2- 7) ถึง (2-13) ปัญหาในการคำนวณ ที่เกิดขึ้นได้คือการหารด้วย 0 การคำนวณตามสูตรดังกล่าว สะท้อนถึงลักษณะการจัดองค์ประกอบเกี่ยวกับการมอง ซึ่งถือว่าแกน y ของระบบพิกัดในการมองอันเป็นแกนตั้งของระบบนั้น ชี้ขึ้นข้างบนเหนือจุดกำเนิดระบบ และแกน x ขนานกับแนวระดับเสมอ สำหรับทิศทางการมอง



รูปที่ 3.2 แสดงระบบงานแปลงลักษณะในการมอง

ใด ๆ การหารด้วย 0 เกิดขึ้นได้ใน 2 กรณีคือ เมื่อทิศทางของการมอง (แกน z ของระบบพิกัดในการมอง) อยู่ในแนวดิ่ง ซึ่งทำให้แกน Y อยู่ในแนวระดับ ไม่สามารถกำหนดตำแหน่งที่แน่นอนได้ และเมื่อจุดมองเป็นจุดเกี่ยวกับจุดแสดงทิศทางการมอง สำหรับแนวทางแก้ไขโดยทั่วไปคือ

1. ถ้าทิศทางการมองอยู่ในแนวดิ่งจากบนลงล่าง กำหนดให้ทิศทางของแกน y ของระบบพิกัดในการมอง พุ่งไปทางเบื้องหน้า ซึ่งเป็นทิศทางเดียวกับแกน y ของระบบพิกัดพื้นฐานของวัตถุ ผลที่ได้จะเสมือนการที่เราก้มมองวัตถุบนพื้น ซึ่งทิศทางของศีรษะพุ่งไปเบื้องหน้า

2. ถ้าทิศทางการมองอยู่ในแนวดิ่งจากล่างขึ้นบน กำหนดให้กลับกับข้อ 1

3. ถ้าจุดมองเป็นจุดเดียวกับจุดแสดงทิศทางการมองกำหนดให้เหมือนข้อ 1

วิธีการคำนวณในการแปลงลักษณะในการมองดังกล่าว แสดงขั้นตอนวิธีไว้ในรูป 3.3 ในลักษณะโปรแกรมย่อย VIEWING-TRANSFØRM ซึ่งรับข้อมูลจุดพิกัดพื้นฐานของวัตถุที่อยู่ในลักษณะข้อมูลชุด ซึ่งมี n จุด คำนวณและให้ผลลัพธ์เป็นข้อมูลชุดของจุดพิกัดในการมอง n จุดเช่นกันออกมา

ข้อสังเกต คือ เราสามารถคำนวณหาค่าคงที่ต่าง ๆ เกี่ยวกับการมอง เพื่อใช้กับจุดพิกัดทุกจุดของวัตถุ ซึ่งเป็นข้อมูลวัตถุเพียงชนิดเดียวที่เกี่ยวข้องกับการทำงานขั้นตอนนี้ และจากลักษณะดังกล่าว การทำงานจะมีประสิทธิภาพสูง ถ้าสามารถส่งข้อมูลจุดพิกัดทุก ๆ จุดของวัตถุเข้ามาทีเดียวทั้งหมด และทำการคำนวณแบบเรียงลำดับไปทีละจุด โดยไม่ต้องคำนึงถึงข้อมูลชนิดอื่น ๆ ของวัตถุเลย

3.3 การขลิบขอบภาพ และการแปลงลักษณะทัศนียภาพ

ข้อมูลจุดพิกัดในการมองจะต้องถูกโปรเจคให้เป็นข้อมูลภาพ ซึ่งเกิดจากการมอง อย่างไรก็ดี ก่อนที่จะสามารถทำการโปรเจคได้นั้น จำเป็นต้องมีการขลิบขอบภาพ กล่าวคือ ตัดส่วนของวัตถุที่อยู่พ้นพีระมิดของการมองเสียก่อน ส่วนของวัตถุที่จะต้องพิจารณา คือ เส้น

```
VIEWING_TRANSFORM:  PROCEDURE  (N,XW,YW,ZW,XE,YE,ZE,
                                XF,YF,ZF,XT,YT,ZT);
/*  (WC = World coordinate, EC = Eye coordinate)
    N         =>  Number of vertices
    Xw,Yw,Zw  =>  Arrays of WCs of object's vertices
    Xe,Ye,Ze  =>  Arrays of ECs of those vertices
    Xf,Yf,Zf  =>  Wc of Point-of-View-From (eyepoint)
    Xt,Yt,Zt  =>  Wc of Point-of-View-To              */
BEGIN;
   A := Xt - Xf;   /* A,B,C are viewing constants  */
   B := Yt - Yf;
   C := Zt - Zf;
   IF  (A = 0) & (B = 0)  THEN  /* Vertical eye direction  */
     IF C > 0 THEN              /* upward looking */
       DO I := 1 TO N;
         Xe(I) := Zf - Xw(I);
         Ye(I) := Yw(I) - Yf;
         Ze(I) := Zw(I) - Zf;
       END;
     ELSE
       DO I := 1 TO N;          /* downward looking,  or  */
         Xe(I) := Xw(I) - Xf;   /* direction unspecified  */
         Ye(I) := Yw(I) - Yf;
         Ze(I) := Zf - Zw(I);
       END;
   ELSE
     DO;   /* D,E,F, are also viewing constants */
       D := SQRT(A*A + B*B);
       E := SQRT(D*D + C*C);
       F := D*E;
       DO I := 1 TO N;
         X := Xf - Xw(I);
         Y := Yf - Yw(I);
         Z := Zf - Zw(I);
         S := A*X + B*Y;
         Xe(I) := (A*X - B*Y)/D;
         Ye(I) := (C*S + D*D*Z)/F;
         Ze(I) := -(S + C*Z)/E;
       END;
     END;
 END;
END VIEWING_TRANSFORM;
```

รูปที่ 3.3 แสดงขั้นตอนวิธีสำหรับการแปลงลักษณะในการมอง

3.3.1 การทดสอบเส้นกับขอบภาพ

การขลิบขอบภาพก็คือ การตรวจสอบและคำนวณหาข้อมูลของเส้นที่จะปรากฏได้ในภาพ ข้อมูลเส้นก็คือ ข้อมูลชุดที่ประกอบด้วยค่าจุดพิกัด 2 จุด คือ จุดปลายทั้งสองของเส้น สำหรับค่าจุดพิกัดในขั้นตอนนี้ก็คือ ค่าจุดพิกัดในระบบพิกัดของการมอง

ขั้นตอนการทำงานเริ่มด้วยการทดสอบตำแหน่งของจุดพิกัดของปลายทั้งสองของเส้นกับพีระมิดของการมอง โดยอาศัยเงื่อนไขตามสมการ (2-20) ซึ่งเป็นเงื่อนไขที่แสดงว่า จุดพิกัดใด ๆ จะปรากฏอยู่ภายในพีระมิดของการมอง การทดสอบจะกระทำที่ระดับความลึก (Ze) ของจุดพิกัดที่ต้องการตรวจสอบเสมอ โดยถือเสมือนว่าพื้นรับภาพอยู่ที่นั่น และเราสามารถคำนวณหาความกว้างของพื้นรับภาพ ณ ระดับความลึก Ze ของจุดที่พิจารณาได้จากสมการ

$$W = \frac{Ze}{D/S}$$

โดย D/S เป็นอัตราส่วนระหว่างระยะทางตามทิศทางการมอง วัดจากจุดมองไปยังพื้นรับภาพ (D) กับความกว้างของพื้นรับภาพ วัดจากจุดกึ่งกลางถึงขอบ (S) อัตราส่วน D/S จะคงที่เสมอสำหรับระบบงานสร้างภาพหนึ่ง ๆ

จากการเปรียบเทียบค่า W กับค่าพิกัด Xe และ Ye ก็จะสามารถบอกได้ว่าจุดพิกัดนั้น ๆ อยู่ในพีระมิดของการมองหรือไม่ อย่างไรก็ดี เส้น 1 เส้น จะต้องทำการทดสอบจุดปลายเส้น 2 จุด ในทิศทางของแกนต่าง ๆ ทั้ง 3 แกน ซึ่งจะได้ผลสรุป ดังนี้

1. ถ้าจุดทั้งสองอยู่พ้นพีระมิดของการมองในด้านเดียวกัน (เช่น ในทิศทางของแกน+x) เส้นดังกล่าวไม่ปรากฏส่วนใดในพีระมิดเลย

2. ถ้าจุดทั้งสองอยู่ภายในพีระมิด เส้นทั้งเส้นจะอยู่ในพีระมิด

3. ถ้าจุดหนึ่งอยู่นอกพีระมิด อีกจุดหนึ่งอยู่ในพีระมิด เส้นดังกล่าวจะปรากฏเพียงบางส่วน ซึ่งจะต้องทำการขลิบ

4. ถ้าจุดทั้งสองอยู่พ้นพีระมิด แต่คนละด้านกัน เส้นดังกล่าวอาจอยู่พ้นพีระมิดเหมือนข้อ 1 หรืออาจตัดผ่านพีระมิดเป็นบางส่วนก็ได้

การขลิบจะกระทำต่อเส้นในลักษณะข้อ 3 และ 4 สำหรับเส้นตามข้อ 4 นั้น ขั้นตอนการขลิบ จะบ่งบอกโดยอัตโนมัติ ว่าเส้นดังกล่าวจะปรากฏอยู่ในพีระมิดบางส่วนหรือไม่

3.3.2 การขลิบเส้น

กรรมวิธีในการขลิบเส้น เริ่มด้วยการกำหนดขนาดความยาวของเส้นเป็น 1 หน่วย โดยตำแหน่งของจุดปลายเส้นทั้งสองเป็น 0 และ 1 ตามลำดับ เมื่อปรากฏว่าจุดปลายเส้นจุดใดอยู่พ้นพีระมิด เราจะคำนวณหาตำแหน่งที่ปลายเส้นตัดกับพีระมิด ณ ด้านนั้น ๆ ในรูปของอัตราส่วนกับความยาวเส้นทั้งหมด ถ้าปลายแรกถูกขลิบจะได้ค่าตำแหน่งมากกว่า 0 ถ้าปลายที่สองถูกขลิบจะได้ค่าตำแหน่งที่น้อยกว่า 1 เมื่อคำนวณครบทุกด้านแล้ว ถ้าค่าตำแหน่งของปลายแรกมากกว่าค่าตำแหน่งของปลายที่สอง เส้นดังกล่าวจะเป็นเส้นในกรณีที่ 4 ซึ่งไม่มีส่วนใดตัดผ่านพีระมิดของการมอง และจะต้องถูกตัดทิ้งออกจากรายการเส้น เช่นเดียวกับเส้นในกรณีที่ 1 ของการทดสอบ ถ้ามิเช่นนั้นแล้ว เราก็จะคำนวณหาค่าพิกัดที่แท้จริงของตำแหน่งจุดปลายเส้นทั้งสองออกมาได้ ซึ่งก็จะได้ผลเป็นข้อมูลเส้นที่ถูกขลิบแล้วนั่นเอง

3.3.3 การโปรเจคภาพ

ข้อมูลเส้นที่ผ่านการทดสอบและขลิบมาแล้ว จะเป็นข้อมูลที่พร้อมที่ทำการแปลงลักษณะทัศนียภาพได้ งานขั้นตอนนี้ ไม่มีอะไรซับซ้อน กล่าวคือ สามารถใช้สูตรคำนวณสำหรับการแปลงลักษณะทัศนียภาพในบทที่แล้ว คือสูตร (2-14) หรือ (2-15) ได้ทันที สำหรับสูตร (2-15)นั้น ค่าจุดพิกัดที่ได้จะอยู่ในรูปของอัตราส่วนกับขนาดของพื้นรับภาพ ซึ่งสะดวกกับการแปลงเป็นข้อมูลพิกัดสื่อแสดงภาพ (ตามรายละเอียดในหัวข้อที่ 2.6) ต่อไป

สำหรับกรรมวิธีในขั้นตอนนี้ ทั้งหมดแสดงขั้นตอนวิธีไว้ในรูป 3.4 ในลักษณะโปรแกรมย่อยชื่อ PERSPECTIVE-TRANSFORM ข้อมูลเกี่ยวกับพื้นรับภาพคือ ค่าในตัวแปร SX, SY และ D

```
PERSPECTIVE_TRANSFORM: PROCEDURE  (X1,Y1,Z1,X2,Y2,Z2,SX,SY,D,
                                   VISIBLE,XS1,YS1,XS2,Ys2);
  /* X1,Y1,Z1 => Eye coordinate of 1st end of a line
     X2,Y2,Z2 => Eye coordinate of 2nd end of the line
     SX,SY    => Screen (picture plane) width from center
     D        => Relative distance from eye point to screen
     VISIBLE  => Boolean Flag
     XS1,YS1,XS2,YS2 => Screen coordinates of both points */
  BEGIN;
  /* Step0:  Find pyramid boundary at depth of both points */
    WX1  := SX*Z1/D;  WY1  := SY*Z1/D;    /* 1st point */
    WX2  :- SX*Z2/D;  WY2  := SY*Z2/D;    /* 2nd point */
  /* Step1:  Test both points against viewing pyramid */
    A(1)  := WX1 + X1;  B(1)  := WX2 + X2;   /* left edge  */
    A(2)  := WX1 - X1;  B(2)  :- WX2 - X2;   /* right edge */
    A(3)  := WY1 + Y1;  B(3)  := WY2 + Y2;   /* bottom edge */
    A(4)  := WY1 - Y1;  B(4)  := WY2 - Y2;   /* top edge   */
    A(5)  := Z1;        B(5)  := Z2;         /* depth      */
    DO I  := 1 TO 5;
      IF A(I)<0 & B(I)<0 THEN
        DO;   VISIBLE  := FALSE;   /* both points lie on the */
              RETURN;              /* same side, off screen. */
        END;
    END;
  /* Step2:  Clipping:  find proportional clipped points */
    T1 := 0;   T2 := 1;   /* Set line length = 1 unit */
    DO I := 1  TO 5;
      IF A(I)<0 1 B(I)<0 THEN
        DO;  T := A(I)/(A(I) - B(I));
             IF  A(I)<0 THEN
                   IF T>T1 THEN T1 := T;  /* 1st end */
             ELSE  IF T<T2 THEN T2 := T;  /* 2nd end*/
        END;
    END;
    IF T2<T1 THEN
      DO;  VISIBLE := FALSE;  /* No part of the line */
           RETURN;            /* falls within screen.*/
      END;
  /* Step3: Find coordinates of the clipped point(s) */
    DX := X2 - X1;  DY := Y2 - Y1;  DZ := Z2 - Z1;
    IF T1< >0 THEN  /* 1st end */
      DO;  X1:=X1+T1*DX;  Y1:=Y1+T1*DY;  Z1:=Z1+T1*DZ;  END;
    IF T2< >1 THEN  /* 1nd end */
      DO;  X2:=X1+T2*DX;  Y2:=Y1+T2*DY;  Z2:=Z1+T2*DZ;  END;
    VISIBLE := TRUE;
  /* Step4; Perspective Projection */
    WX1 := SX/D;  WY1 := SY/D;
    XS1 := X1/(WX1*Z1);  YS1 := Y1/(WY1*Z1);
    XS2 := X2/(WX1*Z2);  YS2 := Y2/(WY1*Z2);
    RETURN;
  END;
END PERSPECTIVE_TRANSFORM;
```

รูปที่ 3.4 แสดงขั้นตอนวิธีสำหรับการชลิบภาพและโปรเจคภาพ

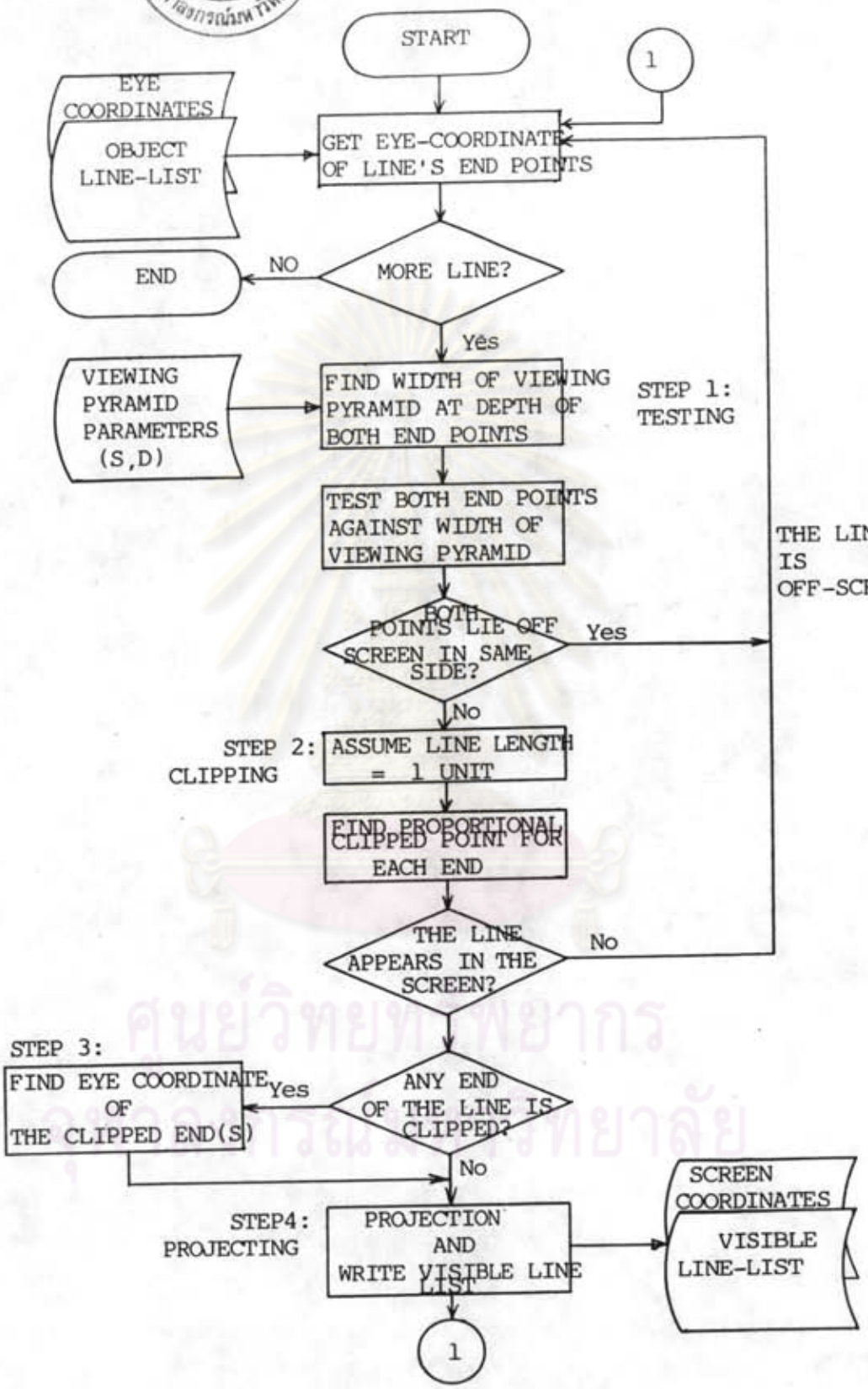
ซึ่งถูกกำหนดไว้ก่อน โดยอาศัยขนาดของอาณาเขตของการมองเห็น กล่าวคือ ถ้าระบบกำหนดให้อาณาเขตของการมองเห็นกินรัศมี 15 องศา รอบทิศทางการมอง และให้ระยะห่างเปรียบเทียบจากจุดมองถึงพื้นรับภาพ (D) เป็น 1 ค่า SX และ SY จะเท่ากับค่า TAN ของมุม 15 องศา แต่อย่างไรก็ตาม เราสามารถกำหนดให้น้อยกว่านี้ได้ และ SX ไม่จำเป็นต้องเท่ากับ SY และเพื่อความชัดเจนของขั้นตอนของแอลกอริทึม ข้อมูลนำเข้าจึงเป็นข้อมูลจุดพิกัดในระบบพิกัดของการมองของจุดปลายทั้งสอง ของเส้นตรงเส้นเดียว คือ X1, Y1, Z1 และ X2, Y2, Z2 และข้อมูลที่ส่งคืนให้โปรแกรมที่เรียกมาคือ XS1, YS1 และ XS2, YS2 ซึ่งเป็นค่าพิกัดในระบบพิกัดพื้นรับภาพพร้อมกับ VISIBLE ซึ่งเป็นแฟลกที่ระบุว่า เส้นดังกล่าวปรากฏส่วนหนึ่งส่วนใด หรือทั้งหมดในพื้นรับภาพหรือไม่ ในกรณีนี้ถือว่า โปรแกรมที่เรียกมา จะต้องส่งค่าพิกัดจุดปลายทั้งสองของเส้นมาให้ทีละเส้นต่อการเรียก 1 ครั้ง และรับผลการโปรเจคเส้นดังกล่าวบนพื้นรับภาพกลับไป

ลักษณะการทำงานของระบบงานขั้นตอนนี้ แสดงไว้ในผังงานตามรูปที่ 3.5

3.4 การโปรเจคภาพแบบขนาน

เมื่อทิศทางการมองถูกกำหนดแล้ว ขั้นตอนแรกของการโปรเจคภาพแบบขนานก็คือ การแปลงลักษณะในการมอง ซึ่งขั้นตอนวิธีในการแปลงลักษณะในการมองสำหรับภาพแบบขนานนั้น ไม่แตกต่างกับการแปลงลักษณะในการมองสำหรับทัศนียภาพดังกล่าวในหัวข้อ 3.2 เลย และสามารถใช้แอลกอริทึมเดียวกันได้ อย่างไรก็ดี สำหรับภาพแบบออโธกราฟฟิคนั้น เราสามารถพัฒนาแอลกอริทึมที่ง่ายและเร็วกว่าแอลกอริทึมในหัวข้อ 3.2 ได้

เมื่อผ่านการแปลงลักษณะในการมองแล้ว เราจะได้จุดพิกัดในระบบพิกัดของการมองสำหรับภาพแบบขนานนั้น อาณาเขตของการมองเห็นไม่มี ดังนั้นไม่มีความจำเป็นใด ๆ ในการขลิบขอบภาพ และเมื่อทำการโปรเจคภาพแบบขนานจุดพิกัดที่ได้นั้นจะมีค่าเป็นจุดพิกัดเดิมนั่นเอง ดังนั้น สำหรับภาพแบบขนานการแปลงลักษณะในการมองเพียงขั้นตอนเดียว ก็จะเป็นการโปรเจคภาพเสร็จสิ้นไปในตัวด้วย



รูปที่ 3.5 แสดงผังงานการชลิบภาพและการโปรเจคภาพ

3.4.1 ภาพแบบออโธกราฟฟิค

ทิศทางการมองสำหรับภาพแบบนี้ จะขนานกับแกน x,y หรือ z ของระบบพิกัดพื้นฐานของวัตถุเสมอ ซึ่งมีอยู่ 6 ทิศทางทั้งบวกและลบ ดังนั้นเราสามารถพัฒนาแอลกอริทึมการแปลงลักษณะในการมอง โดยไม่ต้องคำนึงถึงจุดมอง หรือจุดแสดงทิศทางการมองได้ กล่าวคือ เราสามารถใช้ค่าพิกัดในระบบพิกัดพื้นฐานของวัตถุได้โดยตรง เพียงแค่เปลี่ยนเครื่องหมายบวกลบให้ถูกต้องกับทิศทางที่ต้องการ เช่น รูปด้านหน้าจะมีทิศทางการมองขนานกับแกน y ไปในทางบวก ค่าพิกัดแกน +x ของระบบพิกัดในการมอง จะเป็นค่าในแกน +x ของระบบพิกัดพื้นฐาน ค่าพิกัดแกน +y ของระบบพิกัดในการมอง จะเป็นค่าในแกน +zของระบบพิกัดพื้นฐาน เป็นต้น

รูปที่ 3.6 แสดงขั้นตอนวิธีโปรเจคภาพออโธกราฟฟิค ในลักษณะดังกล่าว

3.4.2 ภาพแบบอะโซโนเมตริก

สำหรับภาพแบบอะโซโนเมตริกนั้น จำเป็นต้องใช้ขบวนการคำนวณแบบเดียวกับภาพเพอสเพคทีฟตามที่กล่าวมาแล้ว เพียงแต่ภาพอะโซโนเมตริกแบบมาตราฐานคือ ไอโซเมตริก ไดเมตริก และไตรเมตริก นั้น ระบบงานสามารถกำหนดจุดมอง และจุดแสดงทิศทางการมองเอง โดยผู้ใช้งานระบุว่าต้องการมองจากซีกใดของวัตถุ (ในระบบพิกัดพื้นฐาน) เช่น ด้านหน้า-ขวา ด้านหลัง-ขวา เป็นต้น สำหรับภาพอะโซโนเมตริกแบบอื่น ๆ อาจให้ผู้ใช้กำหนดทิศทางการมองที่ต้องการเองได้ ในทำนองเดียวกับภาพเพอสเพคทีฟ เพียงแต่ตำแหน่งที่แท้จริงของจุดมองไม่มีผลต่อลักษณะภาพ โดยทั่วไป เพื่อความรวดเร็วและประหยัดขั้นตอนการคำนวณเราจะให้จุดกำเนิดของระบบพิกัดพื้นฐานเป็นจุดมอง และคำนวณตำแหน่งจุดแสดงทิศทางการมองโดยมีค่าในแกน x และ y เป็น ± 1 และค่าในแกน z คำนวณขึ้นจากมุมมอง เช่น ภาพแบบไอโซเมตริก จะมีค่า z เป็น tan (1/sin 45) เสมอ เป็นต้น

ข้อสังเกตประการหนึ่งก็คือ ขนาดของภาพอะโซโนเมตริก ที่ได้จากการโปรเจคจะเล็กกว่าภาพเดียวกัน ที่ได้จากการเขียนแบบเสมอ เช่น ภาพไอโซเมตริกโปรเจคชั่น จะมีขนาดเป็น 1/1.23 เมื่อเทียบกับแบบเขียนไอโซเมตริก ส่วนภาพอะโซโนเมตริกแบบอื่น ๆ

```
ORTHO-PROJECTION:  PROCEDURE  (VIEW-TYPE,N,WCO,SCO);
  /* WCO(I,1..3) => Array of world coordinates, x..z
     SCO(I,1..3) => Array of projecting-screen coordinates
     N           => Number of coordinates                    */
  BEGIN;
    IF VIEW-TYPE = 'TOP' | VIEW-TYPE = 'BOTTOM' THEN
      DO;
        POINT1 := 1;
        POINT2 := 2;
        POINT3 := 3;
      END;
    ELSE
      IF VIEW-TYPE = 'LEFT' | VIEW-TYPE = 'RIGHT' THEN
        DO;
          POINT1 := 2;
          POINT2 := 3;
          POINT3 := 1;
        END;
      ELSE
        DO;
          POINT1 := 1;
          POINT2 := 3;
          POINT3 := 2;
        END;
   DO  I := 1 TO N;
      IF VIEW-TYPE = 'BOTTOM' |
         VIEW-TYPE = 'LEFT'   |
         VIEW-TYPE = 'REAR'   THEN
        SCO(I,1) := -WCO(I,POINT1);
      LESE
        SCO(I,1) := WCO(I,POINT1);
      SCO(I,2) := WCO(I,POINT2);
      IF VIEW-TYPE = 'TOP'   |
         VIEW-TYPE = 'RIGHT' |
         VIEW-TYPE = 'REAR'  THEN
       SCO(I,3) := -WCO(I,POINT3);
      ELSE
        SCO(I,3) := WCO(I,POINT3);
    END;
    RETURN;
  END;
END ORTHO-PROJECTION;
```

รูปที่ 3.6 แสดงขั้นตอนวิธีสำหรับการโปรเจคภาพออโธกราฟฟิค

```
ISOMETRIC:  PROCEDURE  (SIDE,SIZE,N,WCO,SCO);
  /* SIDE          => Viewing side
     SIZE          => Drawing/Projecting size indicator
     N             => Number of coordinates
     WCO(I,1..3)   => Array of world coordinates
     SCO(I,1..3)   => Array of screen coordinates      */
  BEGIN;
  /* Define Point-of-view-to where eye point is (0,0,0)  */
    IF SIDE = 'FRONT-RIGHT' | SIDE = 'REAR-LEFT' THEN
      Xt := -1;
    ELSE
      Xt := 1;
    IF SIDE = 'REAR-RIGHT' | SIDE = 'REAR-LEFT' THEN
      Yt := -1;
    ELSE
      Yt := 1;
    Zt := TAN(1/SIN(45)); /* Degree mode is assumed */
  /* Begin Transformation, (i.e. Projection) */
    D := SQRT(2.0);
    E := SQRT(2.0 + Zt*Zt);
    F := D*E;
    DO I := 1 TO N;
      SCO(I,1) := (Yt*WCO(I,2) - Xt*WCO(I,1))/D;
      S := -(Xt*WCO(I,1) + Yt*WCO(I,2));
      SCO(I,2) := (Zt*S - 2*WCO(I,3))/F;
      SCO(I,3) := (Zt*WCO(I,3)+ S)/E;
    END;
  /* To this point, projection is finished and the image
     can be transformed into drawing size if needed      */
    IF SIZE = 'DRAWING' THEN
     DO I := 1 TO N;
       DO J := 1 TO 3;
         SCO(I,J) := SCO(I,J)*1.23;
         /* where 1.23 is Isometric scaling constant */
       END;
     END;
   RETURN;
 END;
END ISOMETRIC;
```

รูปที่ 3.7 แสดงขั้นตอนวิธีสำหรับการโปรเจคภาพไอโซเมตริก

จะขึ้นอยู่กับการกำหนดมุมชันของเส้นในแต่ละแกนสำหรับแต่ละหน่วยงาน

รูปที่ 3.7 แสดงขั้นตอนวิธีโปรเจคภาพไอโซเมตริกตามนัยที่กล่าวมานี้ สูตรคำนวณใช้สูตรเดียวกับที่ใช้ในหัวข้อ 3.2 แต่จะเห็นได้ว่าลดขั้นตอนลงไปได้มาก สำหรับภาพอะโซโนเมตริกแบบอื่น ก็ใช้แอลกอริทึมเดียวกันนี้ได้ทุกประการ ยกเว้นการคำนวณค่าพิกัด Z ของจุดแสดงทิศทางการมอง และค่าแฟคเตอร์แสดงอัตราส่วนของภาพโปรเจคชันกับภาพจากการเขียนแบบ

3.4.3 การปรับขนาดภาพแบบขนาน

ลักษณะสำคัญที่เกิดจากการโปรเจคภาพแบบขนานตามที่กล่าวมาในหัวข้อที่ 3.4.1 และ 3.4.2 คือ ข้อมูลที่ได้เป็นข้อมูลในระบบพิกัดพื้นรับภาพ ซึ่งไม่มีขอบเขตจำกัด ทางทฤษฎีกล่าวคือ ขนาดของภาพที่ได้ขึ้นอยู่กับขนาดของวัตถุโดยตรง ดังนั้นในขั้นตอนต่อ ๆ ไป ในการแปลงข้อมูลลงบนสื่อแสดงภาพ ภาพที่ได้อาจมีขนาดโตเกินช่องแสดงภาพ ปัญหาข้อนี้แก้ได้ 2 ลักษณะคือ

ก. ทำการชลิบภาพ ให้เหลือเฉพาะส่วนที่พอดีกับช่องแสดงภาพ ขบวนวิธีชลิบภาพนี้จะเป็นการชลิบภาพ 2 มิติ ซึ่งเราสามารถใช้ขั้นตอนวิธี ในการชลิบภาพ อันเดียวกับการชลิบภาพใน 3 มิติ ดังที่ปรากฏในรูปประกอบที่ 3.4 ได้โดยไม่จำเป็นต้องคำนวณหาค่าความกว้าง (W) ของขอบภาพจากความลึกใด ๆ แต่สามารถใช้ความกว้างของช่องแสดงภาพนั่นเองในการตรวจสอบลักษณะการทำงานอื่น ๆ ในการชลิบภาพจะเหมือนกันหมด

ข. ทำการลดขนาดวัตถุในภาพลง ให้สามารถแสดงในช่องแสดงภาพได้ทั้งหมด วิธีการนี้ เหมาะสำหรับงานสร้างภาพแบบขนานโดยทั่วไป ซึ่งมักต้องการให้เห็นวัตถุทั้งหมด จุดสำคัญในการลดขนาดภาพก็คือ การหาอัตราส่วนที่เหมาะสม วิธีการหาอัตราส่วนก็คือ การหาค่าพิกัดที่มากที่สุด และน้อยที่สุดในแกน X และ Y ของวัตถุในภาพ ค่าที่ได้ทั้ง 4 ค่า จะเป็นค่าที่เสมือนกำหนดกรอบอาณาเขต (Bounding Box) ของวัตถุในภาพ และจากการเปรียบเทียบขนาดกรอบอาณาเขตของวัตถุ กับขนาดช่องแสดงภาพ เราก็จะได้อัตราส่วนในการเปลี่ยนขนาดที่ต้องการได้ รูปที่ 3.8 แสดงขั้นตอนวิธี ตรวจหากรอบอาณาเขตของวัตถุ อัตราส่วนภาพและการลดขนาดภาพที่โตเกินช่องแสดงภาพลงให้พอดีกับช่องแสดงภาพ

```
VIEWPORT_ADJUST:  PROCEDURE  (N,SCO,VSX,VSY);
  /* SCO(I,1..3) => Array of screen coordinates
     N           => Number of coordinates
     VSX,VSY     => Widths of viewport from its center */
  BEGIN;
  /* Find Boundary Box */
    MAX-X := SCO(1,1);
    MIN-X := SCO(1,1);
    MAX-Y := SCO(1,2);
    MIN-Y := SCO(1,2);
    DO I := 2 TO N;
      IF MAX-X < SCO(I,1) THEN MAX-X := SCO(I,1);
      IF MIN-X > SCO(I,1) THEN MIN-X := SCO(I,1);
      IF MAX-Y < SCO(I,2) THEN MAX-Y := SCO(I,2);
      IF MIN-Y > SCO(I,2) THEN MIN-Y := SCO(I,2);
    END;
  /* Find scale in Z */
    SMAX := ABS(VSX/MAX-X);
    SMIN := ABS(VSX/MIN-X);
    IF SMAX > SMIN THEN
      SMAX := SMIN;
  /* Find scale in Y */
    SMAX-Y := ABS(VSY/MAX-Y);
    SMIN   := ABS(VSY/MIN-Y);
    IF SMAX-Y > SMIN THEN
      SMAX-Y := SMIN;
  /* Find overall scale */
    IF SMAX > SMAX Y THEN
      SMAX := SMAX Y;
  /* Scaling transform if image is bigger than viewport */
    IF SMAX < 1.0 THEN
      DO I := 1 TO N;
        DO J := 1 TO 3;
          SCO(I,J) := SCO(I,J)*SMAX;
        END;
      END;
    RETURN;
  END;
END VIEWPORT_ADJUST;
```


รูปที่ 3.8 แสดงขั้นตอนวิธีสำหรับการปรับขนาดภาพกับช่องภาพ

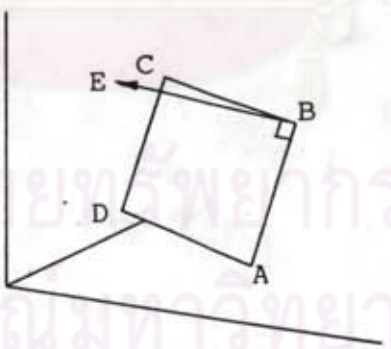
3.5 การลบส่วนที่มองไม่เห็น (Hidden Line Elimination)

การกำหนดว่าผิวหน้าใดของวัตถุเป็นผิวหน้าที่มองไม่เห็น ทำได้โดยการตรวจความสัมพันธ์เชิงมุมระหว่างนอร์แมลเวคเตอร์ของมัน กับตำแหน่งจุดมองตามหลักวิธีในหัวข้อ 2.8 ดังนั้นก่อนอื่นเราจำเป็นต้องพิจารณาการสร้างข้อมูลนอร์แมลเวคเตอร์ของผิวหน้าต่าง ๆ ของวัตถุ

3.5.1 การสร้างข้อมูลผิวหน้าและนอร์แมลเวคเตอร์

ผิวหน้าของวัตถุ มีคุณสมบัติเป็นพื้นที่ซึ่งวางต่อเนื่องกัน และพื้นที่ประกอบขึ้นจากเส้นซึ่งปิดล้อม ข้อมูลพื้นที่ก็คือ รายการชุดของข้อมูลเส้นนั่นเอง ในขณะเดียวกันข้อมูลเส้นก็อธิบายด้วยข้อมูลจุดปลายทั้งสองของแต่ละเส้น

อย่างไรก็ดีพื้นที่ดังกล่าวมีด้านที่เป็นผิวหน้าอยู่ 2 ด้าน ในการอธิบายผิวหน้าของวัตถุ 3 มิติให้ครบถ้วน จำเป็นจะต้องมีข้อมูลที่บ่งชี้ได้ว่า ผิวหน้าด้านใด มีคุณสมบัติเป็นผิวหน้าด้านนอก ที่อาจมองเห็นได้ (เมื่อวัตถุดังกล่าวเป็นวัตถุทึบแสง) และผิวหน้าใดมีคุณสมบัติเป็นผิวหน้าด้านใน (back face) เราสามารถอธิบายคุณสมบัติของผิวหน้าดังกล่าว โดยอาศัยสิ่งที่เรียกว่า นอร์แมลเวคเตอร์



รูปที่ 3.9 แสดงลักษณะพื้นที่ผิวหน้าและนอร์แมลเวคเตอร์

โดยที่นอร์แมลเวคเตอร์ดังกล่าว มีจุดกำเนิดบนจุดใดจุดหนึ่งบนพื้นที่ผิวหน้า และพุ่งในทิศทางตั้งฉากออกไปจากพื้นที่ผิวหน้าในด้านใดด้านหนึ่ง โดยการกำหนดความสัมพันธ์ระหว่างทิศทางของ

นอร์แมลเวคเตอร์กับพื้นที่ผิวหน้า เราก็จะสามารถอ้างอิงได้ในภายหลังว่า ด้านใดของพื้นที่ผิวหน้าหนึ่ง ๆ เป็นผิวหน้าด้านนอกหรือด้านใน ดังตัวอย่างในรูปที่ 3.9 BE เป็นนอร์แมลเวคเตอร์ มีจุดกำเนิดที่จุด B ของพื้นที่ผิวหน้า ABCD และพุ่งออกไปในทิศทางเดียวกับผิวหน้าด้านนอก (Expose face)

ลักษณะสำคัญในการสร้างข้อมูลผิวหน้า และนอร์แมลเวคเตอร์ที่สามารถตรวจสอบได้ในภายหลังก็คือ การจัดลำดับของจุดพิกัดยอดมุมของพื้นที่อย่างแน่นอน และการกำหนดตำแหน่งจุดกำเนิดนอร์แมลเวคเตอร์ที่เป็นระเบียบ เพื่ออ้างถึงได้ในขั้นตอนการทำงานที่เกี่ยวข้อง ดังตัวอย่างที่กำลังกล่าวถึงนี้ การอธิบายพื้นที่ผิวหน้าด้วยลำดับของจุดพิกัด A-B-C-D จะสามารถบอกถึงผิวหน้าด้านที่พิจารณาได้ว่า เป็นด้านใดด้านหนึ่ง (ในที่นี้คือด้านนอก) และถ้าเรากำหนดตายตัวไปว่า จุดกำเนิดของนอร์แมลเวคเตอร์คือ จุดพิกัดจุดที่ 2 ของข้อมูลลำดับพิกัดของพื้นที่ผิวหน้า และมีทิศทางพุ่งออกมาทางด้านที่เรากำหนด (ในที่นี้คือด้านนอก) แล้ว การเพิ่มข้อมูล E ประกอบลงไปในข้อมูลผิวหน้าก็เป็นการเพียงพอที่จะอธิบายข้อมูลเกี่ยวกับตำแหน่งขอบเขต และคุณสมบัติของผิวหน้าได้เพียงพอ ตัวอย่างข้อมูลในที่นี้ก็คือ ABCD:E คือผิวหน้า ABCD ซึ่งมีนอร์แมลเวคเตอร์BE พุ่งจากจุด B มาทางด้านที่มองเห็น ซึ่งคือด้านที่ A-B-C-D เรียงหวนเข็มนาฬิกาเมื่อมองสวนทิศทางของนอร์แมล-เวคเตอร์เข้าไป

ค่าจุดพิกัด E จะได้มาจากการคำนวณ โดยมีข้อมูลจุดพิกัด B และจุดข้างเคียงคือ Aและ C โดยกำหนดให้ BA และ BC เป็นเวคเตอร์ที่ 1 และ 2 การคำนวณหาค่าครอสโปรดัคของเวคเตอร์ BA และ BC จะได้เป็นเวคเตอร์ BE(ดูรายละเอียดในภาคผนวก ค)

ถ้าเรากำหนดให้ SURFACE (I, J) เป็นข้อมูลชุดของผิวหน้าที่ I ในฐานข้อมูล ซึ่งประกอบด้วยข้อมูลเกี่ยวกับจุดพิกัดยอดมุม n จุดตามลำดับ หวนเข็มนาฬิกาของผิวหน้าด้านนอก ซึ่งเก็บไว้ที่ SURFACE (I, J) ถึง SURFACE (I, n) ในรูปของตัวบ่งชี้ (pointer) ไปยังชุดข้อมูลจุดพิกัดในระบบพิกัดพื้นฐานของวัตถุคือ WCØ (n, K) โดยที่ K = 1,2,3 หมายถึงค่าพิกัดตามแกน X,Y,Z ตามลำดับ ถ้า NV (I,K) เป็นข้อมูลจุดปลายนอร์แมลเวคเตอร์ของผิวหน้าที่ I เราจะสร้างข้อมูลนอร์แมลเวคเตอร์ ที่พุ่งออกมาทางผิวหน้าด้านนอกได้ตามแอลกอริทึมในรูปที่ 3.10

```
CREATE-NV:  PROCEDURE  (SURFACE,WCO,NV,P);
  /* SURFACE(P,1..N) => Vertices list of surface P
     WCO(I,1..3)     => Array of world coordinates
     NV(P,1..3)      => Array of world coordinates of the
                        end point of Normal Vector of the surface P
     P               => Number of surfaces                          */
 BEGIN;
   I := 1;
   DO WHILE (I <=P);
     /* Get pointers, i.e. subscripts of WCO array */
          A := SURFACE(I,1);
          B := SURFACE(I,2);
          C := SURFACE(I,3);
     /* Translate vectors BA, BC to the origin of world coordinate system */
          V1 := WCO(A,1) - WCO(B,1);
          V2 := WCO(A,2) - WCO(B,2);
          V3 := WCO(A,3) - WCO(B,3);
          W1 := WCO(C,1) - WCO(B,1);
          W2 := WCO(C,2) - WCO(B,2);
          W3 := WCO(C,3) - WCO(B,3);
     /* Find normal vector in term of origin vector */
          NV(I,1) := V2*W3 -V3*W2;
          NV(I,2) := V3*W1 -V1*W3;
          NV(I,3) := V1*W2 -V3*W2;
     /* Translate normal vector to its origin:  point B */
          NV(I,1) := NV(I,1) + WCO(B,1);
          NV(I,2) := NV(I,2) + WCO(B,2);
          NV(I,3) := NV(I,3) + WCO(B,3);
     I := I + 1;
   END;
   RETURN;
 END;
END CREATE-NV;
```

รูปที่ 3.10 แสดงขั้นตอนวิธีสำหรับการสร้างข้อมูลนอร์แมลเวคเตอร์

3.5.2 การตรวจสอบผิวหน้าที่มองเห็น

การตรวจสอบว่าผิวหน้าใด จะมองเห็นหรือไม่นั้น อาศัยหลักการที่ว่า ผิวหน้าใดที่มองเห็นจะต้องทำมุมน้อยกว่า 180 องศากับทิศทางการมองจากจุดมองไปยังผิวหน้านั้น มีขั้นตอนดังนี้

1. กำหนดจุดกำเนิดของนอร์แมลเวคเตอร์บนผิวหน้าเป็นจุดแสดงทิศทางการมอง จุดมองเป็นจุดเดิม

2. จากจุดมองและทิศทางการมองตามข้อ 1 ทำการแปลงลักษณะในการมองของจุด

กำเนิดและจุดปลายของนอร์แมลเวคเตอร์

3. เปรียบเทียบผลที่ได้ ถ้าค่าพิกัดความลึก (Z) ในระบบพิกัดจากการมองของจุดปลายของนอร์แมลเวคเตอร์ น้อยกว่าค่าพิกัดความลึกในระบบเดียวกันของจุดกำเนิดนอร์แมลเวคเตอร์ ผิวหน้าด้านนั้นจะเป็นผิวหน้าที่มองเห็น (นั่นคือ นอร์แมลเวคเตอร์ทำมุมน้อยกว่า 90 องศากับทิศทางการมองดังกล่าว ซึ่งหมายความว่าผิวหน้าทำมุมน้อยกว่า 180 องศา (ดูหัวข้อ 2.8 และรูปที่ 2.9))

การตรวจสอบดังกล่าวมีขั้นตอนวิธีแสดงในรูปที่ 3.11 โดยถือว่าข้อมูลที่เกี่ยวข้อง เก็บไว้ในลักษณะเดียวกับในขั้นตอนวิธีในรูป 3.10 ขั้นตอนวิธีนี้เป็นขั้นตอนวิธี ที่ใช้ได้ทั้งในการโปรเจคทัศนียภาพและการโปรเจคภาพแบบขนาน

```
TEST-NV:  PROCEDURE  (S,SURFACE,WCO,NV,Xf,Yf,Zf);
  /* S                  => Surface number to be tested
     SURFACE(S,1..n)    => Vertices list of surface S: pointers
     WCO(I,1..3)        => Array of world coordinates
     NV(S,1..3)         => Array of coordinates of normal vector of
                           surface S
     Xf,Yf,Zf           => World coordinate of eye point        */
  BEGIN;
    BB := SURFACE(S,2);   /* Origin of normal vector */
    A := WCO(BB,1) - Xf;
    B := WCO(BB,2) - Yf;
    C := WCO(BB,3) - Zf;
    X := WCO(BB,1) - NV(S,1);
    Y := WCO(BB,2) - NV(S,2);
    Z := WCO(BB,3) - NV(S,3);
    RETURN(A*X + B*Y + C*Z);
  END;
END TEST-NV;
```

รูปที่ 3.11 แสดงขั้นตอนวิธีสำหรับการตรวจสอบการมองเห็นได้ของผิวหน้า

โปรแกรมย่อย TEST-NV จะทำการตรวจสอบผิวหน้า S และส่งค่าที่มากกว่า 0 คืนมา ถ้าผิวหน้า S เป็นผิวหน้าที่มองเห็น การคำนวณในโปรแกรมย่อยนี้ ทำตามขั้นตอนที่กล่าวไว้ข้างต้น แต่ได้ตัดทอนให้เหลือเฉพาะส่วนสรุปที่สำคัญของการคำนวณเท่านั้น

ผลการทำงานในขั้นตอนนี้ จะทำให้เราสามารถเลือกได้ว่า ผิวหน้าด้านใดของวัตถุเป็นผิวหน้าที่จะปรากฏในภาพ และผิวหน้าด้านใดมองไม่เห็น ซึ่งจะไม่ปรากฏในภาพ ข้อมูลผิวหน้าด้านที่มองไม่เห็น จะกลายเป็นข้อมูลที่ไม่เป็นประโยชน์สำหรับการทำงานขั้นต่อ ๆ ไปอีก และสามารถตัดทิ้งออกไปจากขบวนการทำงานได้ ในแง่นี้จะเห็นได้ว่า ถ้าเราสามารถวางการทำงานขั้นตอนนี้ไว้ในขั้นตอนแรก ๆ ของระบบการสร้างภาพ เราจะลดภาระด้านเวลาและปริมาณหน่วยความจำของการทำงานขั้นต่อ ๆ ไปได้ อย่างไรก็ดี จะต้องพิจารณาถึงลักษณะการสร้างข้อมูลวัตถุ ตลอดจนลักษณะข้อมูลที่เหลือสำหรับการทำงานขั้นอื่น ๆ อีกด้วย ในขั้นตอนการโปรเจคภาพในหัวข้อก่อน ๆ นั้น จะเห็นได้ว่า ข้อมูลหลักที่ใช้คือ ข้อมูลเส้น ซึ่งก็คือข้อมูลที่แสดงรูปร่างของผิวหน้าต่าง ๆ นั่นเอง ถ้าเราวางขั้นตอนการลบผิวหน้าส่วนที่มองไม่เห็นไว้ก่อนหน้าการโปรเจคภาพแบบขนาน หรือการโปรเจคทัศนียภาพ ก็จะสามารถลดจำนวนเส้นในการทำงานของการโปรเจคภาพดังกล่าวลงได้ สิ่งที่เราจะต้องพิจารณาก็คือ

ก. ลักษณะข้อมูลวัตถุที่เกี่ยวกับเส้น และการสร้างข้อมูลดังกล่าว ซึ่งจะมีผลต่อการสร้างข้อมูลเส้นที่จะปรากฏในภาพ การตัดรายการเส้นที่จะไม่ปรากฏในภาพ เนื่องจากเป็นส่วนของผิวหน้าที่มองไม่เห็น และรายการเส้นที่เหลือที่จะมองเห็น ที่จะใช้ในการทำงานขั้นต่อ ๆ มา

ข. โครงสร้างข้อมูลเกี่ยวกับนอร์แมลเวคเตอร์ และเส้นที่แสดงผิวหน้าหนึ่ง ๆ

ลักษณะทั้งสองประการนี้เกี่ยวพันกัน และเกี่ยวพันกับขั้นตอนต่าง ๆ ในการทำงานเกี่ยวกับเส้น โดยเฉพาะในการลบส่วนที่ถูกบังของวัตถุในภาพ ดังนั้นรายละเอียดจะได้กล่าวถึงในภายหลัง

3.6 <u>การลบส่วนที่ถูกบัง</u> (Obscured line elemination)

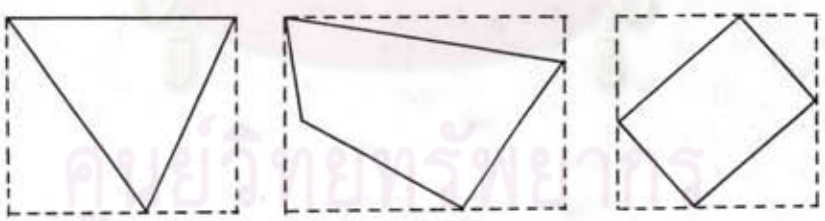
หลังจากขั้นตอนทั้งหลายในการแปลงข้อมูลพื้นฐานของวัตถุ 3 มิติ ไปเป็นข้อมูลภาพแล้ว ข้อมูลที่ได้ ยังไม่สมบูรณ์พอที่จะนำไปใช้ในขั้นตอนการแสดงภาพสำหรับวัตถุทึบได้ ทั้งนี้เพราะแม้จะได้มีการลบเส้นของผิวหน้าด้านที่มองไม่เห็นของวัตถุแต่ละชิ้นทิ้งไปแล้ว วัตถุในภาพก็ยังอาจเกิดการทับเกยกันหรือซ้อนกันได้ ซึ่งถ้านำไปแสดงภาพเลย ก็อาจเกิดภาพที่ผิดธรรมชาติ เพราะเส้นส่วนที่ถูกวัตถุอื่นบังยังปรากฏอยู่

ในการมองวัตถุจริงด้วยตาของมนุษย์ แสงจากส่วนต่าง ๆ ของวัตถุที่เข้ามาสู่ตา อาจถูกขวางด้วยส่วนของวัตถุชิ้นอื่น ๆ ทำให้เราไม่สามารถมองเห็นส่วนที่ถูกบังได้ แต่ในระบบคอมพิวเตอร์นั้น การโปรเจควัตถุลงในภาพ ไม่มีการลบส่วนที่ถูกบังของวัตถุโดยอัตโนมัติแบบสายตา ทุก ๆ ส่วนของวัตถุ สามารถปรากฏในภาพได้หมด ดังนั้น เราจึงจำเป็นต้องมีขบวนการที่จะทำการลบส่วนที่ถูกบังอีกขั้นตอนหนึ่ง ก่อนที่จะนำข้อมูลไปแสดงเป็นภาพที่สมบูรณ์

กรรมวิธีขั้นตอนนี้ เริ่มด้วยการตรวจสอบว่าผิวด้านของวัตถุคู่ใด ๆ บังกันหรือไม่ ถ้าปรากฏว่ามีการบังกัน ซึ่งอาจจะเป็นการบังกันบางส่วน (ทับเกยกันในภาพ) หรือซ้อนกันก็ได้ ก็จะต้องมีวิธีการตรวจสอบว่าวัตถุใดถูกบัง และส่วนใดของวัตถุนั้น เป็นส่วนที่จะต้องลบทิ้ง การทำงานของขั้นตอนนี้ ทั้งหมดเป็นการทำงานที่มีความซับซ้อนและยุ่งยากมากที่สุด และยังมีผลต่อการสร้างโครงสร้างข้อมูล ซึ่งเกี่ยวพันกับงานขั้นตอนอื่น ๆ ด้วย อย่างไรก็ดี จะได้กล่าวเป็นลำดับไปดังนี้

3.6.1 การทดสอบการบังกันของวัตถุ

เรายังจำเป็นจะต้องพิจารณาวัตถุในลักษณะเป็นผิวด้าน เพราะเส้นของวัตถุที่จะถูกบังนั้นก็โดยผิวด้านอื่น ๆ ของวัตถุ วิธีการคือ ทำการตรวจสอบว่ามีผิวด้าน (ซึ่งมองเห็น) ใด ๆ ที่ทับเกย



รูปที่ 3.12 แสดงกรอบอาณาเขตของผิวด้าน (เส้นประ)

กันหรือซ้อนกันหรือไม่ โดยการพิจารณาตามแกน X และ Y ของภาพ ถ้าผิวด้านคู่ใดไม่เกยกันในแกน X และ Y แสดงว่าผิวด้านคู่นั้นไม่บังกัน การตรวจสอบขั้นนี้ ใช้วิธีการที่เรียกว่า Minimax test นั่นคือ พิจารณาค่าที่มากที่สุด และน้อยที่สุดในแกน X หรือ Y ของแต่ละผิวด้าน นั่นคือ ถ้า X ที่มากที่สุดของผิวด้านหนึ่ง น้อยกว่าค่า X ที่น้อยที่สุดของอีกผิวด้านหนึ่ง ผิวด้านทั้ง 2 จะไม่บังกัน

ในแกน Y ก็พิจารณาเช่นเดียวกัน การใช้ Minimax test ในแกน X และ Y ร่วมกัน เรียกว่า Bounding-Box Test ซึ่งเปรียบเสมือนกำหนดให้ผิวด้านแต่ละด้านล้อมรอบด้วยกรอบสี่เหลี่ยม ซึ่งเรียกว่ากรอบอาณาเขต ของผิวด้าน (ดูรูปที่ 3.12)

การบังกันของวัตถุแต่ละชิ้นในภาพ (โดยพิจารณาวัตถุทั้งชิ้น เป็นพื้นที่ผิวด้าน 1 อัน) เกิดได้ในกรณีต่าง ๆ ดังนี้ คือ

กรณีที่ 1 กรอบอาณาเขตไม่เกยหรือซ้อนกัน
ในแกน X
ผิวด้านไม่บังกัน

กรณีที่ 2 กรอบอาณาเขตไม่เกยหรือซ้อนกัน
ในแกน Y
ผิวด้านไม่บังกัน

กรณีที่ 3 กรอบอาณาเขตทับเกยกัน
ในแกน X และ Y
แต่ผิวด้านไม่บังกัน

กรณีที่ 4 กรอบอาณาเขตทับเกยกัน
ในแกน X และ Y
ผิวด้านเกยกัน

กรณีที่ 5 กรอบอาณาเขตซ้อนกัน
และผิวด้านเกยกัน

กรณีที่ 6 กรอบอาณาเขตซ้อนกัน
และผิวด้านซ้อนกัน

กรณีที่ 7 กรอบอาณาเขตซ้อนกัน
แต่ผิวด้านไม่บังกัน

ในการลบส่วนที่ถูกบังของวัตถุทั้งหลายในภาพนั้น วัตถุแต่ละชิ้นจะต้องทำการตรวจสอบกับวัตถุชิ้นอื่นทุกชิ้นในภาพ ดังนั้น ถ้ากล่าวอย่างพื้น ๆ โปรแกรมในขั้นตอนการทดสอบนี้ จะต้องประกอบด้วยวงจรการทำงาน (loop) 2 ชั้น ถ้ามีวัตถุในภาพ n ชิ้น ก็จะต้องมีการทดสอบเป็นจำนวน $\frac{n.\ (n-1)}{2}$ รอบ เช่น วัตถุ 100 ชิ้น จะจับคู่ในการทดสอบได้ถึง 4,950 คู่ วัตถุ 200 ชิ้น จะเป็น 19,900 คู่ วัตถุ 1000 ชิ้น จะเป็น 495,000 คู่ เป็นต้น ซึ่งจะเห็นว่ายิ่งจำนวนวัตถุมากขึ้นเท่าใด เวลาที่ใช้ในการทดสอบยิ่งเพิ่มขึ้นมากมหาศาล ส่วนที่จะช่วยเพิ่มประสิทธิภาพการทำงานในขั้นตอนนี้ ส่วนแรกก็คือ การใช้กรอบอาณาเขตของผิวด้านของวัตถุ หรือของวัตถุทั้งชิ้น ซึ่งจะช่วยให้การตรวจสอบทำได้เร็วขึ้น จะเห็นได้ว่า ความสัมพันธ์ของกรอบอาณาเขตในกรณีที่ 1 และ 2 สามารถตรวจสอบได้โดยง่าย และถ้าพบกรณีดังกล่าวก็ถือว่า เสร็จสิ้นขบวนการตรวจสอบและลบเส้นระหว่างวัตถุคู่นั้น ๆ ไปได้เลย เนื่องจากไม่บังกัน อย่างไรก็ดี เราจำเป็นต้องมีส่วนการทำงานที่จะจัดลำดับค่าจุดพิกัดต่าง ๆ ของแต่ละผิวด้าน เพื่อให้ได้ค่ามากที่สุด และน้อยที่สุดตามแกน X และ Y ของผิวด้านหรือวัตถุแต่ละชิ้นไว้แล้วก่อนหน้านี้ ซึ่งจะมีอยู่ 4 ค่า ต่อผิวด้าน

หรือวัตถุแต่ละชั้น

ส่วนที่จะช่วยเพิ่มประสิทธิภาพการทำงานขั้นตอนการลบส่วนที่ถูกบังอีกส่วนหนึ่งก็คือ การจัดเรียงลำดับของผิวด้านหรือวัตถุ เนื่องจากการทำงานจะประกอบด้วยวงจร 2 ชั้น ดังกล่าวแล้ว การจัดเรียงลำดับผิวด้านหรือวัตถุ จะช่วยให้สามารถหลุดจากวงจรชั้นในได้เร็วขึ้น โดยอาศัยการตรวจสอบกรอบอาณาเขต นั่นคือ สมมุติ A เป็นวัตถุที่พิจารณาจากวงจรชั้นนอก โดยที่วัตถุทั้งหมดมีจำนวน n ชิ้น และ B เป็นวัตถุจากวงจรชั้นใน ที่จะนำมาทำการตรวจสอบกับวัตถุ A ถ้าเราจัดเรียงลำดับวัตถุทั้งหมดจากน้อยไปหามากด้วยค่ามาก และน้อยที่สุดตามแกน X ของวัตถุแต่ละชิ้น และวงจรแต่ละชั้นหยิบวัตถุตามลำดับคือ A ตั้งแต่ 1 ถึง n-1 และ B ตั้งแต่ A+1 ถึง n เราสามารถจบวงจรชั้นในได้ทันทีที่พบว่า กรอบอาณาเขตของวัตถุ B ไม่ทับเกยกับกรอบอาณาเขตของวัตถุ A (กรณีที่ 1) ซึ่งทำให้ไม่ตรวจไปจนถึง Bn

อีกส่วนหนึ่งที่อาจช่วยเพิ่มประสิทธิภาพการทำงานก็คือ การพิจารณาในระดับวัตถุแทนผิวด้าน วัตถุชิ้นหนึ่ง ๆ ในภาพ อาจประกอบด้วยผิวด้านหลาย ๆ ด้าน ซึ่งทำให้จำนวนผิวด้านที่ต้องมาจับคู่ในการทดสอบมีมากกว่าจำนวนวัตถุ ดังนั้น ถ้าทำการทดสอบโดยพิจารณาวัตถุทั้งชิ้นแทนก็จะลดปริมาณการทดสอบลงได้ อย่างไรก็ดี วิธีการนี้ เราจำเป็นต้องระบุได้ว่า เส้นใดเป็นเส้นรอบรูปของภาพวัตถุ และเส้นใดเป็นเส้นภายใน ซึ่งจะต้องกระทำมาก่อนหน้างานในขั้นตอนนี้

สำหรับขั้นตอนในการตรวจสอบและลบส่วนที่ถูกบังของวัตถุ มีดังนี้คือ

1. ตรวจสอบกรอบอาณาเขตของผิวด้านหรือวัตถุแต่ละคู่ ในกรณีที่ 1 และ 2 ถ้าพบแสดงว่าไม่มีการบังกัน และเปลี่ยนคู่ได้เลย

2. ถ้าการตรวจสอบในข้อ 1 ไม่พบ จะมีทางเป็นไปได้อีก 2 ลักษณะคือ กรอบอาณาเขตเกยกันบางส่วน (กรณีที่ 3 และ 4) หรือกรอบอาณาเขตซ้อนกัน (กรณีที่ 5,6 และ 7) ในขั้นตอนนี้เราทำโดยการตรวจว่าเส้นรอบรูปของวัตถุ หรือผิวด้านทั้ง 2 ตัดกันหรือไม่ ถ้าตัดกัน แสดงว่าวัตถุเกยกัน และ ณ จุดที่ตัดกัน เราเปรียบเทียบความลึกได้ ซึ่งทำให้บอกได้ว่าวัตถุหรือผิวด้านใดอยู่ลึกกว่า แล้วทำการลบเส้นที่ถูกบังของวัตถุชิ้นนั้น (กรณีที่ 4 และ 5)

3. ถ้าไม่ปรากฏว่ามีเส้นรอบรูปคู่ใดตัดกัน และกรอบอาณาเขตเกยกันเพียงบางส่วน แสดงว่าวัตถุไม่บังกัน ผ่านได้เลย (กรณีที่ 3)

4. ถ้าเส้นรอบรูปไม่ตัดกัน แต่กรอบอาณาเขตซ้อนกัน เราต้องตรวจสอบอีกชั้นว่าวัตถุซ้อนกันหรือไม่ ถ้าไม่แสดงว่าวัตถุไม่บังกัน (กรณีที่ 7)

5. ถ้าวัตถุซ้อนกัน ตรวจสอบว่าชิ้นใดอยู่ลึกกว่า ถ้าชิ้นเล็กอยู่ลึกกว่า ก็ลบไปทั้งชิ้น ถ้าชิ้นเล็กอยู่ใกล้กว่า เราสามารถผ่านไปเลย เพราะแม้วัตถุชิ้นเล็กจะบังชิ้นใหญ่ แต่เส้นรอบรูปของวัตถุทั้งสอง จะไม่ถูกกระทบกระเทือน และในภาพแบบลายเส้นเราสนใจเฉพาะเส้นเท่านั้น (กรณีที่6) อย่างไรก็ดี ถ้าเราพิจารณาวัตถุทั้งชิ้นแทนผิวด้าน เราจำเป็นต้องคำนึงถึงเส้นภายในเพราะวัตถุชิ้นเล็กที่บังอยู่ข้างหน้า อาจทำให้เส้นภายในของวัตถุชิ้นใหญ่ถูกบังด้วย ซึ่งต้องทำการตรวจสอบเพื่อลบเส้นภายในส่วนที่ถูกบังด้วย

3.6.2 การตรวจผิวด้านหรือวัตถุที่เกยกัน

ขั้นตอนนี้ คือการตรวจสอบว่าเส้นรอบรูปของผิวด้านหรือวัตถุคู่หนึ่งตัดกัน ณ จุดใด หรือถ้ามีการตัดกัน แสดงว่าผิวด้านหรือวัตถุคู่นั้นเกยกัน (บังกันบางส่วน) การทำงานประกอบด้วยวงจร 2 ชั้น เช่นเดียวกัน เพื่อตรวจสอบการตัดกันของเส้นแต่ละเส้นของผิวด้านหรือวัตถุชิ้นหนึ่งกับแต่ละเส้นของวัตถุอีกชิ้นหนึ่ง ถ้าวัตถุทั้งสองชิ้นมีเส้นรอบรูปจำนวน n และ m เส้นตามลำดับจำนวนการจับคู่ในการตรวจสอบมากที่สุดคือ $n \times m$ คู่ แต่อาจน้อยกว่าได้เมื่อพบการตัดกันระหว่างเส้นคู่ใด ๆ ซึ่งทำให้จบการตรวจสอบได้เลย ถ้าตรวจสอบครบทุกคู่แล้วไม่ปรากฏว่ามีการตัดกัน แสดงว่าวัตถุทั้งสองอาจซ้อนกันอยู่ หรือไม่บังกันเลย

ลักษณะการทำงานอาศัยค่าจุดพิกัดX และ Y ของจุดปลายทั้งสองของเส้นทั้งสอง มาทำการคำนวณหาจุดตัดของเส้น ซึ่งจะตัดกันจริงก็ต่อเมื่อค่า X,Y ของจุดตัดอยู่ในพิสัยค่าจุดพิกัดของปลายทั้งสองของเส้นทั้งสอง และโดยการเปรียบเทียบความลึก ณ จุดตัดของเส้นทั้งสอง ก็จะทราบได้ว่าวัตถุใดอยู่หน้าวัตถุใดอยู่หลัง ในกรณีที่ค่าความลึกเท่ากัน เราจะไม่สามารถกำหนดลำดับความลึกของวัตถุได้ แต่เส้นรอบรูปตัดกันจะมีจุดตัดเป็นจำนวนคู่เสมอ เราสามารถข้ามไปหาจุดตัดอื่นได้

อย่างไรก็ดี การใช้สูตรโดยตรงอาจทำให้เกิดการหารหรือคูณด้วย 0 ได้ ดังนั้นการพัฒนาแอลกอริทึม จึงต้องคำนึงถึงข้อนี้ กรณีที่จะเกิดขึ้นได้ก็คือ เมื่อเส้นคู่ที่ตรวจสอบขนานกันหรือเส้นใดเส้นหนึ่ง (หรือทั้งสองเส้น) ขนานกับแกน x หรือ y

การตรวจผิวด้านและวัตถุทับเกยกัน มีขั้นตอนวิธีดังรูปที่ 3.13 และ 3.14 โปรแกรมย่อย ØVERLAP-TEST ที่แสดงในรูปที่ 3.13 เป็นส่วนที่ควบคุมวงจรการตรวจสอบเส้นรอบรูปของวัตถุ A และ B ค่าพิกัดที่เกี่ยวข้องคือ ค่าพิกัดจอภาพซึ่งเป็นข้อมูล 3 มิติ ซึ่งอยู่ในชุดข้อมูล SCØ โดยที่ SCØ (LINE(1,1),1..3) คือค่าจุดพิกัดในแกน X,Y,Z ของจุดปลายที่ 1 ของเส้นที่ 1 และ SCØ(LINE(1,2),1 ..3) คือค่าพิกัดจุดปลายที่สอง ØVERLAP-TEST จะส่งค่าดังกล่าวของเส้นซึ่งระบุด้วย LB และ LA ไปให้โปรแกรมย่อย CRØSSLINE ทำการคำนวณหาจุดตัดของเส้น เมื่อพบว่าตัดกัน ก็จะทำการคำนวณเปรียบเทียบความลึก แล้วส่งค่าพิกัดของจุดตัด (x,y) พร้อมกับวัตถุที่ถูกบังในตัวแปร ØVERLAPPED-ØBJECT คืนไปให้โปรแกรมหลัก โปรแกรมย่อย CRØSSLINE ซึ่งแสดงในรูป 3.14 จะทำการตรวจหาจุดตัดของเส้นแต่ละคู่ ถ้าเส้นทั้งคู่ขนานกันหรือจุดตัดอยู่นอกพิสัยของเส้นใดเส้นหนึ่ง (จุดตัดภายนอก) จะส่งค่า false คืนให้ OVERLAP-TESTในตัวแปร CRØSS ถ้าจุดตัดเป็นจุดภายใน CRØSS จะมีค่าเป็น true และส่งคืนให้พร้อมกับค่าพิกัด X,Y ของจุดซึ่งเส้นทั้งสองตัดกัน ซึ่งกรณีหลังหมายถึงเส้นรอบรูปของวัตถุ หรือผิวด้านที่พิจารณาทับเกยกัน

3.6.3 การลบเส้นรอบรูปที่ถูกบัง

เมื่อได้ตรวจพบการตัดกันของเส้นรอบรูปของผิวด้านหรือวัตถุแล้ว เส้นรอบรูปของวัตถุหรือผิวด้านที่อยู่ลึกกว่า จะถูกบังไปบางส่วน เส้นหรือส่วนของเส้นรอบรูปของวัตถุ หรือผิวด้านส่วนดังกล่าวจะต้องถูกลบออกก่อนจะแสดงภาพได้ ในการลบเส้นรอบรูปที่ถูกบังนั้น เริ่มจากจุดที่ตัดกันของเส้นรอบรูปจุดหนึ่ง ไปจนถึงจุดตัดอีกจุดหนึ่ง ดังตัวอย่างในรูปที่ 3.15 เส้นรอบรูปของวัตถุ ABCDEF ถูกบังด้วยเส้นรอบรูปของวัตถุ LMNOPQ จุดตัดของเส้นรอบรูปทั้งสองคือ S และ T ตามภาพนี้ เส้น SF,FE และ ET คือส่วนที่ถูกบังและจะต้องลบทิ้ง ในการตรวจการตัดกันของเส้นรอบรูปตามหัวข้อย่อย 3.6.2 นั้น เราจะพบจุด S หรือ T จุดใดจุดหนึ่งก่อน สมมุติเราพบจุด S เราก็จะทราบว่า S เกิดจากการตัดกันของเส้น AF และ MN และจากการตรวจลำดับความลึก ณ จุด S ตามที่

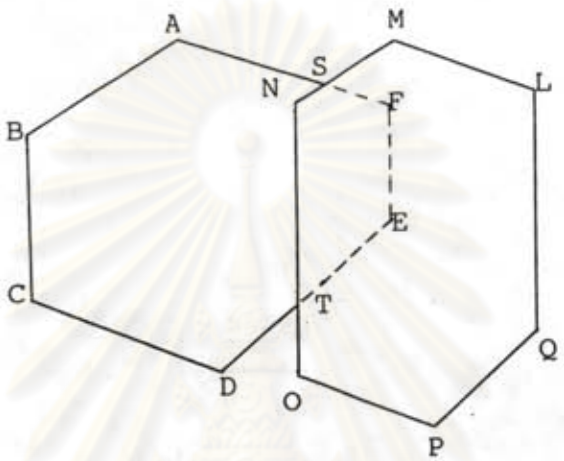
```
OVERLAP_TEST:  PROCEDURE  (A,B,SCO,LINE,LA,LB,X,Y,
                                    OVERLAPPED-OBJECT);
  BEGIN;
    A1 := 1st line number of object A;
    A2 := last line number of object A;
    DO LA := A1 TO A2;
      Xa := SCO(LINE(LA,1),1); /* 1st end of line LA */
      Ya := SCO(LINE(LA,1),2);
      Xb := SCO(LINE(LA,2),1); /* 2nd end of line LA */
      Yb := SCO(LINE(LA,2),2);
      B1 := 1st line number of object B;
      B2 := last line number of object B;
      DO LB := B1 TO B2;
        Xp := SCO(LINE(LB,1),1); /* 1st end of line LB */
        Yp := SCO(LINE(LB,1),2);
        Xq := SCO(LINE(LB,2),1); /* 2nd end of line LB */
        Yq := SCO(LINE(LB,2),2);
        CALL CROSSLINE(Xa,Ya,Xb,Yb,Xp,Yp,Xq,Yq,X,Y,CROSS);
        IF CROSS THEN
          DO;
            Za := SCO(LINE(LA,1),3); /* Depths of LA */
            Zb := SCO(LINE(LA,2),3);
            Zp := SCO(LINE(LB,1),3); /* Depths of LB */
            Zq := SCO(LINE(LB,2),3);
              /* Find depth at X,Y of LA */
            ZA := (Zb -Za)*(X - Xa)/(Xb -Xa) + Za;
              /* Find depth at X,Y of LB */
            ZB := (Zq - Zp)*(X - Xp)/(Xq - Xp) + Zp;
            IF ZA < > ZB THEN
              DO;
                IF ZA > ZB THEN
                     OVERLAPPED-OBJECT := A;
                ELSE OVERLAPPED-OBJECT := B;
                RETURN;
                /* Overlapping found found on LA & LB at X,Y */
              END;
          END;
      END;
    END;
    OVERLAPPED-OBJECT := 0;
    RETURN;  /* No overlapping found on object A & B */
  END;
END OVERLAP_TEST;
```

รูปที่ 3.13 แสดงขั้นตอนวิธีสำหรับการตรวจการทับเกยของผิวด้านหรือวัตถุ

```
CROSSLINE:  PROCEDURE  (Xa,Ya,Xb,Yb,Xp,Yp,Xq,Yq,X,Y,CROSS);
  X1 := Xa - Xb;
  Y1 := Ya - Yb;
  X2 := Xp - Xq;
  Y2 := Yp - Yq;
  A := Y1/X1;
  B := Y2/X2;
  CROSS := FALSE;
  IF (X1=0 & X2=0) | (Y1=0 & Y2=0) | (A=B & A< >0) THEN
    RETURN;   /* Parallel lines */
  IF Z1 = 0 THEN
    DO;
      X := Xa;
      IF Y2 = 0 THEN
           Y := Yp;
      ELSE Y := B*(X - Xp) + Yp;
    END;
  ELSE
    IF X2 = 0 THEN
      DO;
        X := Xp;
        IF Y1 = 0 THEN
             Y := Ya;
        ELSE Y := A*(X - Xa) + Ya;
      END;
    ELSE
      IF Y1 = 0 THEN
        DO;
          Y := Ya;
          X := (Y - Yp)/B + Xp;
        END;
      ELSE
        IF Y2 = 0 THEN
          DO;
            Y := Yp;
            X := (Y - Ya)/A + Xa;
          END;
        ELSE
          DO;
            X := (Yp - Ya + A*Xa - B*Xp)/(A - B);
            Y := A*(X - Xa) + Ya;
          END;
  IF (Xa-X)*(X-Xb)*(Xp-X)*(X-Xq) THEN
    CROSS := TRUE;
  RETURN;
END CROSSLINE;
```

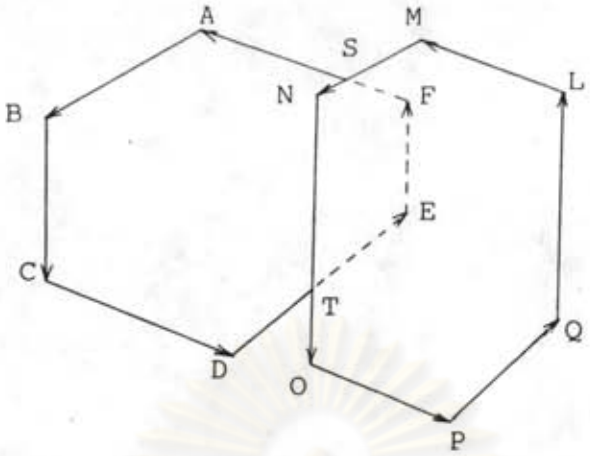
รูปที่ 3.14 แสดงขั้นตอนวิธี การตรวจและคำนวณหาจุดตัดของเส้น

กล่าวมาแล้ว เราก็ทราบว่าวัตถุ ABCDEF อยู่ลึกกว่า และเป็นวัตถุที่จะต้องลบเส้นส่วนที่ถูกบัง ถ้าเราทำการตรวจจุดตัดของเส้นรอบรูปต่อไป เราก็จะพบว่าเส้น DE และ NO ตัดกันที่จุด T เราบอกได้ทันทีว่าเส้นรอบรูปตั้งแต่จุด S ต่อเนื่องไปเรื่อย ๆ จนถึง จุด T (ตามเส้นประ) จะเป็นส่วนที่ถูกลบ



รูปที่ 3.15 แสดงลักษณะเส้นรอบรูปที่ถูกบัง

ปัญหาก็คือ เส้นรอบรูประหว่างจุด S และ T มี 2 ชีก การทำงานของคอมพิวเตอร์จะรู้ได้อย่างไรว่าชีกใดคือ ชีกที่ถูกบังอยู่ (ในตัวอย่างนี้คือ SFET) ซึ่งจะต้องถูกลบ ในการแก้ปัญหาข้อนี้ เราจำเป็นต้องกำหนดทิศทางของเส้นรอบรูปทุกอันให้อยู่ในรูปแบบเดียวกัน ในตัวอย่างรูปที่ 3.15 นั้น ถ้าเรากำหนดทิศทางของเส้นรอบรูปไปทางเดียวกันตลอด คือ ตามลำดับตัวอักษร ซึ่งเป็นทิศทางทวนเข็มนาฬิกา (ดังรูปที่3.16) ทิศทางดังกล่าวจะช่วยให้การตรวจสอบทำได้เป็นอย่างดีว่าส่วนใดแน่ที่จะถูกลบ จะเห็นได้ว่าถ้าเส้นรอบรูปมีทิศทางทวนเข็มนาฬิกาแล้ว เมื่อเรามองตามทิศทางของเส้นแต่ละเส้น ที่ประกอบเป็นเส้นรอบรูป เราบอกได้ทันทีว่า ส่วนที่อยู่ทางด้านซ้ายของทิศทางหนึ่ง ๆ คือ ส่วนภายในอาณาเขตเส้นรอบรูป และส่วนที่อยู่ทางขวา คือ ส่วนที่อยู่ภายนอก ดังนั้น ส่วนใด ๆ ของวัตถุที่ถูกบังที่อยู่ภายในอาณาเขตเส้นรอบรูปของวัตถุที่บังมันอยู่ จะเป็นส่วนที่ถูกลบ เช่น ส่วนของเส้น AF ที่อยู่ด้านซ้ายของทิศทางของเส้น MN จะต้องถูกลบ ในขณะเดียวกัน ทิศทางของเส้นก็จะระบุตัวมันเอง ณ จุดตัดของเส้นรอบรูป ว่าส่วนใดของเส้นอยู่ด้านซ้ายหรือขวาของเส้นรอบรูปที่บังมันอยู่ เช่น เส้น AF มีทิศทางไปทาง A คือ พุ่งออกไปทางด้านขวาของ MN ดังนั้น ส่วนของเส้น

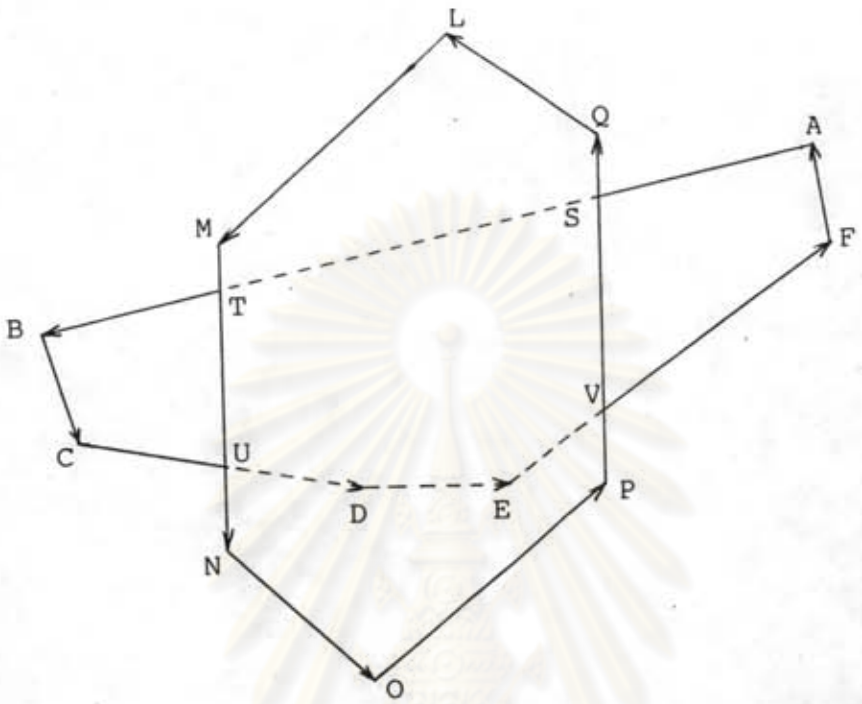


รูปที่ 3.16 แสดงเส้นรอบรูปที่ระบุทิศทางไว้ด้วย

AF ที่อยู่ด้านซ้ายของ MN ก็คือ SF เป็นส่วนที่ถูกบังและส่วนด้านขวาคือ SA ไม่ถูกบัง

การทำงานของขั้นตอนการลบเส้นรอบรูปที่ถูกบัง จะเริ่มที่เส้นของวัตถุที่ถูกบัง ซึ่งตรวจพบจุดตัดจุดแรก ในตัวอย่างนี้คือ เส้น AF ตัด MN ณ จุด S จากการตรวจทิศทางของเส้น พบว่าส่วนต้นของ AF คือ FS ถูกบัง และ SA ไม่ถูกบัง การทำงานจะต้องตรวจสอบเรียงไปตามลำดับเส้นรอบรูปที่ต่อเนื่อง คือ จาก FA (ตามทิศทางของเส้น) ไปยัง AB BC CD ฯลฯ จนถึงเส้นสุดท้าย คือ EF การทำงานคือ ตรวจเส้นเหล่านี้ทีละเส้นว่า ตัดกับเส้นรอบรูปของวัตถุที่บังหรือไม่ ถ้าไม่เส้นนั้นจะมีสถานะเหมือนปลายเส้นเส้นแรก (ในกรณีนี้คือ ไม่ถูกบัง) ตามตัวอย่างคือ AB BC CD เป็นเส้นที่มีสถานะเดียวกับ SA เมื่อตรวจถึง DE จะพบว่าตัดกับNO ที่ T การพบจุดตัดใหม่จะทำให้เปลี่ยนสถานะ DE จะถูกแบ่งเป็น 2 ส่วน คือ DT และ TE DT จะมีสถานะเหมือนปลายเส้นก่อนหน้ามัน (ไม่ถูกบัง) ปลายเส้น DE คือ TE จะเปลี่ยนสถานะเป็นถูกบัง จากนั้นก็ตรวจต่อไปเรื่อย ๆ โดยลักษณะเดียวกัน เส้นต่อไปคือ EF ก็จะต้องมีสถานะเหมือนปลายเส้นก่อนหน้ามัน (TE) คือถูกบัง การทำงานจะดำเนินไปในลักษณะนี้ ไม่ว่าจะพบจุดตัดอีกกี่จุดก็ตาม จนกว่าจะครบทุกเส้น (ในที่นี้ EF เป็นเส้นสุดท้าย)

ตัวอย่างต่อไปนี้ แสดงการลบเส้นรอบรูปโดยกำหนดสถานะของเส้นที่ต่อเนื่องกัน โดยกำหนดให้เส้นรอบรูปมีทิศทางทวนเข็มนาฬิกา (ดูรูป3.17) และจากการตรวจการตัดกันของ



รูปที่ 3.17 แสดงตัวอย่างการลบเส้นรอบรูป

เส้นรอบรูปพบว่า CD ตัด MN ที่จุด U และวัตถุ ABCDEF ถูกบัง การทำงานจะเป็นดังนี้:-

ลำดับที่		ผล : เส้นถูกบัง	เส้นไม่ถูกบัง
1.	ตรวจทิศทางของเส้น CD และ MN		CU
2.	เส้น CD ไม่ตัด LMNOPQ ที่อื่นอีก	UD	
3.	เส้นถัดไป DE ไม่ตัด LMNOPQ	DE	
4.	เส้นถัดไป EF ตัด PQ ที่ V	EV	
5.	EF ไม่ตัด LMNOPQ ที่อื่นอีก		VF
6.	เส้นถัดไป FA ไม่ตัด LMNOPQ		FA
7.	เส้นถัดไป AB ตัด PQ ที่ S		AS
	และ AB ตัด MN ที่ T แต่ S ใกล้ A มากกว่า	ST	
8.	ไม่ตัด LMNOPQ ที่อื่นอีก		TB
9.	เส้นถัดไป BC ไม่ตัด LMNOPQ		BC
10.	ครบทุกเส้นแล้ว		

จากตัวอย่างนี้ จะเห็นได้ว่าลักษณะของการทำงานมีข้อต้องคำนึงดังนี้

ก. เส้นรอบรูปจะต้องมีทิศทางไปในทางเดียวกัน

ข. การตรวจเส้นต้องกระทำไปตามลำดับเส้นที่ต่อเนื่องกัน

ค. ทุกครั้งที่พบจุดตัดของเส้น จะแบ่งเส้นเป็น 2 ส่วน ส่วนต้นของเส้นจะมีลักษณะเหมือนส่วนปลายของเส้นก่อนหน้ามัน และส่วนปลายของเส้นจะเปลี่ยนลักษณะเป็นตรงข้าม

ง. เส้นที่ไม่ตัดเส้นรอบรูปที่บังมัน จะมีสถานะเหมือนปลายเส้นก่อนหน้ามัน

จ. เส้นหนึ่ง ๆ อาจตัดเส้นรอบรูปที่บังมันได้มากกว่า 1 จุด จะต้องพิจารณาจากจุดที่อยู่ใกล้จุดต้นของเส้นก่อนตามลำดับ

ข้อพิจารณาเหล่านี้ ครอบคลุมไปถึงการบริหารข้อมูล และการจัดข้อมูลเกี่ยวกับเส้นด้วย ซึ่งจะเห็นได้ว่า ข้อมูลเกี่ยวกับเส้นรอบรูป จะต้องจัดไว้อย่างเหมาะสมตั้งแต่เริ่มแรก คือ ต้องแสดงความสัมพันธ์ในเชิงต่อเนื่องของเส้นแต่ละเส้น ซึ่งเป็นเส้นรอบรูป ในขณะเดียวกัน ข้อมูลจุดต้นและปลายของแต่ละเส้น ก็จะต้องจัดไว้ในลักษณะที่ระบุทิศทางได้ เราจะพบว่า ขั้นตอนการทำงานต่างๆ ก่อนหน้าการลบส่วนที่ถูกบังของวัตถุ ก็มีส่วนที่ทำงานกับข้อมูลเส้น ได้แก่ การสร้างข้อมูลเส้นของวัตถุ (ซึ่งยังไม่ได้กล่าวถึง) การขลิบขอบภาพ ซึ่งมีผลต่อการเปลี่ยนแปลงข้อมูลเกี่ยวกับเส้น เป็นต้น การทำงานของขั้นตอนเหล่านั้น จะต้องไม่ทำให้ลักษณะการสร้างความสัมพันธ์ของข้อมูลเกี่ยวกับเส้นเสียหายไป สิ่งหนึ่งที่จะต้องคำนึงก็คือ ข้อมูลเส้นรอบรูปจะต้องมีครบเป็นเส้นรอบรูปจริง ๆ มิฉะนั้น การพัฒนาแอลกอริทึมในขั้นตอนนี้จะยุ่งยาก หรือเป็นไปไม่ได้เลย ในขณะนี้เราสามารถสรุปเกี่ยวกับการทำงานกับข้อมูลเส้นตั้งแต่ต้นได้ดังนี้

ก. การสร้างข้อมูลพื้นฐานของวัตถุ จะต้องสร้างข้อมูลเส้นอย่างเป็นระบบ เพื่อให้ความสัมพันธ์ของเส้น และทิศทางของเส้นระบุได้อย่างถูกต้อง

ข. ขั้นตอนการทำงานต่าง ๆ ต้องไม่ทำให้ความสัมพันธ์ดังกล่าว ถูกทำลายลงก่อนขั้นตอนการลบส่วนถูกบังของวัตถุ ซึ่งเป็นขั้นตอนสุดท้ายก่อนการนำไปแสดงภาพ

ค. ขั้นตอนการขลิบขอบภาพ อาจทำให้วัตถุบางชิ้นในภาพขาดหายไปบางส่วน ซึ่งทำให้เส้นรอบรูปไม่ครบถ้วน วิธีแก้คือ ใช้เส้นขอบภาพในส่วนที่ถูกขลิบของวัตถุในภาพแทนเส้นรอบรูปส่วนที่ขาดหายไป (ความลึกของเส้นรอบรูปที่ใช้เส้นขอบภาพแทนไม่มีผลต่อการทำงานขั้นตอนนี้ เพราะไม่มีเส้นใด ๆ ที่จะตัดเส้นดังกล่าวได้ เพราะถ้าวัตถุ 2 ชิ้น ถูกขลิบในส่วนเดียวกันของขอบภาพ ถ้าวัตถุทับเกยกัน เราจะตรวจพบการตัดกัน ณ จุดอื่น ๆ ของเส้นรอบรูปที่แท้จริง และถ้าไม่พบก็มีอยู่เพียง 2 กรณีที่เป็นไปได้ คือ วัตถุซ้อนกันหรือไม่บังกันเลย) อย่างไรก็ดี เส้นขอบภาพที่นำมาใช้เป็นเส้นรอบรูปก็เพียงเพื่อให้การทำงานไปตามลำดับต่อเนื่องของเส้นรอบรูปเป็นไปได้ แต่เส้นดังกล่าว จะต้องระบุไว้ว่าไม่ใช่ส่วนที่จะนำไปแสดงภาพ

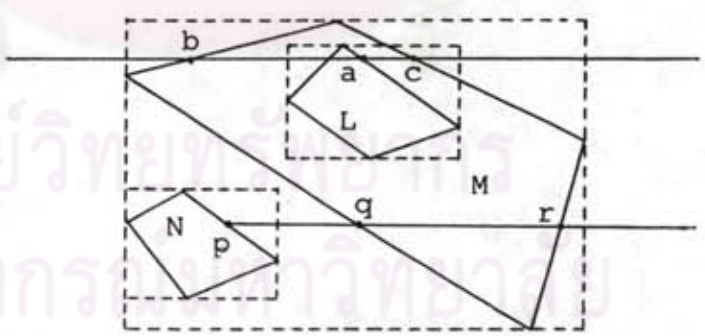
ข้อพิจารณาสำคัญอีกประการหนึ่งก็คือ ในการลบส่วนที่ถูกบังของวัตถุนั้น วัตถุหรือผิวด้านแต่ละชิ้น จะต้องทำการตรวจสอบการบังกันกับวัตถุอื่น ๆ จนครบทุกชิ้นก่อน จึงจะได้เป็นข้อมูลสมบูรณ์

เมื่อตรวจสอบกับวัตถุชิ้นต่าง ๆ นั้น แต่ละครั้ง เส้นรอบรูปของวัตถุจะต้องครบถ้วนเสมอ ดังนั้น เมื่อตรวจพบว่าถูกบังด้วยวัตถุอื่นชิ้นหนึ่ง ๆ การลบเส้นที่ถูกบัง จะต้องไม่ใช่การลบข้อมูลเส้นส่วนที่ถูกบังออกไปจริง ๆ แต่จะต้องเป็นการกำหนด รหัสกำกับเส้นไว้ว่า เส้นใด ๆ เป็นเส้นที่ถูกบัง เส้นใดไม่ถูกบัง เมื่อตรวจพบว่ามีวัตถุชิ้นอื่น ๆ อีกที่บังมัน เส้นทั้ง 2 ชนิด จะต้องถูกนำมาใช้อีก เพียงแต่ว่าเส้นใดที่มี รหัสว่าถูกบังไปแล้ว ไม่ต้องกำหนดสถานะใหม่อีก

ข้อพิจารณาข้อสุดท้าย เป็นเรื่องเกี่ยวกับข้อมูล ก็คือ จำนวนข้อมูลและค่าจุดพิกัดของวัตถุแต่ละชิ้นจะเปลี่ยนแปลงไป และไม่สามารถกำหนดแน่นอนได้เลย ข้อพิจารณาข้อนี้ เป็นปัจจัยสำคัญข้อหนึ่งในการพัฒนาระบบโครงสร้างข้อมูลของระบบงานสร้างภาพวัตถุ 3 มิติ

3.6.4 การตรวจการซ้อนกันของวัตถุในภาพ

ในกรณีที่กรอบอาณาเขตของผิวด้านหรือวัตถุซ้อนกัน แต่เส้นรอบรูปไม่ตัดกัน จะต้องมีขั้นตอนการตรวจสอบในอีกลักษณะหนึ่ง เนื่องจากมีอีกเพียง 2 กรณี ที่เป็นไปได้คือ วัตถุทั้งสองไม่บังกันเลย (กรณีที่ 7) หรือวัตถุทั้งสองซ้อนกันอยู่ (กรณีที่6) การตรวจสอบในขั้นนี้ ทำโดย



รูปที่ 3.18 แสดงการตรวจการซ้อนกันของวัตถุหรือผิวด้าน

การลากเส้นตรงเส้นหนึ่งผ่านวัตถุทั้งสอง แล้วเปรียบเทียบความสัมพันธ์ของเส้นรอบรูปของวัตถุทั้งสอง ณ. จุดต่าง ๆ ที่ถูกตัดด้วยเส้นดังกล่าว วิธีทำโดยเลือกจุดพิกัดจุดใดจุดหนึ่งบนเส้นรอบรูป

ของวัตถุชิ้นเล็ก คือวัตถุซึ่งมีกรอบอาณาเขตเล็กกว่า (อยู่ภายในกรอบอาณาเขตของวัตถุที่ใหญ่กว่า) เพราะเส้นตรงที่ลากผ่านจุดดังกล่าว จะต้องตัดเส้นรอบรูปของวัตถุอีกชิ้น (ชิ้นใหญ่) อย่างแน่นอน ดังตัวอย่างในรูปที่ 3.18 ในกรณีของวัตถุ L และ M จุด aเป็นจุดเป็นเส้นรอบรูปของวัตถุชิ้นเล็ก (L) และ b กับ c เป็นจุดซึ่งเส้นตรงซึ่งลากผ่านจุด a ตัดกับวัตถุชิ้นใหญ่ (M) เราบอกได้ทันทีว่า L ซ้อนอยู่ภายใน M ทั้งนี้เพราะจุด a อยู่ระหว่างจุด b และ c กล่าวคือ เป็นจุดภายในเส้นตรง bc ส่วนกรณีวัตถุ N และ M จุด q และ r อยู่ทางขวามือของจุด p ทั้งคู่ กล่าวคือ จุด p เป็นจุดภายนอกของเส้นตรง qr ซึ่งแสดงว่าวัตถุ N อยู่ภายนอกวัตถุ M นั่นคือ วัตถุ N และ M ไม่บังหรือซ้อนกันเลย

จะเห็นว่าการตรวจสอบอาศัยเงื่อนไขความสัมพันธ์ของจุด 3 จุด ที่ได้จากการลากเส้นตรงผ่านวัตถุคู่ที่พิจารณา สรุปเป็นขั้นตอนได้ดังนี้

1. เลือกจุดพิกัดใด ๆ ก็ได้ ซึ่งอยู่ในวัตถุที่มีกรอบอาณาเขตเล็กกว่า ให้เป็นจุด a (ที่ง่ายที่สุดก็คือ จุดปลายเส้นใดเส้นหนึ่งในวัตถุดังกล่าวนั่นเอง)

2. โปรเจคเส้นตรงเส้นหนึ่งให้ผ่านจุด a และขนานกับแกน x (หรือ y ก็ได้) หาจุดพิกัด b และ c ซึ่งเป็นจุดที่เส้นดังกล่าวตัดเส้นรอบรูปของวัตถุอีกชิ้นหนึ่ง

3. ถ้าค่าพิกัดของจุด a มากกว่าหรือน้อยกว่าค่าพิกัดของ b และ c ทั้งคู่ตามแกน x (หรือ y) วัตถุทั้งสองไม่มีการบังกัน ถือว่าเสร็จสิ้นการตรวจสอบ

4. ถ้าไม่เป็นจริงตามข้อ 3 แสดงว่าวัตถุเล็กซ้อนอยู่ในวัตถุใหญ่

ถ้าผลการตรวจสอบเป็นไปตามข้อ 4. เราจะต้องตรวจสอบว่า วัตถุชิ้นใดอยู่หน้าหรือหลัง ถ้าวัตถุชิ้นเล็กอยู่หลัง เราตัดข้อมูลวัตถุชิ้นเล็กทั้งชิ้นออกไปได้เลย เพราะจะไม่ปรากฏในภาพ ในกรณีที่วัตถุชิ้นเล็กอยู่ข้างหน้า ถ้าการตรวจสอบกระทำในระดับผิวด้านต่อผิวด้าน (ไม่มีเส้นภายใน) เราสามารถผ่านไปเลยได้ เพราะเส้นรอบรูปของผิวด้านทั้งสองจะไม่ถูกกระทบกระเทือน ถ้าการตรวจสอบกระทำในระดับวัตถุต่อวัตถุ (มีเส้นภายใน) เราจะต้องทำการลบเส้นภายในส่วนที่ถูกบังด้วย

การตรวจสอบลำดับความลึกของวัตถุที่ซ้อนกัน ทำได้โดยการโปรเจคเส้นตรงจากจุดใดจุดหนึ่ง

บนวัตถุชิ้นเล็กไปตามแกน Z แล้วเปรียบเทียบความลึกของจุดนั้น กับจุดที่เส้นโปรเจคตัดกับพื้นที่ผิวด้าน (ที่มองเห็น) ของวัตถุชิ้นใหญ่ ถ้าจุดบนวัตถุชิ้นเล็กมีค่าตามแกน Z มากกว่า แสดงว่าวัตถุชิ้นเล็กอยู่ข้างหลัง ถ้าไม่เช่นนั้นก็แสดงว่าวัตถุชิ้นเล็กอยู่ข้างหน้า

3.6.5 สรุปขั้นตอนการลบส่วนที่ถูกบัง

จะเห็นได้ว่าการลบส่วนที่ถูกบังของวัตถุในภาพ มีความละเอียดและซับซ้อนที่สุดในระบบงานสร้างภาพของวัตถุ 3 มิติ การทำงานของขั้นตอนนี้ทั้งหมด แสดงไว้ในผังงานรูปที่ 3.19 ถึง 3.23 ในรูปที่ 3.19 (n) ซึ่งเป็นส่วนที่ทำการจับคู่ของวัตถุในการเปรียบเทียบนั้น วัตถุ A ในวงจรชั้นนอกเริ่มตั้งแต่วัตถุที่ 1 ถึง n แต่วงจรชั้นในวัตถุ B มี 2 ชุดคือ ตั้งแต่ A+1 ถึง n และ A-1 ถอยหลังไปจนพบว่าวัตถุ B ไม่มีกรอบอาณาเขตตามแกน X (หรือ Y) (เมื่อวัตถุทั้งหมดจัดเรียงลำดับตามค่าจุดพิกัดสูงสุดในแกน X (หรือ Y)) ทับเกยกับกรอบอาณาเขตของวัตถุ A หรือ B เป็น 1

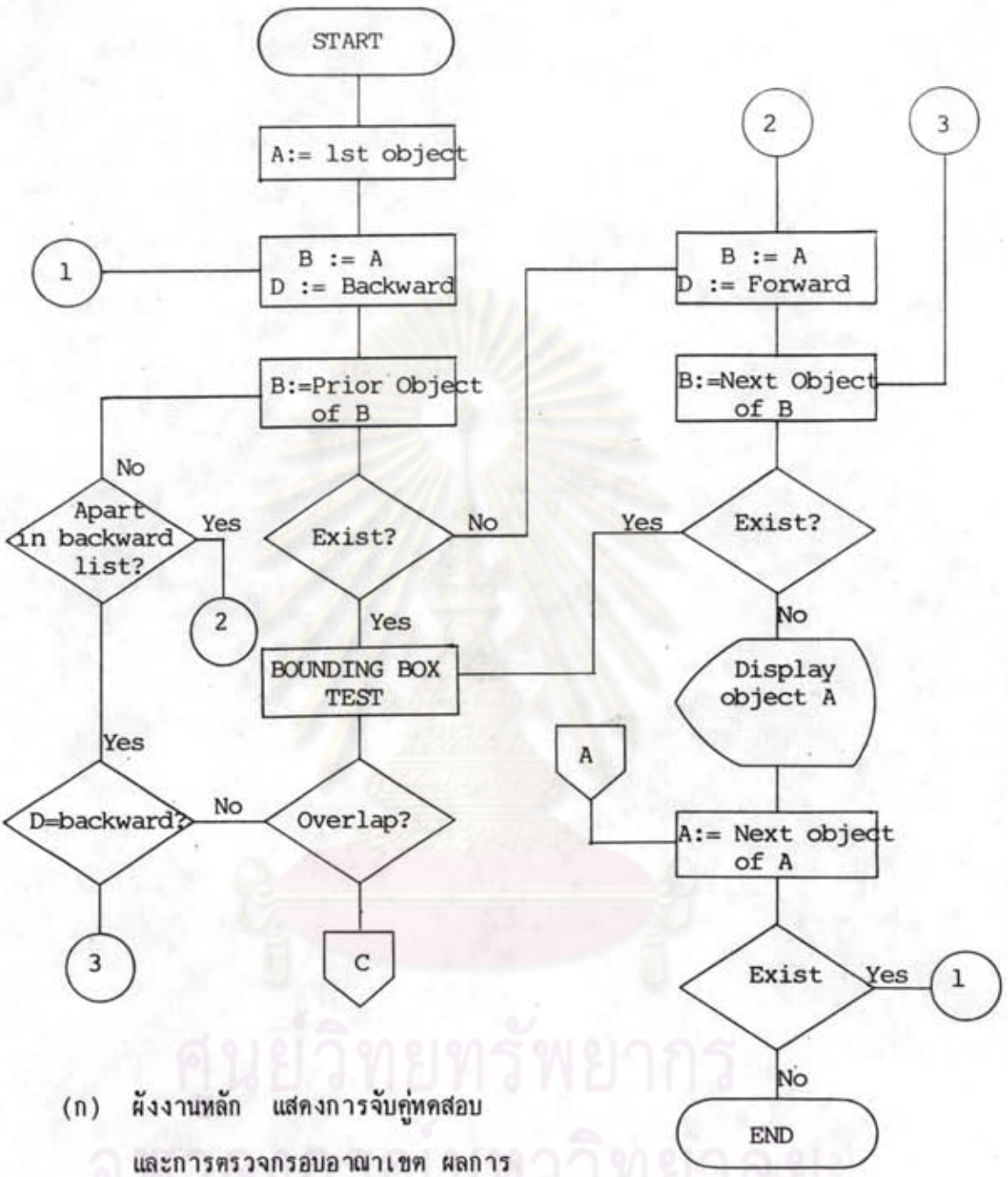
3.7 โครงสร้างข้อมูล

ข้อมูลในระบบงานแยกเป็น 2 ประเภทคือ ข้อมูลพื้นฐานของวัตถุ และข้อมูลซึ่งเกิดขึ้นในขบวนการสร้างภาพ

3.7.1 ข้อมูลพื้นฐานของวัตถุ

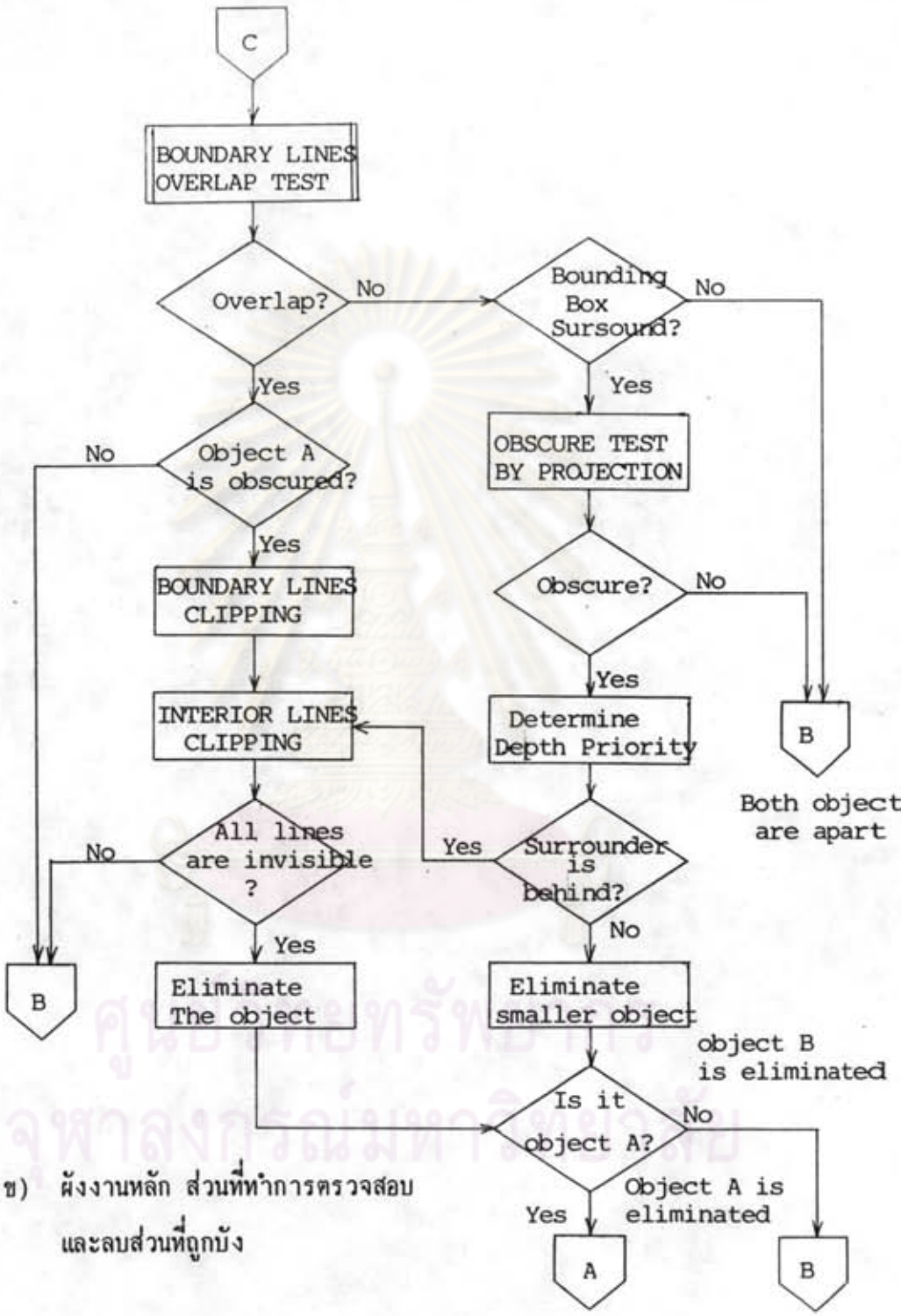
ประกอบด้วย

1. ข้อมูลจุดพิกัดยอดมุมต่าง ๆ ของวัตถุ ซึ่งเป็นข้อมูลขั้นพื้นฐานที่สำคัญที่สุด
2. ข้อมูลเส้น คือ ข้อมูลที่ระบุแนวสันขอบต่าง ๆ ของวัตถุ ปลายทั้งสองของเส้นก็คือจุดยอดมุม
3. ข้อมูลผิวหน้า คือ ข้อมูลเส้นที่ประกอบขึ้นเป็นผิวหน้า และข้อมูลเกี่ยวกับทิศทางการหันของแต่ละผิวหน้า ได้แก่ นอร์แมลเวคเตอร์



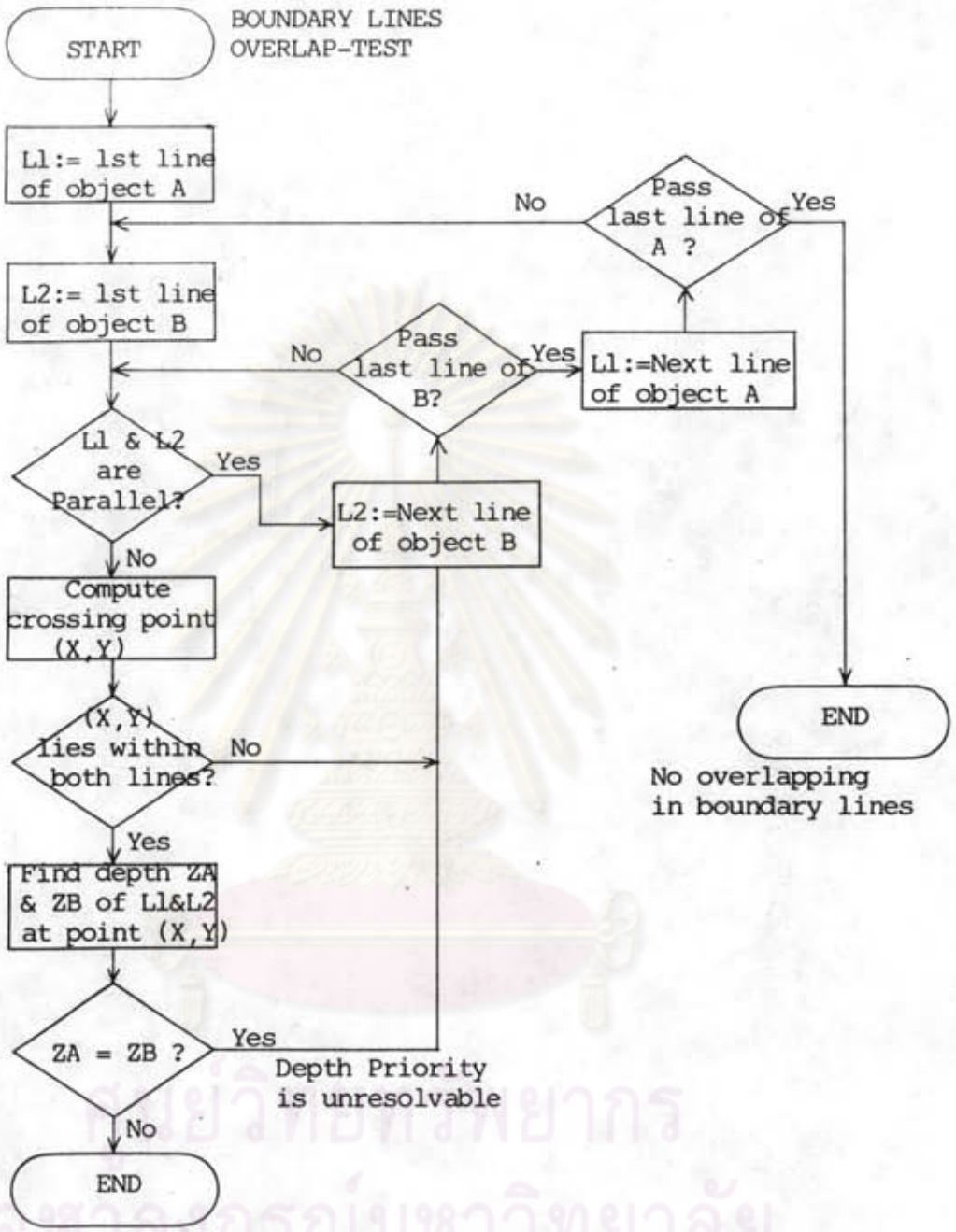
(ก) ผังงานหลัก แสดงการจับคู่ทดสอบ และการตรวจกรอบอาณาเขต ผลการทดสอบที่ใช้แสดงภาพคือวัตถุ A

รูปที่ 3.19 แสดงผังงานการลบส่วนที่ถูกบังของวัตถุในภาพ (ยังมีต่อ)

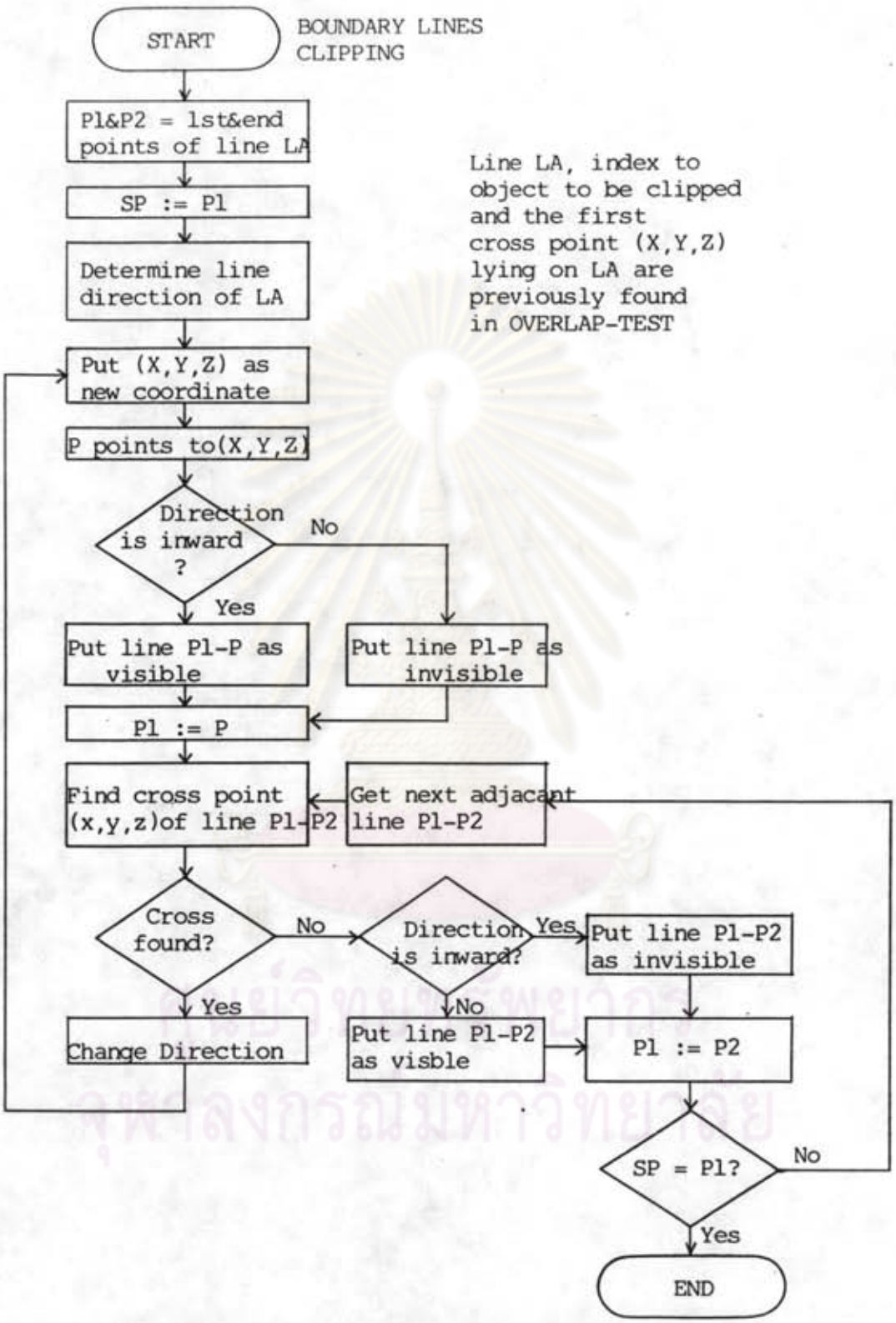


(ข) ผังงานหลัก ส่วนที่ทำการตรวจสอบ และลบส่วนที่ถูกบัง

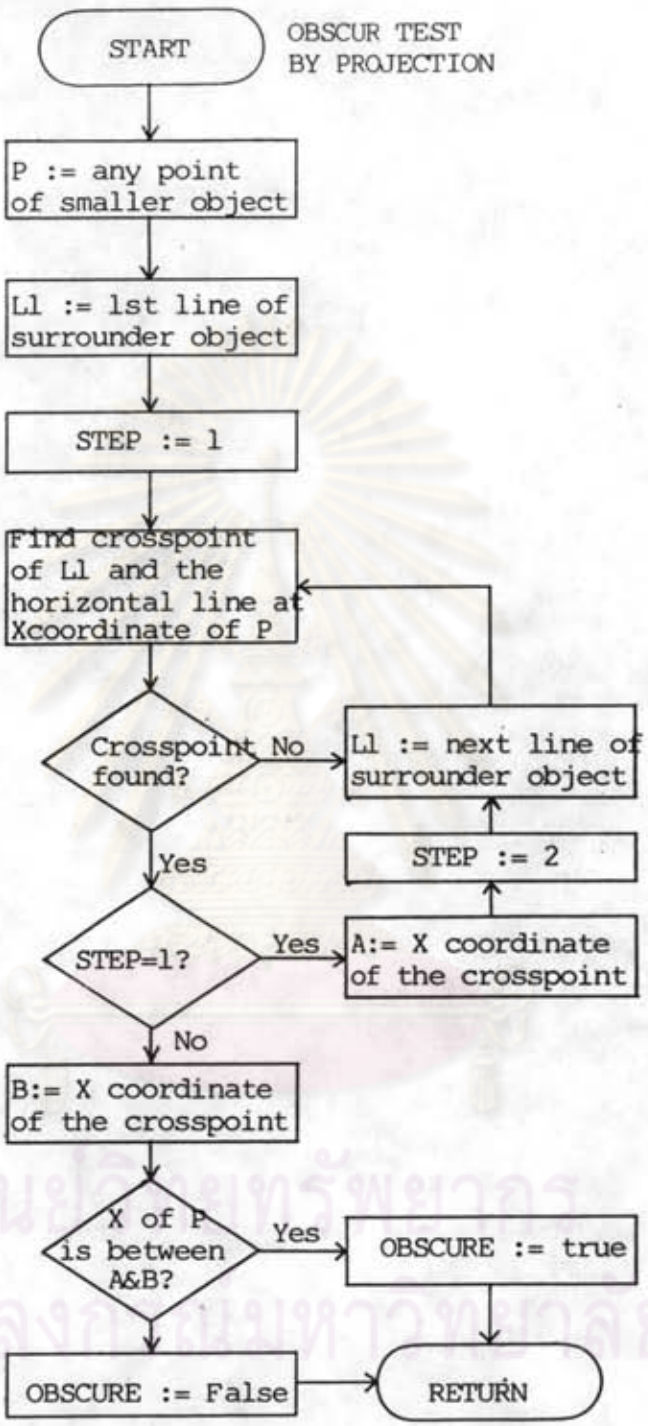
รูปที่ 3.19 (ต่อ)



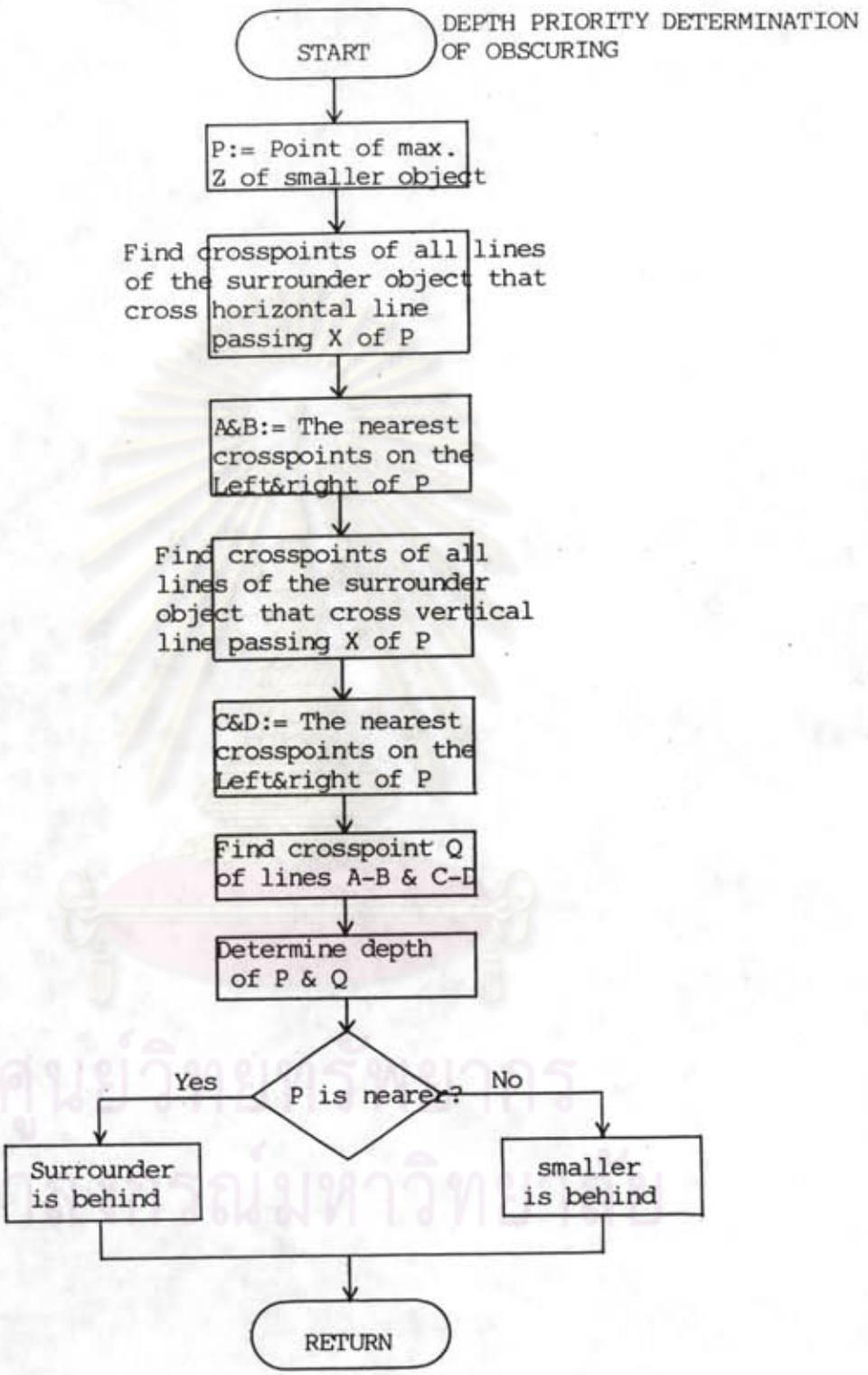
รูปที่ 3.20 แสดงผังงานตรวจการทับเกยของเส้นรอบรูปและลำดับความลึก



รูปที่ 3.21 แสดงผังงานการลบเส้นรอบรูปวัตถุส่วนที่ถูกบัง

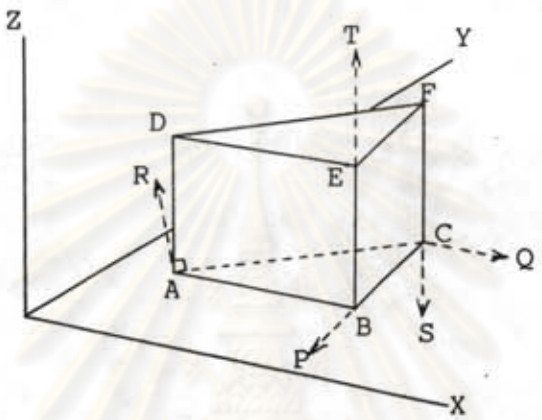


รูปที่ 3.22 แสดงผังงานการตรวจการซ้อนกันของวัตถุ



รูปที่ 3.23 แสดงผังงานการเปรียบเทียบความลึกของวัตถุที่ซ้อนกัน

ข้อมูลทั้งหมดนี้ โดยสรุปก็คือ ข้อมูลจุดพิกัดทั้งหลายของวัตถุ และความสัมพันธ์ของจุดพิกัดเหล่านั้นนั่นเอง ข้อมูลจุดพิกัดดังกล่าวเป็นค่าตัวเลขที่แท้จริงเกี่ยวกับวัตถุที่เก็บในระบบ ในขณะที่ข้อมูลเส้นและผิวหน้าจะเป็นเพียงชุดของดรรชนี หรือตัวบ่งชี้ที่อ้างถึงค่าตัวเลขของจุดพิกัด ดังตัวอย่างต่อไปนี้ (ดูรูป3.24)



รูปที่ 3.24 แสดงตัวอย่างข้อมูลพื้นฐานของวัตถุ

รายการจุดพิกัดยอดมุม :	ดรรชนี/ตัวบ่งชี้	ค่าจุดพิกัด
	A	(2,4,0)
	B	(10,4,0)
	C	(10,9,0)
	D	(2,4,6)
	E	(10,4,6)
	F	(10,9,6)

รายการจุดพิกัดนอร์แมลเวคเตอร์ :	ดรรชนี/ตัวบ่งชี้	ค่าจุดพิกัด
	P	(10,1,0)
	Q	(14,9,0)

รายการจุดพิกัดนอร์แมลเวคเตอร์ :	ดรรชนี/ตัวบ่งชี้	ค่าจุดพิกัด
	R	(-2,11,0)
	S	(10,9,-4)
	T	(10,9,10)

รายการข้อมูลผิวหน้า :	ผิวหน้าที่	ข้อมูล
	1	ABEDA
	2	BCFEB
	3	CADFC
	4	ACBA
	5	DEFD

จะเห็นได้ว่ารายการข้อมูลทั้งหมดนี้ เพียงพอที่จะอธิบายวัตถุตามรูปที่ 3.24ได้ครบถ้วนแล้ว รายการข้อมูลผิวหน้าจะแสดงถึงรายการข้อมูลเส้นไปด้วยในตัว เช่น ผิวหน้าที่ 1 ABEDA จะแสดงถึงเส้น AB, BE, ED และ DA โดยมีทิศทางของเส้นไปในทางเดียวกัน และในขณะที่ BP ก็จะแสดงถึงนอร์แมลเวคเตอร์ของผิวหน้าที่ 1 ไปด้วย ในทางปฏิบัติระบบงานหนึ่ง ๆ อาจจะแยกข้อมูลแต่ละส่วนออกให้เด่นชัดเลยก็ทำได้ อย่างไรก็ดี รายการข้อมูลตามตัวอย่างนี้ เป็นข้อมูลที่รวบรัดที่สุดเท่าที่จะทำได้สำหรับวัตถุในรูปแบบพื้นฐานชิ้นหนึ่ง

รายการข้อมูลประเภทต่าง ๆ ของวัตถุ ซึ่งมีรูปแบบพื้นฐานแต่ละชิ้น จะต้องสามารถอ้างถึงได้ทั้งหมด เมื่อระบบงานเข้าไปเกี่ยวข้องกับวัตถุชิ้นนั้น ๆ ในขณะเดียวกัน ข้อมูลทั้งหมดของวัตถุแต่ละชิ้น ก็จะต้องสามารถจัดแยกออกจากกันได้ โดยเฉพาะในภาวะปกติของการปรับปรุงข้อมูลดิบของระบบงานทั่ว ๆ ไป ที่จะสามารถตัดหรือเพิ่มข้อมูลวัตถุทั้งชิ้นเข้าหรือออกจากแฟ้มข้อมูล รวมทั้งการเปลี่ยนแปลงข้อมูลของวัตถุชิ้นใดชิ้นหนึ่งในระบบ ดังนั้นข้อมูลทุกประเภทของวัตถุชิ้นหนึ่ง ๆ จะต้องถูกรวบรวมเข้าอยู่ในชุดเดียวกันในลักษณะใดลักษณะหนึ่ง และข้อมูลชุดของวัตถุแต่ละชิ้นก็จะต้องถูกรวบรวมเข้า แฟ้มข้อมูลในลักษณะของรายการข้อมูลแยกตามวัตถุ (Segment list) อย่างไรก็ดี วิธีการจัดเก็บลงในหน่วยความจำ จะต้องคำนึงถึงลักษณะของวิธีการสร้างภาพในขั้นตอนต่าง ๆ ด้วย

3.7.2 ข้อมูลซึ่งเกิดในขบวนการสร้างภาพ

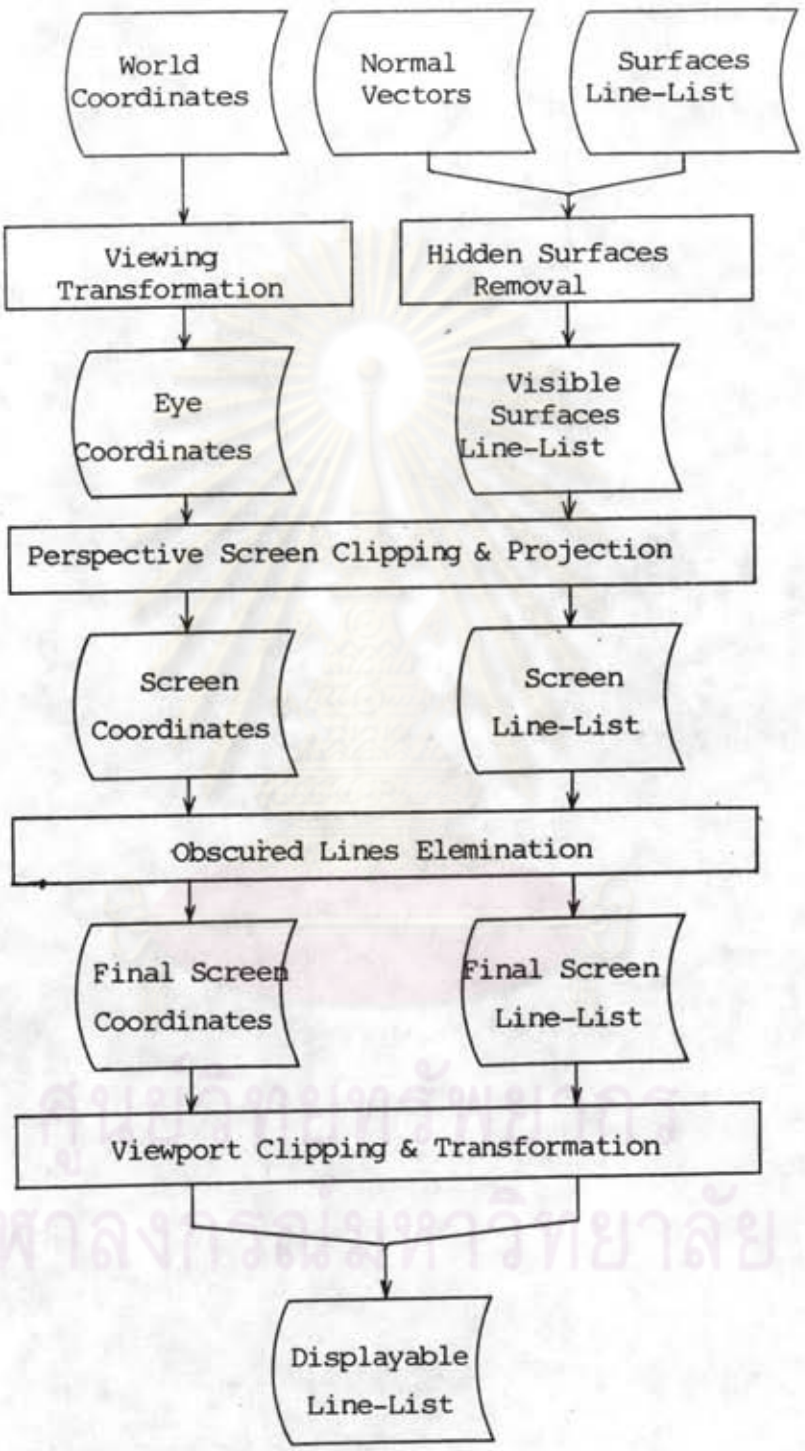
ข้อมูลพื้นฐานของวัตถุ จะต้องถูกแปลงรูปไปเป็นข้อมูลในลักษณะต่าง ๆ ตามขั้นตอนของการสร้างภาพ กรรมวิธีการทำงานและลักษณะข้อมูลในแต่ละขั้นตอน ทำให้สามารถสรุปลำดับการทำงานและความเคลื่อนไหวของข้อมูลได้ตามรูปที่ 3.25 กล่าวคือ

3.7.2.1 การแปลงข้อมูลจุดพิกัดพื้นฐานไปเป็นจุดพิกัดในการมอง ขั้นตอนนี้ ทำงานกับจุดพิกัดอย่างเดียว การจัดโครงสร้างข้อมูลที่เอื้ออำนวยต่อการทำงานมากที่สุด คือจัดเป็นข้อมูลชุด (Array) ของจุดพิกัดทั้งหลายของวัตถุทุกชิ้น

3.7.2.2 การลบผิวหน้าที่มองไม่เห็น เป็นขั้นตอนที่อาศัยข้อมูลพื้นฐานทั้งหมดของวัตถุ ผลที่ได้คือ ข้อมูลเฉพาะผิวหน้าที่มองเห็นเท่านั้น ซึ่งจะช่วยลดปริมาณข้อมูลสำหรับขั้นตอนต่อ ๆ ไป ขั้นตอนนี้จะไม่มีผลเปลี่ยนแปลงต่อข้อมูลจุดพิกัด ดังนั้นจึงเป็นขั้นตอนที่สามารถทำควบคู่กับขั้นตอนในข้อ 3.7.2.1 ได้ และทั้งสองขั้นตอนนี้จะสลับลำดับกันก็ได้ แต่อย่างไรก็ดี จะเป็นสองขั้นตอนแรกของการทำงานทั้งหมด

ผลลัพธ์ที่ได้จากการลบผิวหน้าที่มองไม่เห็นนั้น จะต้องออกมาในลักษณะของรายการข้อมูลเส้นของวัตถุแต่ละชิ้น ซึ่งระบุทิศทางของเส้นอย่างเป็นระเบียบด้วย เนื่องจากขั้นตอนต่อ ๆ ไปนั้นทำงานกับข้อมูลระดับเส้น

3.7.2.3 การขลิบขอบภาพและการโปรเจคภาพ ข้อมูลที่ได้จาก 2 ขั้นตอนแรก จะต้องผ่านขั้นตอนนี้เพื่อให้ได้รายการจุดพิกัดและเส้นของภาพ โครงสร้างข้อมูลที่ได้จากขั้นตอนนี้และขั้นตอนต่อ ๆ ไป จะเหมือนกับข้อมูลที่ได้จากการลบผิวหน้าที่มองไม่เห็น อย่างไรก็ดี จำนวนจุดพิกัดและเส้นจะเปลี่ยนแปลงไป สำหรับรายการข้อมูลที่ได้จากขั้นตอนนี้ จุดพิกัดและเส้น อาจถูกตัดทิ้งไปและเพิ่มขึ้นมา เนื่องจากการขลิบขอบภาพ และสถานะของเส้นจะเป็นข้อมูลที่เพิ่มเติมขึ้นมาเนื่องจากวัตถุที่ถูกขลิบ จะต้องเพิ่มเส้นรอบรูปให้ครบ (โดยอาศัยเส้นกรอบภาพ) เส้นดังกล่าวจะใช้ในขบวนการลบส่วนที่ถูกบัง แต่ไม่เป็นข้อมูลที่แท้จริงของภาพ



รูปที่ 3.25 แสดงลำดับการทำงานและข้อมูลโดยสรุป

3.7.2.4 การลบส่วนที่ถูกบัง ผลที่ได้จากขั้นตอนนี้ จะเป็นรายการข้อมูลจุดพิกัดและเส้นซึ่งมีจำนวนเปลี่ยนแปลงไปเช่นเดียวกับขั้นตอนที่แล้ว สิ่งสำคัญก็คือ ระหว่างการทำงานนั้น ข้อมูลเดิมจากขั้นตอนที่แล้วของวัตถุแต่ละชิ้น จะต้องคงสภาพเดิม เพื่อใช้ในการตรวจสอบการบังกันกับวัตถุชิ้นอื่น ๆ แม้ว่าตัวมันเองจะทำให้เกิดข้อมูลใหม่อันเนื่องจากการบังกันขึ้นมา จนกว่าการลบส่วนที่ถูกบังของวัตถุชิ้นหนึ่ง ๆ อันเนื่องจากวัตถุชิ้นอื่น ๆ ทั้งหมดจะเสร็จสิ้นลง ข้อมูลเดิมของวัตถุชิ้นนั้น จึงจะหมดประโยชน์ และสิ่งที่ได้ก็จะเป็นรายการจุดพิกัดและเส้นที่พร้อมสำหรับขั้นตอนการแสดงภาพ รายการข้อมูลเส้นจะยังคงต้องระบุสถานะไว้ว่าเส้นใดคือเส้นที่จะใช้แสดงภาพหรือไม่

3.7.2.5 การแสดงภาพ ข้อมูลเส้นและจุดพิกัดจะถูกแปลงเป็นข้อมูลภาพเพื่อใช้แสดงเป็นขั้นตอนสุดท้าย ข้อสังเกตคือ ข้อมูลจากขั้นตอนก่อน ๆ จะเป็นข้อมูลที่สามารถจัดแยกได้ว่าเป็นของวัตถุชิ้นใด ๆ และตัวข้อมูลเส้นจะเป็นเพียงดรรชนีหรือตัวบ่งชี้ไปยังตำแหน่งที่ถูกต้องของค่าจุดพิกัดปลายเส้นซึ่งเก็บไว้ในรายการข้อมูลจุดพิกัด ข้อมูลเหล่านี้จะถูกแปลงให้อยู่ในรูปของรายการเส้นต่อเนื่องเพียงรายการเดียว โดยแต่ละเส้นเก็บในลักษณะค่าจุดพิกัดปลายเส้นเลยทีเดียว ซึ่งเป็นรูปแบบข้อมูลที่ใช้ในการวาดภาพโดยตรง

3.8 การจัดการข้อมูลในลักษณะโมดูลาร์

ในแนวความคิดเกี่ยวกับระบบการสร้างภาพวัตถุ 3 มิตินั้น วัตถุในรูปทรงใด ๆ ก็ตาม จะประกอบขึ้นจากรูปทรงพื้นฐานเสมอ การกำหนดว่าอะไรก็ข้อมูลรูปทรงพื้นฐานจะขึ้นอยู่กับลักษณะงาน ในที่นี้จะกล่าวเฉพาะในแง่ของรูปทรงพื้นฐานที่เป็นวัตถุ 3 มิติเท่านั้น

รูปทรงพื้นฐานของวัตถุ 3 มิติ ตามขอบเขตของวิทยานิพนธ์ฉบับนี้ก็คือ วัตถุรูปทรงสี่เหลี่ยมมุมฉาก (Box) ซึ่งมีขนาดต่าง ๆ กัน วัตถุในรูปทรงพื้นฐานหนึ่ง ๆ นั้น จะมีปริมาณข้อมูลและความสัมพันธ์เหมือนกันหมด เช่น รูปทรงสี่เหลี่ยมมุมฉาก จะมีจุดยอดมุม 8 จุด ผิวด้าน6 ด้าน เส้นขอบวัตถุ 12 เส้นเสมอ โดยที่ทุกเส้นตั้งฉากกัน ทุก ๆ จุดยอดมุมจะเป็นจุดร่วมของปลายเส้น 3 เส้นเสมอ ด้วยข้อเท็จจริงอันนี้ การสร้างข้อมูลของวัตถุในรูปทรงพื้นฐานขึ้นหนึ่งก็จะทำได้โดยง่าย ผู้ใช้งานสามารถอธิบายวัตถุที่ต้องการ โดยอาศัยข้อมูลสรุปเพียงไม่กี่ตัว ได้แก่ ขนาดความกว้าง ความยาว

ความสูง ตำแหน่งที่วาง และมุมที่วาง จากข้อมูลเหล่านี้ ระบบงานสามารถสร้างข้อมูลพื้นฐานของวัตถุชิ้นนั้น ๆ ได้ครบถ้วน โดยอาศัยการคำนวณเพื่อหาค่าจุดพิกัดยอดมุมทั้ง 8 จุด รายการผิวหน้าและเส้นก็สามารถจัดวางเป็นรูปแบบที่แน่นอนได้โดยการกำหนดลักษณะความสัมพันธ์ของจุดต่าง ๆ ในระดับของเส้นและผิวด้านไว้ตายตัว และค่าจุดพิกัดนอร์แมลเวคเตอร์ก็คำนวณมาจากจุดพิกัดและความสัมพันธ์ดังกล่าว เช่น กำหนดให้ตำแหน่งที่ตั้งของวัตถุ เป็นค่าจุดพิกัดจุดแรก และเป็นจุดมุมซ้ายล่างของวัตถุเมื่อมองเข้าไปตามแกน Z โดยที่ความกว้าง ความยาว และความสูง หมายถึงระยะตามแกน X Y และ Z ตามลำดับ และจุดอื่น ๆ จัดเรียงไปในลักษณะใดลักษณะหนึ่งที่แน่นอนเสมอ ลักษณะเช่นนี้ ทำให้เกิดข้อมูลอีกชุดหนึ่งขึ้นมาในระบบงานก็คือ ข้อมูลแบบรูปทรงของวัตถุซึ่งใช้เป็นโครงร่างในการสร้างข้อมูลพื้นฐานของวัตถุจากข้อมูลเพียง 5 ชิ้น ที่ผู้ใช้ป้อนให้ระบบ โปรแกรมในการสร้างข้อมูลพื้นฐานของวัตถุจะสอดคล้องกับแบบรูปทรงดังกล่าว แต่อย่างไรก็ดี ข้อมูลที่ป้อนเข้ามาดังกล่าว ก็จะต้องมีความสอดคล้องกับระบบด้วย

การมีข้อมูลแบบรูปทรงวัตถุ ประกอบกับการจัดข้อมูลรูปทรงดังกล่าวในลักษณะดรรชนี หรือตัวบ่งชี้ จะช่วยลดปริมาณข้อมูลของวัตถุแต่ละชิ้นในแฟ้มข้อมูลลงได้ กล่าวคือ ข้อมูลเฉพาะของวัตถุแต่ละชิ้นประกอบด้วย ค่าจุดพิกัดยอดมุม และค่าจุดพิกัดนอร์แมลเวคเตอร์เท่านั้น ส่วนข้อมูลเส้นและผิวด้าน ซึ่งเป็นข้อมูลพื้นฐานจะมีเพียงชุดเดียวสำหรับวัตถุในรูปทรงพื้นฐานเดียวกันคือ ข้อมูลแบบรูปทรงวัตถุนั่นเอง

ลักษณะทางโมดูลาร์ดังกล่าว สามารถใช้กับรูปทรงวัตถุใด ๆ ก็ได้ ในระบบงานที่สร้างภาพของชุดวัตถุที่ประกอบด้วยวัตถุรูปทรงเดียวกันหลาย ๆ ชิ้น ซึ่งใช้อยู่เสมอ ๆ และในระบบงานเดียวกันก็อาจประกอบด้วยวัตถุในรูปทรงโมดูลาร์หลาย ๆ รูปทรง โดยแต่ละรูปทรงก็มีข้อมูลแบบรูปทรงตลอดจนวิธีการสร้างข้อมูลพื้นฐานของรูปทรงนั้นเป็นของตัวเอง

3.9 ข้อสรุปเกี่ยวกับโครงสร้างข้อมูล

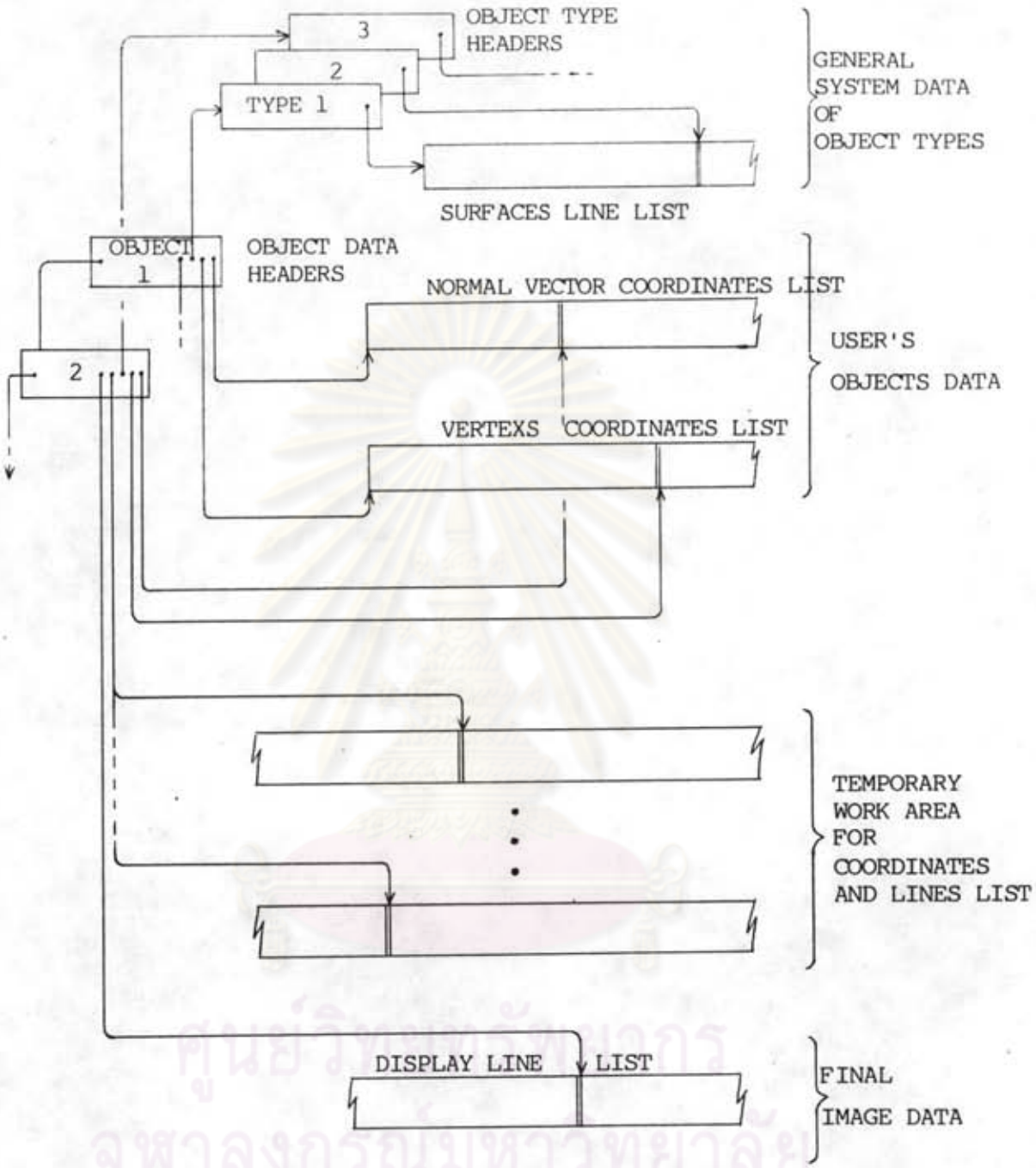
การจัดโครงสร้างข้อมูล มีเป้าหมายสำคัญที่ได้พิจารณาอยู่ 2 ประเด็น คือ ความสะดวกในการสร้างและบำรุงรักษา และความมีประสิทธิภาพในการประมวลผลตามขั้นตอนการสร้างภาพตามที่ได้กล่าวมาแล้ว ทั้ง 2 ประเด็นนี้ นำไปสู่การจัดโครงสร้างข้อมูลแบบแบ่งเป็นหมวดหมู่ ซึ่งมีอยู่ 2 ลักษณะ คือ

1. การจัดเป็นรายการข้อมูลแยกชุดตามวัตถุ (Segment List) ข้อมูลทุกชนิดของวัตถุแต่ละชิ้น จะจัดแยกไปตามวัตถุนั้น ๆ ลักษณะเช่นนี้ ทำให้การปรับปรุงข้อมูลทำได้สะดวก เนื่องจากการปรับปรุงข้อมูลจะทำต่อวัตถุทั้งชิ้น ซึ่งมีผลต่อข้อมูลทุกชนิดของวัตถุชิ้นนั้น ๆ ทั้งหมดเสมอ เช่นการย้ายที่ตั้ง การลบวัตถุออกจากฐานข้อมูล การเพิ่มวัตถุชิ้นใหม่เข้าในฐานข้อมูล เป็นต้น นอกจากนี้ขั้นตอนการสร้างภาพ ก็ยังจำเป็นต้องอ้างถึงข้อมูลที่เป็นของวัตถุแต่ละชิ้นได้ด้วย

2. การจัดข้อมูลแยกตามชนิดของข้อมูล (Data Catagories) กล่าวคือ จัดรวมข้อมูลประเภทเดียวกันไว้ด้วยกัน เช่น รายการข้อมูลจุดพิกัดยอดมุมของวัตถุทั้งหลายรายการจุดพิกัดนอร์แมลเวคเตอร์ และรายการเส้น เป็นต้น การจัดข้อมูลลักษณะนี้ จะช่วยให้การจัดการข้อมูลทำได้โดยสะดวกชัดเจน และป้องกันการปะปนกันของข้อมูลในระหว่างการประมวลผลเนื่องจากข้อมูลแต่ละประเภทมีลักษณะต่างกันทั้งในแง่ความหมายของข้อมูลปริมาณหน่วยความจำที่ต้องใช้สำหรับข้อมูลแต่ละหน่วย และจำนวนข้อมูลในระหว่างการประมวลผลซึ่งไม่คงที่ นอกจากนี้การจัดข้อมูลชนิดเดียวกันไว้ด้วยกันก็ยังช่วยเพิ่มประสิทธิภาพของการทำงานได้ในหลาย ๆ ขั้นตอน เช่น การแปลงค่าจุดพิกัดพื้นฐานไปเป็นค่าจุดพิกัดในการมอง ซึ่งมีประสิทธิภาพสูงขึ้นเมื่อสามารถทำงานแบบเรียงลำดับโดยที่จุดพิกัดทั้งหลายวางเรียงต่อเนื่องกันไปทั้งหมด เป็นต้น

ลักษณะทั้ง 2 ประการนี้ จะต้องผสมผสานใช้ร่วมกัน โครงสร้างข้อมูลในลักษณะดังกล่าวของแบบจำลอง แสดงในรูปที่ 3.26 ซึ่งมีสาระสำคัญดังนี้

ข้อมูลทั้งหมดในระบบแบ่งออกเป็น 4 ส่วน คือ



รูปที่ 3.26 แสดงลักษณะโครงสร้างข้อมูล

3.9.1 ข้อมูลแบบรูปทรงของวัตถุ

ข้อมูลส่วนนี้เป็นส่วนสำคัญที่จะช่วยให้ใช้หน่วยความจำประหยัดขึ้น และลดความซ้ำซ้อนของข้อมูลวัตถุต่าง ๆ ที่มีรูปทรงเดียวกันลงได้ ข้อมูลส่วนนี้จะประกอบด้วยรายละเอียดทุกประการเกี่ยวกับรูปทรงหนึ่ง ๆ ของวัตถุรวมทั้งความสัมพันธ์ต่าง ๆ ของข้อมูลต่างชนิดในวัตถุชิ้นหนึ่ง ๆ ได้แก่ จำนวนจุดยอดมุมเส้นขอบด้านและผิวหน้าวัตถุ รายการเส้นและผิวหน้าของวัตถุซึ่งจัดไว้ในลักษณะที่สะดวกสำหรับการสร้างข้อมูลจุดพิกัดต่างๆ รวมทั้งสะดวกสำหรับขั้นตอนต่าง ๆ ในการสร้างภาพ เช่น การตรวจสอบผิวหน้าที่มองเห็นการสร้างรายการเส้นที่เป็นระเบียบ ซึ่งเป็นสิ่งจำเป็นสำหรับขั้นตอนการลบเส้นที่ถูกบัง เป็นต้น อย่างไรก็ดีเนื่องจากข้อมูลเส้นและผิวหน้าของแต่รูปทรงของวัตถุมีจำนวนไม่เท่ากัน ข้อมูลส่วนนี้จึงแบ่งออกเป็น 2 ส่วนย่อย คือส่วนที่เก็บข้อมูลที่มีขนาดคงที่อันได้แก่ข้อมูลจำนวนจุดพิกัดยอดมุมจำนวนเส้นขอบด้านและจำนวนผิวหน้า อีกส่วนจัดเก็บข้อมูลรายการเส้นและผิวหน้าซึ่งมีขนาดไม่คงที่

3.9.2 ข้อมูลจุดพิกัดวัตถุ

ส่วนนี้จัดเก็บเฉพาะค่าจุดพิกัดยอดมุม และค่าจุดพิกัดนอร์แมลเวคเตอร์ ซึ่งต่างกันไปตามแต่ละวัตถุ ข้อมูลส่วนนี้แยกเป็น 2 ส่วนย่อยเช่นกัน ค่าจุดพิกัดทั้ง 2 ชนิดของวัตถุทั้งหลาย จะจัดเก็บไว้ด้วยกัน การอ้างถึงทำโดยตัวบ่งชี้ซึ่งเก็บไว้ในส่วนข้อมูลนำ (Header) ของวัตถุแต่ละชิ้น สำหรับส่วนข้อมูลนำนี้เป็นส่วนสำคัญ เพราะจะเก็บดรรชนีหรือตัวบ่งชี้ไปยังข้อมูลทุกชนิดในฐานข้อมูลที่เป็นของวัตถุชิ้นหนึ่ง ๆ

3.9.3 ข้อมูลชั่วคราวในการสร้างภาพ

ประกอบด้วยข้อมูลจุดพิกัดและรายการเส้นที่เกิดขึ้น และหายไปในระหว่างการทำงานไปตามขั้นตอนต่าง ๆ ของการสร้างภาพ ปริมาณข้อมูลที่เกิดขึ้นนั้นไม่คงที่ แต่จะเปลี่ยนแปลงอยู่เสมอ ข้อมูลประเภทนี้จะมีตัวบ่งชี้มาจากส่วนข้อมูลนำของวัตถุทุกชิ้นอยู่ตลอดเวลา

3.9.4 ข้อมูลภาพ

คือข้อมูลขั้นสุดท้ายที่ได้จากการสร้างภาพ ข้อมูลส่วนนี้เป็นข้อมูลชุดเดียวซึ่งเก็บข้อมูลภาพของวัตถุทุกชิ้นไว้รวมกัน ในการแสดงภาพจากข้อมูลชุดนี้ไม่มีความจำเป็นต้องแบ่งแยกวัตถุ อย่างไรก็ดีในแบบจำลองข้อมูลส่วนนี้เก็บภาพที่ใช้ในการตอบรับการป้อนข้อมูลด้วย ดังนั้นส่วนข้อมูลนำของวัตถุตามข้อ 3.9.2 ก็จะเก็บตัวบ่งชี้มายังข้อมูลภาพส่วนที่เป็นของวัตถุแต่ละชิ้นด้วย

ข้อสังเกตุสำคัญประการหนึ่งคือ แม้แนวความคิดเกี่ยวกับโครงสร้างข้อมูลซึ่งสรุปไว้ในวิทยานิพนธ์นี้ จะเป็นแนวความคิดที่เหมาะสมสำหรับงานด้านกราฟฟิคทั่วไป แต่ก็ไม่ได้หมายความว่าการจัดโครงสร้างข้อมูลในขั้นรายละเอียดจะเป็นลักษณะที่ดีที่สุดสำหรับงานทั่ว ๆ ไปด้วย เนื่องจากการจัดโครงสร้างข้อมูลที่เหมาะสมสำหรับระบบงานหนึ่ง ๆ ขึ้นอยู่กับเงื่อนไขปลีกย่อยหลายประการที่นอกเหนือจากลักษณะงานเอง อาทิเช่น คุณสมบัติของเครื่องอุปกรณ์ ปริมาณหน่วยความจำโปรแกรมควบคุมระบบของเครื่องอุปกรณ์และภาษาที่ใช้ในการพัฒนาระบบงาน เป็นต้น