[illegible]

[illegible]

[illegible]

โดย

[illegible]

[illegible]

[illegible]

แบบเสนอโครงงานวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม
ประจำภาคการศึกษาที่ 1/2550
เรื่อง การจำลองระบบ MIMO ภายในอาคารในรูปแบบกราฟฟิก

โดย

นายกิตติชัย	มีหมู่	รหัส	B4600466
นายธนากร	จิรคงสวัสดิ์	รหัส	B4603474
นายวิระเดช	แพงแซง	รหัส	B4608516

รายงานนี้เป็ส่วนหนึ่งของรายวิชา 427499โครงงานวิศวกรรมโทรคมนาคม
ประจำภาคการศึกษาที่ 1 ปีการศึกษา2550
หลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมโทรคมนาคม
สำนักวิศวกรรมศาตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

โครงงาน	การจำลองระบบ MIMO ภายนอาคารในรูปกราฟฟิก
ผู้ดำเนินงาน	นายวิระเดช แพงแซง
	นายกิตติชัย มีหมู่
	นายธนากร จิรดงสวัสดิ์
อาจารย์ที่ปรึกษา	อาจารย์ ดร.พีระพงษ์ อุฑารสกุล
สาขาวิชา	วิศวกรรมโทรคมนาคม
ภาคการศึกษา	1/2550

บทคัดย่อ

ปัจจุบันการสื่อสารไร้สายมีการใช้อย่างแพร่หลาย ซึ่งเทคโนโลยีการสื่อสารไร้สายได้มีการพัฒนาไปในหลายรูปแบบตามจุดประสงค์และหน้าที่ที่ต้องการใช้งาน ซึ่งโครงงานนี้ต้องการศึกษาการทำงานและประสิทธิภาพของเทคโนโลยี MIMO ในการประมวลผลหาความจุของช่องสัญญาณภายในตัวอาคาร โดยทำการเขียนโปรแกรมและประมวลผลออกมาในรูปแบบกราฟฟิก ซึ่งสะดวกต่อผู้ใช้งาน ทำให้ผู้ใช้งานสามารถออกแบบและกำหนดการติดตั้งระบบ MIMO ภายในตัวอาคารได้

กิตติกรรมประกาศ

(Acknowledgement)

การทำโครงงานชิ้นนี้สำเร็จลงได้เพราะ คณะผู้จัดทำโครงงานได้รับความช่วยเหลือจากบุคคลหลานท่าน เริ่มจากอาจารย์ที่ปรึกษาโครงงาน อ.ดร.พีระพงษ์ อุฑารสกุล อาจารย์สาขาโทรคมนาคม ที่คอยดูแลและให้คำปรึกษาในด้านต่างๆแก่คณะผู้จัดทำอย่างใกล้ชิดมาโดยตลอดขอขอบคุณนายบัณฑิต ดวงชาทม นักศึกษาระดับบัณฑิตศึกษาสาขาไฟฟ้า ที่คอยให้คำปรึกษาเกี่ยวกับโปรแกรม MATLAB ขอขอบพระคุณท่านคณาอาจารย์และบุคคลากรของสาขาวิศวกรรมโทรคมนาคมทุกท่านที่คอยช่วยเหลือคณะผู้จัดทำ และขอขอบคุณเพื่อนพี่และน้องสาขาวิศวกรรมโทรคมนาคมที่คอยช่วยเหลือและคอยเป็นกำลังใจให้กันเสมอมา

สุดท้ายนี้คณะผู้จัดทำใคร่ขอกราบขอบพระคุณบิดา มารดา ที่ให้โอกาศทางด้านการศึกษาให้การเอาใจใส่เลี้ยงดู คอยเป็นกำลังใจและให้การสนับสนุนในทุกๆด้านมาโดยตลอด จึงเห็นสมควรที่จะมอบคุณความดีและเกียติคุณแก่ทุกๆท่านที่ได้กล่าวถึงมา ณ ที่นี้ด้วย

นายกิตติชัย มีหมู่

นายธนากร จิรดงสวัสดิ์

นายวิระเดช แพงแซง

สารบัญ

สารบัญรูป

บทที่1
บทนำ

1.1 ความเป็นมา

ปัจจุบันนี้เป็นยุคของเทคโนโลยีและการติดต่อสื่อสารที่ไร้พรมแดนมีการพัฒนเทคโนโลยีทางด้านเครือข่ายเพื่อให้คนทั่วโลกสามารถติดต่อสื่อสารกันได้จากในยุคแรก ๆ ที่เทคโนโลยีทางด้านเครือข่ายเป็นแบบแลนที่ต้องใช้สายในการติดต่อสื่อสารข้อมูลระหว่างคอมพิวเตอร์แต่ละเครื่อง แต่ในปัจจุบันได้มีการพัฒนาให้ทันสมัยมากยิ่งขึ้น โดยระบบเครือข่ายที่ปราศจากสาย สามารถเชื่อมต่อกันโดยไม่ต้องมีปัญหายุ่งยากเรื่องสายอีกต่อไป จากการใช้ระบบเครือข่ายไร้สายที่มีมากมาย ไม่ว่าจะเป็นในวงการธุรกิจที่นักธุรกิจมีความจำเป็นต้องใช้งานเครื่องคอมพิวเตอร์นอกสถานที่ที่ทำงานปกติการนำเสนองานยังบริษัทลูกค้า หรือการนำเครื่องคอมพิวเตอร์ติดตัวไปงานประชุมสัมมนาต่างๆ หรือในวงการศึกษาที่นักศึกษาในมหาวิทยาลัยสามารถใช้งานโน๊ตบุคเพื่อค้นคว้าข้อมูลในห้องสมุดของมหาวิทยาลัย แพทย์สามารถดึงข้อมูลมารักษาผู้ป่วยได้จากเครื่องคอมพิวเตอร์ หรือโน้ตบุ๊คที่เชื่อมต่อกับระบบเครือข่ายไร้สายได้ทันที จะเห็นได้ว่าเครือข่ายไร้สายมีการพัฒนาไปอย่างไม่หยุดยั้งรวมถึงตัวอุปกรณ์ต่างๆ ที่มีการผลิตออกมาเพื่อรองรับกับเทคโนโลยีที่มีการเปลี่ยนแปลงอยู่ตลอดเวลาทำให้ผู้ที่มีความสนใจที่ใช้เทคโนโลยีดังกล่าว จำเป็นต้องมีความรู้เกี่ยวกับอุปกรณ์ต่างๆ ที่ใช้ในการเชื่อมต่อเครือข่ายไร้สาย เพื่อจะได้สามารถใช้เครือข่ายไร้สายได้อย่างมีประสิทธิภาพมากที่สุด แต่ว่าจะเกิดปัญหาด้านสัญญาณภายในอาคารที่ยังไม่ดีพอ เป็นบริเวณที่อับสัญญาณเช่น บริเวณห้องที่อยู่ไกลจากตัว กระจายสัญญาณ(Access Point) ห้องที่มีความสลับซับซ้อนยากต่อสัญญาณจะเข้าไปถึง ปัญหานี้เราจะแก้ปัญหาโดยใช้ ระบบหลายอินพุตหลายเอาท์พุต (MIMO System) เข้ามาช่วยซึ่งระบบนี้จะมีตัวส่งหลายตัวและมีตัวรับหลายตัวทำให้การส่งสัญญาณได้เร็วขึ้น ปริมาณมากขึ้น แถมยังในพื้นที่ที่มีความสลับซับซ้อนยังรับสัญญาณได้ดีอีกด้วย

1.2 วัตถุประสงค์

1. เพื่อศึกษาระบบ MIMO ที่ใช้งานภายในอาคาร
2. เพื่อศึกษาทฤษฎีและหลักการของ Ray tracing
3. เพื่อศึกษาการเขียนโปรแกรมและการใช้ฟังก์ชัน GUI ในโปรแกรม MATLAB

1.3 ขอบเขตงาน

1. ออกแบบและจำลองระดับความแรงของสัญญาณไร้สายภายในอาคาร
2. ศึกษาระบบ MIMO ที่เกี่ยวข้องกับความจุของช่องสัญญาณโดยใช้ทฤษฎี Ray tracing
3. ศึกษาการใช้งานฟังก์ชัน GUI ในโปรแกรม MATLAB
4. เขียนโปรมแกรมเพื่อหาความจุของช่องสัญญาณและจุดที่เหมาะสมในการติดตั้งระบบ MIMO ภายในอาคาร

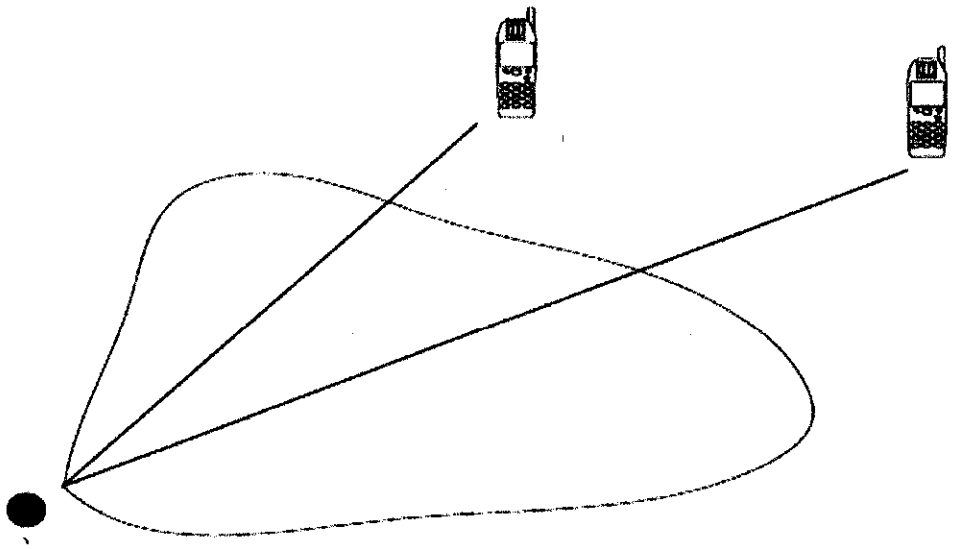
1.4 ขั้นตอนดำเนินงาน

1. ศึกษาและค้นคว้าเกี่ยวกับระบบหลายอินพุตหลายเอาท์พุตเข้า(MIMO System) ศึกษาและค้นคว้าทฤษฎีการตามรอยแสง(Ray tracing)
2. ศึกษาและค้นคว้าเกี่ยวกับโปรแกรม จี ยู ไอ (GUI)
3. ศึกษาและค้นคว้าเกี่ยวกับโปรแกรมแมทแลบ(MATLAB)
4. เขียนโปรแกรมแมทแลบ(MATLAB) เพื่อหาค่าของ (Capacity) และเขียนโปรแกรมแมทแลบ (MATLAB) เพื่อหาค่าของช่องสัญญาณ(Channal) โดยใช้ทฤษฎีการตามรอยแสง (Ray tracing) ในการหา
5. เขียนโปรแกรม จี ยู ไอ (GUI) เพื่อที่จะใช้แสดงผลและรับค่าต่างๆทั้งของระบบหลายอินพุตหลายเอาท์พุตเข้า(MIMOSystem) และการตามรอยแสง (Ray tracing)
6. รวมโปรแกรมทั้ง 3 เข้าด้วยกัน
7. ทดสอบโปรแกรมโดยโหลดแผนที่อาคารแล้วพร้อตกราฟแสดงค่าความแรงของสัญญาณ
8. เปรียบเทียบผลกับผลการทดลองอื่นๆ เพื่อยืนยันความถูกต้อง
9. ทดสอบการใช้งานโปรแกม
10. สรุปและวิเคราะห์การทำงาน

บทที่ 2
ทฤษฎีและหลักการ

2.1 ทฤษฎีและหลักการของ MIMO

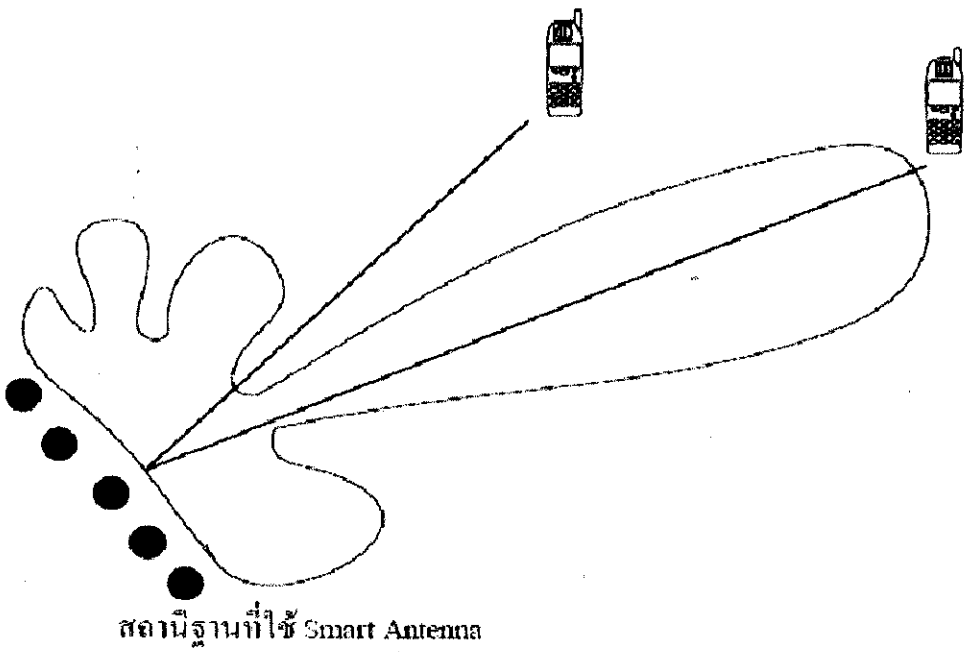
การที่ทำให้อัตราข้อมูลความเร็วสูงเข้าใกล้ Gbps หรือ 1,000,000,000 บิตต่อวินาที สำหรับโครงข่ายแลนไร้สาย (WLAN) นั้น นับว่าเป็นประเด็นหนึ่งที่น่าสนใจ การออกแบบข่ายเชื่อมโยงความเร็วสูงที่ให้คุณภาพของการบริการ (QoS) รวมถึงสามารถใช้งานในสิ่งแวดล้อมที่มีสิ่งกีดขวางถือเป็นหัวข้อหนึ่งที่ท้าทายนักวิจัยเป็นอย่างยิ่ง ณ ขณะนี้การใช้สายอากาศหลายตัว ณ ภาครับและภาคส่งเป็นที่รู้จักกันในนามของการสื่อสารไร้สายแบบ MIMO(Multiple-Input-Multiple-Output) ซึ่งถือว่าเป็นเทคโนโลยีที่มีประสิทธิผลด้านการลงทุนในการที่จะทำให้เกิดข่ายเชื่อมโยงไร้สายด้วยความเร็วขนาด Gbps ได้จริง



รูปที่ 1 ตัวอย่างสถานีฐานที่ใช้สายอากาศแบบเดิมและรูปแบบของการแพร่ของคลื่น [1]

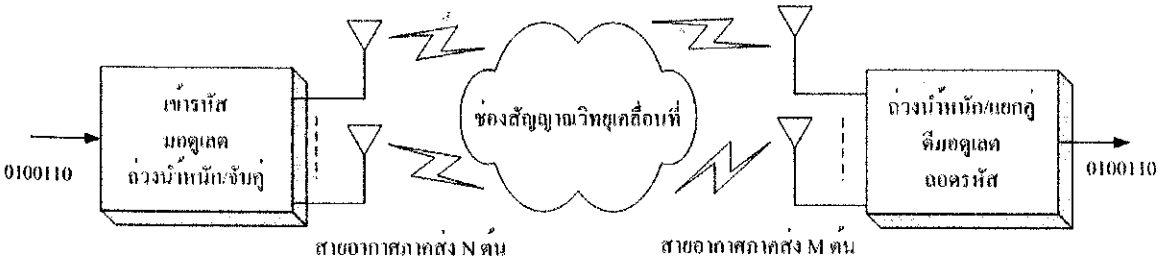
ในรูปที่ 1 ซึ่งมีรูปแบบของการแพร่ของคลื่นแบบไม่เจาะจงเป็นผลให้เกิดสมรรถนะการทำงานที่ไม่ดีนัก เช่น ไม่มีประสิทธิภาพในการใช้พลังงาน บิตผิดพลาดสูง เป็นต้น

ในรูปที่ 2 แสดงการทำงานของ Smart Antenna ซึ่งสามารถตอบคำถามได้ว่าการใช้สายอากาศหลายต้นนั่น คือ การใช้สายอากาศหลายต้นสามารถปรับทิศทางของลำคลื่นให้เป็นไปตามที่ต้องการได้ นับว่าเป็นข้อได้เปรียบเมื่อเปรียบเทียบกับการใช้สายอากาศแบบเดิมดังแสดงใน



รูปที่ 2 ตัวอย่างสถานีฐานที่ใช้ Smart Antenna (สายอากาศหลายต้น) และรูปแบบของการแพร่ของคลื่น [1]

ด้วยหลักการ Smart Antenna ข้างต้นนำไปสู่การพสื่อสารไร้สายแบบ MIMO โดยแนวความคิดเริ่มต้นของการสร้างระบบ MIMO คือ ผลรวมของสัญญาณจากภาคส่งหรือภาครับอันเนื่องมาจากสายอากาศหลายต้นทำให้คุณภาพของข้อมูลหรือบิตผิดพลาดน้อยลง อีกนัยหนึ่งคือทำให้อัตรารับส่งข้อมูลสูงขึ้นระบบ MIMO มีความสามารถทนต่อสัญญาณรบกวนได้เป็นอย่างดี เนื่องจากมีการชดเชยสัญญาณที่ขาดหายไปด้วยสายอากาศต้นอื่นได้ โดยบล็อกไดอะแกรมของระบบ MIMO สามารถแสดงได้ดังรูปที่ 3



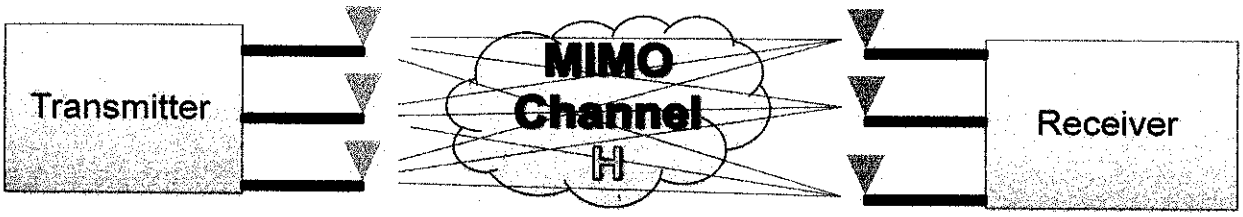
รูปที่ 3 บล็อกไดอะแกรมระบบสื่อสารไร้สายแบบ MIMO [1]

จากรูปที่ 3 สัญญาณไบนารีซึ่งในที่นี้คือ 0100110 ส่งผ่านเข้าไปในภาคส่ง ผ่านการเข้ารหัส การมอดูเลตด้วยวิธีการต่าง ๆ เช่น QPSK (Quaternary Phase-Shift Keying), M-QAM (Multilevel-Quadrature AmplitudeModulation) หรือวิธีการอื่น ๆ ซึ่งทำให้เกิดการแยกสัญญาณ โดยที่แต่ละสัญญาณจะได้รับการจับคู่กับแต่ละสายอากาศ และการจับคู่จะมีการถ่วงน้ำหนักแก่สายอากาศแต่ละต้นเมื่อปรับความถี่ให้สูงขึ้น กรองและขยายสัญญาณ แล้วนำสัญญาณส่งผ่านช่องสัญญาณ โดยที่ภาครับสัญญาณจะถูกดึงออกโดยสายอากาศแต่ละต้นตามค่าถ่วงน้ำหนัก ดีมอดูเลต และถอดรหัสเพื่อให้ได้สัญญาณเดิมกลับมา ทั้งนี้การเลือกใช้วิธีการเข้ารหัสและวิธีการจับคู่มีได้หลากหลายขึ้นกับการประยุกต์ใช้งานปัจจุบันมีหลายหน่วยงานที่พยายามสร้างระบบแลนไร้สายบรอดแบนด์ ที่มีอัตรารับส่งข้อมูลในหน่วยGbps สำหรับสิ่งแวดล้อมที่มีสิ่งกีดขวาง นอกจากนี้ยังมีการกำหนดใช้ MIMO สำหรับแลนไร้สายมาตรฐานIEEE 802.11 โดยกลุ่มที่เรียกว่า WNG (Wireless Next Generation) อย่างไรก็ตามวิวัฒนาการสู่การสื่อสารไร้สายความเร็วสูงยังคงขึ้นอยู่กับฮาร์ดแวร์ด้านความถี่วิทยุหลายช่องสัญญาณและด้านกรรมวิธีสัญญาณแบบดิจิทัล (DSP: Digital Signal Processing)

ทฤษฎี capacity

ระบบ MIMO มีการส่งแบ่งสัญญาณไปแต่ละเสานี่เป็นสาเหตุที่ทำให้ระบบนี้สามารถส่งแต่สิ่งที่จะบอกว่ามันสามารถส่งข้อมูลได้มากเท่าไรคือ capacity ค่า Capacity ของระบบ MIMO เป็นค่าความจุของช่องสัญญาณสามารถหาค่าได้โดยเริ่มจาก

1. ช่องสัญญาณของ MIMO



รูปที่ 4 แสดงถึงช่องสัญญาณ H ของระบบ MIMO

โดยที่กำหนดช่องสัญญาณเป็น H อยู่ในรูปของแมตทริก จะได้

$$H(t) = \begin{bmatrix} H^{11}(t) & H^{12}(t) & \cdots & H^{1M_t}(t) \\ H^{21}(t) & H^{22}(t) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ H^{M_r 1}(t) & \cdots & \cdots & H^{M_r M_t}(t) \end{bmatrix}_{M_r \times M_t}$$

โดยที่ M_t คือ จำนวนของสายอากาศที่เป็นตัวส่ง

M_r คือ จำนวนของสายอากาศที่เป็นตัวรับ

2. capacity

จาก ช่องสัญญาณ H นำมาแทนในสูตร capacity ของ MIMO ได้ดังนี้

$$capacity = \log_2[\det(I_r + SNR*H* H^{t})] \qquad [2]$$

โดยที่

I_r คือ แมตทริกเอกลักษณ์

SNR คือ signal to noise ratio ที่เป็นค่าเฉลี่ยของเสาแต่ละต้น

H คือ ช่องสัญญาณที่อยู่ในรูปของ complex มีขนาดเท่ากับ ที่ M_r x M_t

H^{t} คือ conjugate และtranspost

2.2ทฤษฎีและหลักการของ Ray tracing

ที่มาในการใช้ทฤษฎี Ray tracing

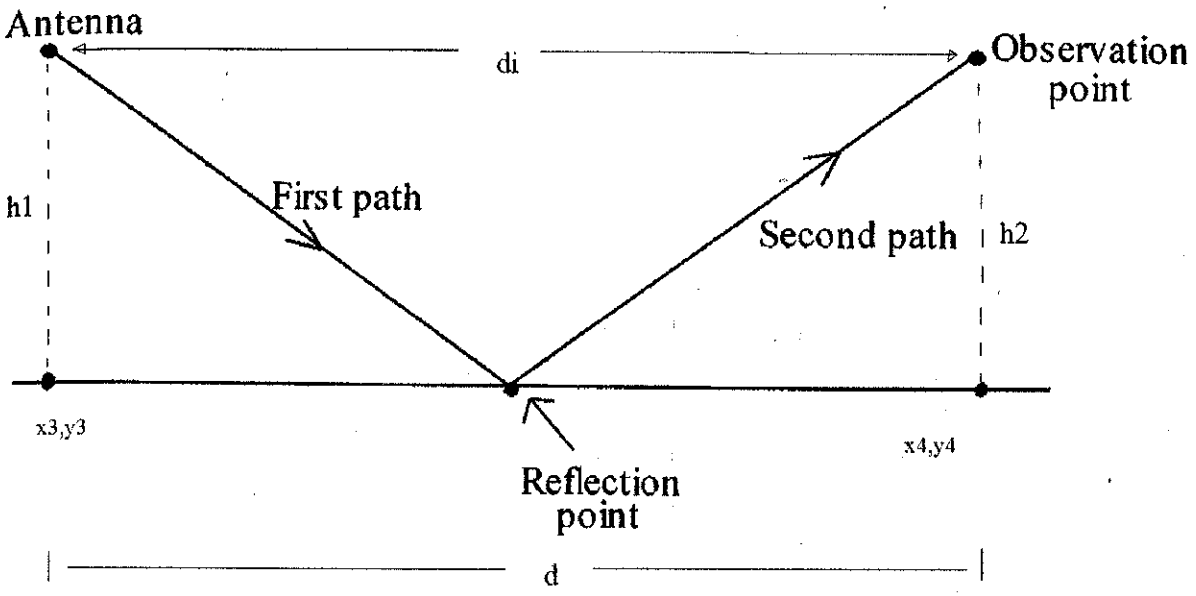
ปัจจุบันการวัดสัญญาณภายในอาคารจะทำการวัดเฉพาะความแรงของสัญญาณ แต่เทคโนโลยี MIMO ที่นำมาใช้ในโครงงานนี้จำเป็นต้องใช้ทั้งแอมพลิจูดและเฟสเพื่อใช้ในการนำไปหาความจุของช่องสัญญาณ ดังนั้นโครงงานนี้จึงได้นำทฤษฎี Ray tracing เพื่อมารองรับความต้องการของเทคโนโลยี MIMO

ทฤษฎี Ray tracing ประกอบด้วยส่วนหลักๆ อยู่ 3 ส่วน คือ

2.2.1 Transmission coefficients สัมประสิทธิ์การส่งผ่าน เป็นค่าสัมประสิทธิ์ที่เกิดจากที่สัญญาณคลื่นวิ่งผ่านสิ่งกีดขวาง ซึ่งค่าสัมประสิทธิ์ที่ได้มาจะขึ้นอยู่กับสิ่งกีดขวางนั้นๆ โดยกำหนดค่าสัมประสิทธิ์ที่ใช้ในโครงงานดังนี้

- สัมประสิทธิ์การส่งผ่านของกำแพงเท่ากับ 0.3
- สัมประสิทธิ์การส่งผ่านของกำพนังกั้นเท่ากับ 0.237
- สัมประสิทธิ์การส่งผ่านของประตูเท่ากับ 0.155
- สัมประสิทธิ์การส่งผ่านของหน้าต่างเท่ากับ 0.09 **[3]**

2.2.2 Reflection coefficients สัมประสิทธิ์การสะท้อน เป็นค่าสัมประสิทธิ์ที่เกิดจากการที่สัญญาณคลื่นตกกระทบกับสิ่งกีดขวางและสะท้อนกลับมา ซึ่งสัญญาณคลื่นที่สะท้อนกลับมาจะถูกลดทอนจากสัมประสิทธิ์การสะท้อน



รูปที่ 5 การสะท้อนของ Reflection coefficients

การหาสัมประสิทธิ์การสะท้อน

1) ทำการหาค่า x_3, y_3กับ x_4, y_4โดย

1.1) ทำการหาความชันของแนวเส้นจากจุดส่งไปจุดรับ และความชันของแนวสิ่งกีดขวาง จาก

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

1.2) หาความชันของเส้นที่ตั้งฉาก จาก

$$m' = -\frac{1}{m}$$

1.3) ทำการหาค่าคงที่สมการเส้นตรง จาก

$$y = mx + c$$

1.4) ทำการแก้สมการหาค่า x_3, y_3 กับ x_4, y_4 โดยจัดให้อยู่ในรูป

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

2) ทำการหาระยะต่างๆ

2.1) หา d_i จาก

$$d_i = \sqrt{\left((x_1 - x_2)^2 + (y_1 - y_2)^2\right)} \quad [4]$$

2.2) หา h_1 จาก

$$h_1 = \sqrt{\left((x_1 - x_3)^2 + (y_1 - y_3)^2\right)} \quad [4]$$

2.3) หา h_2 จาก

$$h_2 = \sqrt{\left((x_2 - x_4)^2 + (y_2 - y_4)^2\right)} \quad [4]$$

2.4) หา d จาก

$$d = \sqrt{\left((x_3 - x_4)^2 + (y_3 - y_4)^2\right)} \quad [4]$$

2.5) หา r จาก

$$r = \sqrt{\left((h_1 + h_2)^2 + d^2\right)} \quad [4]$$

r คือระยะความยาวทั้งหมดของคลื่นสะท้อน

2.6) หา D จาก

$$D = r - d_i$$

D คือระยะผลต่างระหว่างคลื่นสะท้อนกับคลื่นส่งผ่าน

นำค่าที่ได้มาหา ω จาก

$$\psi = 2\pi\left(\frac{D}{\lambda}\right) \quad [5]$$

และนำค่า ω ที่ได้มาหาสัมประสิทธิ์การสะท้อน จาก

$$R = \frac{\cos(\psi_i) - \left(\varepsilon - \sin^2(\psi_i)\right)^{\frac{1}{2}}}{\cos(\psi_i) + \left(\varepsilon - \sin^2(\psi_i)\right)^{\frac{1}{2}}} \quad [5]$$

โดยที่ค่า ε หาได้จาก

$$\varepsilon = \varepsilon_0 + j60\sigma\lambda \quad [4]$$

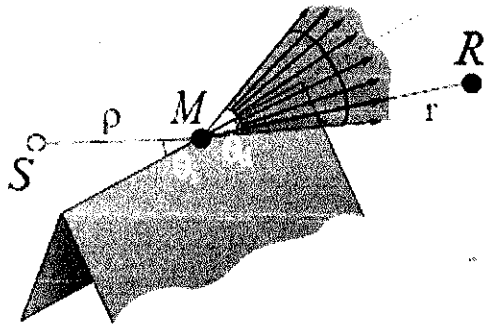
σ คือค่า Conductivity ของชนิดสิ่งกีดขวางที่ได้ข้อมูลจากงานวิจัย [5]

ε_0 คือค่า Relative Permittivity ของชนิดสิ่งกีดขวางที่ได้ข้อมูลจากงานวิจัย [5]

d_i คือระยะระหว่างตัวส่งถึงตัวรับ

r คือผลต่างของระยะทางระหว่างคลื่นสะท้อนกับคลื่นส่งผ่าน

2.2.3Diffraction coefficients สัมประสิทธิ์การเลี้ยวเบนเป็นค่าสัมประสิทธิ์ที่เกิดขึ้นเมื่อ สัญญาณคลื่นวิ่งไปกระทบกับเหลี่ยมหรือมุมของสิ่งกีดขวาง ทำให้เกิดการเลี้ยวเบนไปในหลายทิศทาง ซึ่งจะมีบางทิศทางที่เลี้ยวเบนมาทางตัวรับ ทำให้รับสัญญาณได้



รูปที่ 6 การสะท้อนของ Diffraction coefficients

สัมประสิทธิ์การเลี้ยวเบนสามารถหาได้จาก

$$D(n,k,p,r,\theta_i,\alpha_i,\alpha_d) = -\left(\frac{e^{-i\frac{\pi}{4}}}{2n\sqrt{2k\pi}\sin\theta_i}\right)$$

$$\left[\tan^{-1}\left(\frac{\pi+(\alpha_d-\alpha_i)}{2n}\right)F\left(kLa+(\alpha_d-\alpha_i)\right)+\tan^{-1}\left(\frac{\pi-(\alpha_d-\alpha_i)}{2n}\right)F\left(kLa-(\alpha_d-\alpha_i)\right)\right.$$

$$\left.+\left\{\tan^{-1}\left(\frac{\pi+(\alpha_d+\alpha_i)}{2n}\right)F\left(kLa+(\alpha_d+\alpha_i)\right)+\tan^{-1}\left(\frac{\pi-(\alpha_d+\alpha_i)}{2n}\right)F\left(kLa-(\alpha_d+\alpha_i)\right)\right\}\right]$$

[6]

เมื่อ $k = \left(\frac{2\pi}{\lambda} \right)$

n เป็นมุมของที่กีดขวาง

p ระยะทางจากจุดส่งถึงจุดเลี้ยวเบน

r ระยะทางจากจุดรับถึงจุดเลี้ยวเบน

θ_i มุมกวาด

α_i มุมแงย

α_d มุมแงยที่เกิดจากการเลี้ยวเบน

และเมื่อ $$F(x) = 2i\sqrt{x}e^{ix} \int_{\sqrt{x}}^{+\infty} e^{-ir^2} dr$$

$$L = \frac{pr}{p+r} \sin^2 \theta_i$$

$$a\pm(\beta) = 2\cos^2\left(\frac{2\pi nN \pm -\beta}{2} \right)$$

และ $$2\pi nN^+ - \beta = \pi \quad ; \quad 2\pi nN^- - \beta = -\pi \qquad [6]$$

เมื่อได้ค่าสัมประสิทธิทั้งหมดแล้วก็สามารถหากำลังที่รับได้ที่ตัวรับจากสมการ

$$E = E_0\left(\frac{4\pi d}{\lambda}\right)(\Pi Tr)(\Pi R)(\Pi D)$$

E_0 กำลังงานที่ใช้ส่ง

d ระยะห่างระหว่างจุดที่ตั้งฉากจุดแรกกับจุดที่ 2

E กำลังงานที่รับได้

Tr สัมประสิทธิ์การส่งผ่าน

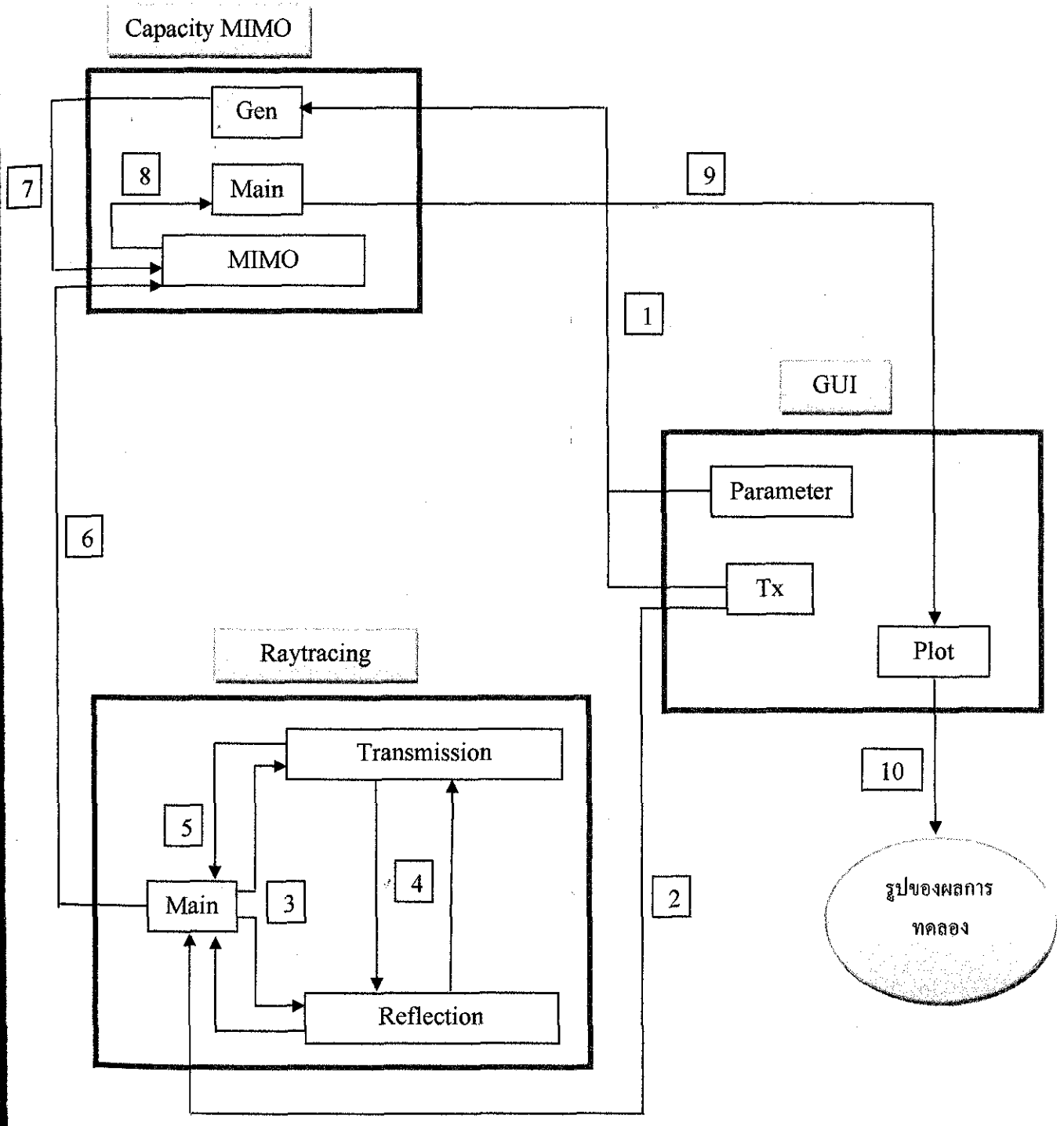
R สัมประสิทธิ์การสะท้อน

D สัมประสิทธิ์การเลี้ยวเบน

บทที่ 3

การทำงานของของโปรแกรมและโครงสร้างของโปรแกรม

Block การทำงานของโปรแกรมการจำลองระบบ MIMO ภายในอาคารในรูปแบบกราฟฟิก



รูปที่ 7 ขั้นตอนการทำงานของโปรแกรม

3.1 โปรแกรม MIMO

โปรแกรมรับค่าต่างๆ ของช่องสัญญาณ H

```
function H = genH(Nt,Nr,angle,dt,dr,Tx,Rx)
load wall.mat
load Tx.mat
load parameter.mat
H = zeros(Nr,Nt);
 for i = 1:Nt

        Xt = (i-1)*dt*cos (angle*180/pi)+Tx(1);
        Yt = (i-1)*dt*sin (angle*180/pi)+Tx(2);
    for j = 1:Nr

        Xr = (j-1)*dr*cos (angle*180/pi)+Rx(1);
        Yr = (j-1)*dr*sin (angle*180/pi)+Rx(2);
        H(j,i) = RayTracing(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,...
                 rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,[Xt Yt],[Xr Yr]);
    end
 end
```

Nt ,Nr คือ จำนวนตัวรับและตัวส่ง

angle คือ มุมที่ทำกับแผนที่ที่เป็นของตัวส่ง

dt ,dr คือ ระยะห่างระหว่างสายอากาศทั้งตัวส่งและตัวรับ

โปรแกรมประมวลผลค่า Capacity ที่มาจากสูตรของ MIMO

```
function Capacity = MIMO(snr_DB,H)
                Ir = eye(size(H));
               snr = 10^(snr_DB/10);
          Capacity = log2(det(Ir + snr*H*H'));
```

SNR_DB คือ signal to noise ratio ที่มีหน่วยเป็น db

H คือ ช่องสัญญาณ H

โปรแกรมหลักรับค่าพิกัดทั้งหมดที่ป้อนเข้ามา และประมวลผลเก็บค่า Capacity

```
function [xC yC C] = main()
load wall.mat
load Tx.mat
load parameter.mat
xmin = min([bwx1(2:length(bwx1)),bwx2(2:length(bwx2)),...
       gwx1(2:length(gwx1)),gwx2(2:length(gwx2)),...
       rwx1(2:length(rwx1)),rwx2(2:length(rwx2)),...
       ywx1(2:length(ywx1)),ywx2(2:length(ywx2))]);
xmax = max([bwx1(2:length(bwx1)),bwx2(2:length(bwx2)),...
       gwx1(2:length(gwx1)),gwx2(2:length(gwx2)),...
       rwx1(2:length(rwx1)),rwx2(2:length(rwx2)),...
       ywx1(2:length(ywx1)),ywx2(2:length(ywx2))]);
ymin = min([bwy1(2:length(bwy1)),bwy2(2:length(bwy2)),...
      gwy1(2:length(gwy1)),gwy2(2:length(gwy2)),...
      rwy1(2:length(rwy1)),rwy2(2:length(rwy2)),...
      ywy1(2:length(ywy1)),ywy2(2:length(ywy2))]);
ymax = max([bwy1(2:length(bwy1)),bwy2(2:length(bwy2)),...
      gwy1(2:length(gwy1)),gwy2(2:length(gwy2)),...
      rwy1(2:length(rwy1)),rwy2(2:length(rwy2)),...
      ywy1(2:length(ywy1)),ywy2(2:length(ywy2))]);
ix=0;
for xr = xmin:1:xmax
    ix=ix+1;
    xC(ix)=xr;
    iy=0;
        for yr = ymin:1:ymax
            iy=iy+1;
            yC(iy)=yr;
    H = genH(Nt,Nr,Angle,Dt,Dr,[xt yt],[xr yr]);
    C(iy,ix)  = MIMO(Snr_dB,H);
        end
    end
```

3.2โปรแกรมในส่วนของ Ray tracing

ในส่วนของ Ray tracing นั้นจะแบ่งโปรแกรมออกเป็น 3 ส่วนคือ

- Ray tracing
- tran
- reflec

และแบ่งการทำงานเป็นดังนี้

โปรแกรม Ray tracing

ทำหน้าที่รับค่าตัวแปรของสิ่งกีดขวางในแนวพิกัด X, Y ตำแหน่งตัวส่งและตำแหน่งตัวรับ, ทำการส่งค่าพิกัดของแนวสิ่งกีดขวางและตำแหน่งตัวส่งตัวรับไปให้โปรแกรมในส่วนของ tran และ reflec และทำหน้าที่รับค่ากำลังงานที่รับได้จากทั้งสองโปรแกรม แล้วนำมารวมกันก่อนจะส่งค่า H (กำลังงานที่รับได้ที่ตัวรับ) ไปให้โปรแกรม genH

โปรแกรม tran

เป็นโปรแกรมทำการรับค่าตัวแปรของสิ่งกีดขวางและตำแหน่งตัวส่งตัวรับจาก Ray tracing และรับค่าตำแหน่งจุดสะท้อนจาก reflec แล้วจากนั้นก็ทำการประมวลผลหาว่าแต่ละกรณีนี้นั้นจะเกิดสัมประสิทธิ์การส่งผ่านหรือไม่ โดยทำการวนรับค่าตัวแปรของสิ่งกีดขวาง, ตำแหน่งตัวส่งตัวรับและตำแหน่งของจุดสะท้อน จากนั้นก็ตรวจสอบว่าแนวเส้นที่ลากจากตัวส่งไปที่จุดรับจุดส่งไปที่จุดสะท้อนและจุดสะท้อนไปยังตัวรับ กับแนวเส้นของสิ่งกีดขวางว่ามีการตัดกันหรือไม่ ถ้ามีการตัดกันแสดงว่ามีสัมประสิทธิ์การส่งผ่าน โดยจะแบ่งสิ่งกีดขวางออกเป็น 4 ประเภท ซึ่งแต่ละประเภทจะมีสัมประสิทธิ์การส่งผ่านที่แตกต่างกัน โดยจะกำหนดค่าเป็นดังนี้

- สัมประสิทธิ์การส่งผ่านของกำแพงเป็น 0.3
- สัมประสิทธิ์การส่งผ่านของพนังกั้นเป็น 0.237
- สัมประสิทธิ์การส่งผ่านของประตูเป็น 0.155
- สัมประสิทธิ์การส่งผ่านของหน้าต่างเป็น 0.09 [3]

และเมื่อได้ค่าสัมประสิทธิ์ในแต่ละกรณีแล้วก็จะทำการหากำลังงานที่รับได้ที่ตัวรับของในกรณีนั้นๆ โดยนำสัมประสิทธิ์การส่งผ่านที่ได้ทั้งหมดในกรณีนั้นๆ มาคูณกันทั้งหมดแล้วนำไปคูณกับสมการ Free-space loss และกำลังที่ทำการส่ง จะได้สมการดังนี้

$$E_{Tr} = E_0 \left(\frac{4\pi d}{\lambda} \right) \Pi Tr$$

และจากนั้นก็จะส่งค่ากำลังงานที่ได้กลับไปให้โปรแกรม Ray tracing สำหรับกรณี direct wave (ระยะทางจากตัวส่งถึงตัวรับ) และส่งไปให้โปรแกรม reflec สำหรับกรณี reflec wave (ระยะทางที่มีการสะท้อน)

โปรแกรม reflec

ทำการรับค่าตัวแปรสิ่งกีดขวางและตำแหน่งตัวส่งตัวรับจาก Ray tracing จากนั้นจะทำการหาว่ามีการสะท้อนหรือไม่ โดยใช้วิธีการเดียวกับการหาจุดตัดในส่วนของ tran และเมื่อทราบว่ามีจุดสะท้อนก็ส่งค่าตำแหน่งจุดสะท้อนนั้นไปให้ tran และรับค่ากำลังงานที่ประมวลผลได้ตั้งแต่จุดส่งไปจุดสะท้อนและจุดสะท้อนไปจุดรับจาก tran กลับมา และนำค่าที่ได้นั้นไปรวมกับกำลังงานที่ได้จากการสะท้อน ซึ่งสัมประสิทธิ์การสะท้อนจะหาได้จากสมการ

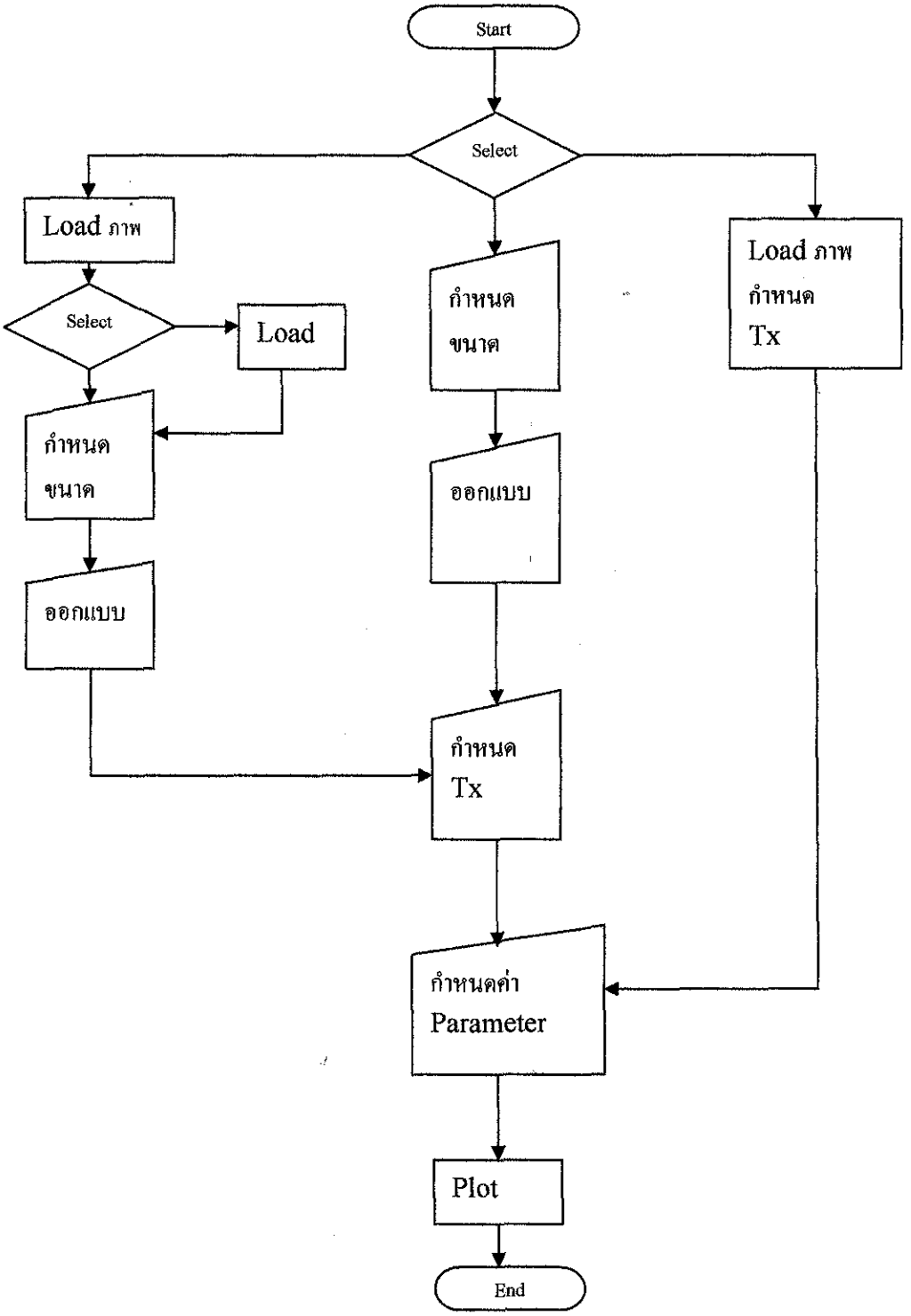
$$R = \frac{\cos(\psi_i) - \left(\varepsilon - \sin^2(\psi_i)\right)^{\frac{1}{2}}}{\cos(\psi_i) + \left(\varepsilon - \sin^2(\psi_i)\right)^{\frac{1}{2}}}$$

และหากำลังที่รับได้ที่ตัวรับจากสมการ

$$E_R = E_0 \left(\frac{4\pi d}{\lambda}\right) \Pi R + E_{Tr}$$

ซึ่งในสมการจะทำการรวมค่ากำลังงานที่ได้จะโปรแกรม tran (E_{Tr}) ในส่วนของ reflec wave ไปด้วย จะทำให้ได้กำลังงานที่ตัวรับที่เป็นส่วนของคลื่นสะท้อนทั้งหมด แล้วส่งค่าที่ได้ไปให้โปรแกรม Ray tracing

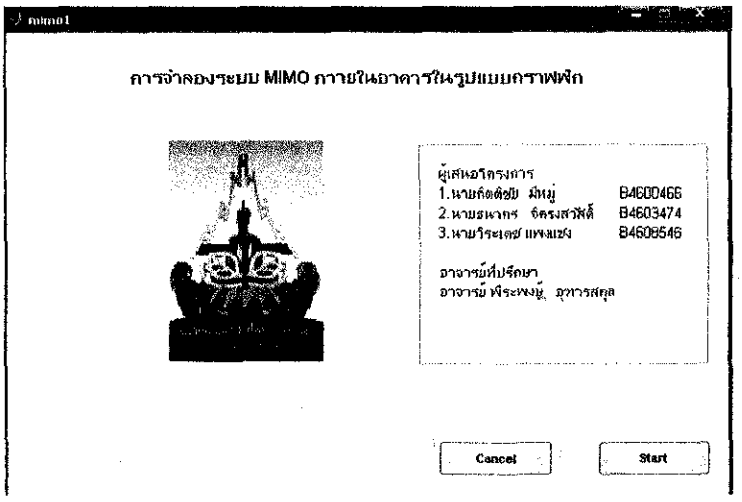
3.3 ขั้นตอนการทำงานของโปรแกรมในส่วน GUI



รูปที่ 8 ขั้นตอนการทำงานของโปรแกรม (GUI)

รูปแบบของหน้าต่างการทำงานของโปรแกรม

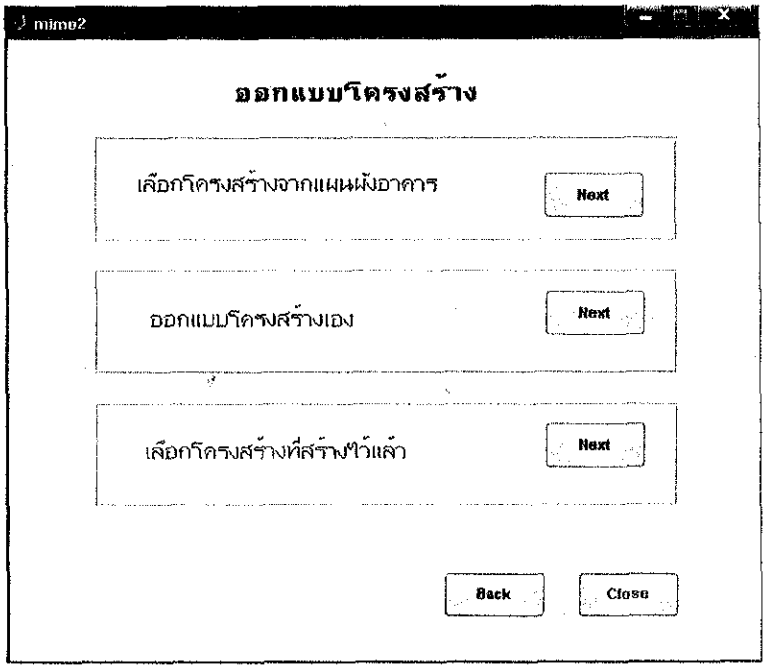
เริ่มใช้งานโปรแกรม



รูปที่ 9 เริ่มการทำงานของโปรแกรม

- เลือก Start เพื่มการทำงานของโปรแกรม
- เลือก Cancel เพื่อเลิกใช้งานโปรแกรม

3.3.1 เลือกวิธีการออกแบบโครงสร้าง

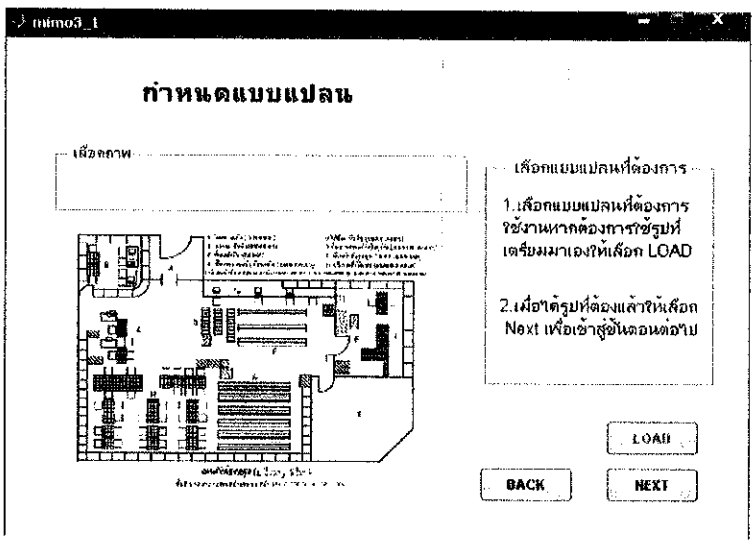


รูปที่ 10 เลือกการออกแบบโครงสร้าง

- เลือกโครงสร้างจากแผนผังอาคา
- เลือกออกแบบโครงสร้างเอง
- เลือกโครงที่สร้างไว้แล้ว
- เลือก Close เพื่อยกเลิกการทำงาน
- เลือก Back เพื่อกลับไปหน้าแรก

3.3.2 เมื่อเลือกโครงสร้างจากแผนผังอาคาร

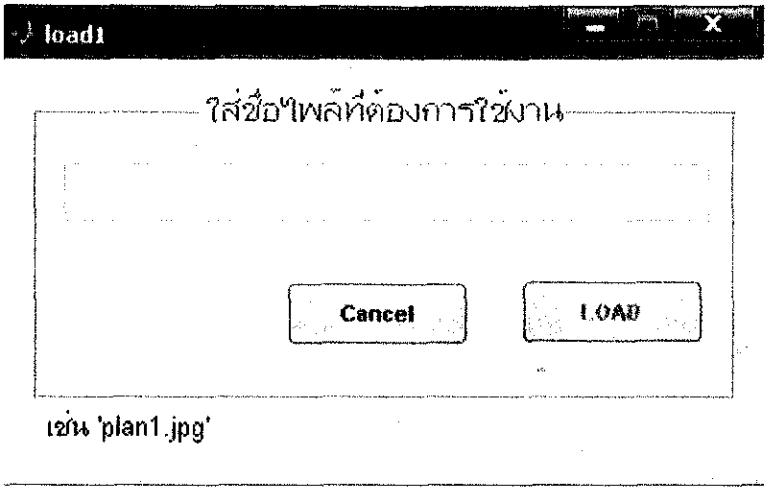
1. เลือกโครงสร้าง



รูปที่ 11 เลือกภาพโครงสร้างที่ต้องการ

- เลือกภาพจากข้อมูลที่มีอยู่แล้วใน Pop-up เมนู หรือเลือกภาพจากโครงงานที่ไม่มีในโปรแกรมโดยเลือก Load
- เลือก Next เพื่อเข้าสู่ขั้นตอนถัดไป
- เลือก Backเพื่อย้อนกลับไปหน้าก่อนหน้านี้

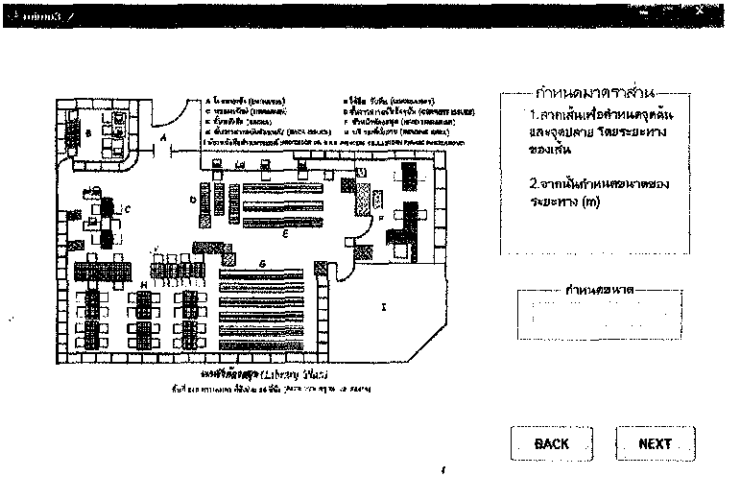
2. เมื่อเลือกภาพจากโครงสร้างที่ไม่มีในโปรแกรม



รูปที่ 12 โหลดภาพจากไฟล์อื่น

- ใส่ข้อมูลของภาพที่ต้องการใช้งาน โดยภาพที่ต้องการใช้นั้นต้องมีนามสกุล jpg หรือ tif เท่านั้นและต้องอยู่ใน Folder เดียวกันกับตัวโปรแกรม(ผู้ใช้งานต้อง Copy มาไว้ใน C:\Program Files\MATLAB\R2006a\work\mimo) เช่น 'plan1.jpg' เป็นต้น
- เลือก Load เพื่อใช้งานภาพโครงสร้างที่ต้องการ
- เลือก Cancel เพื่อยกเลิกการใช้งานจากเมนู Load

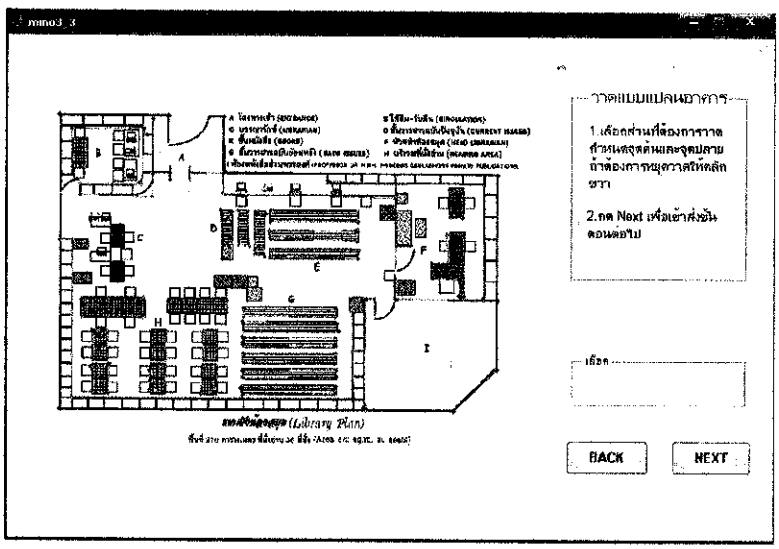
3. กำหนดมาตราส่วน



รูปที่ 13 กำหนดมาตราส่วน

- ลากเส้นเพื่อกำหนดจุดต้นและจุดปลายของขนาดมาตราส่วน
- กำหนดขนาดของระยะทางในช่องว่าง (มีหน่วยเป็น m)
- เลือก Next เพื่อเข้าสู่ขั้นตอนต่อไป
- เลือก Back เพื่อย้อนกลับไปหน้าก่อนหน้านี้

4. วาดโครงสร้างตามภาพโครงสร้างที่เลือกมา

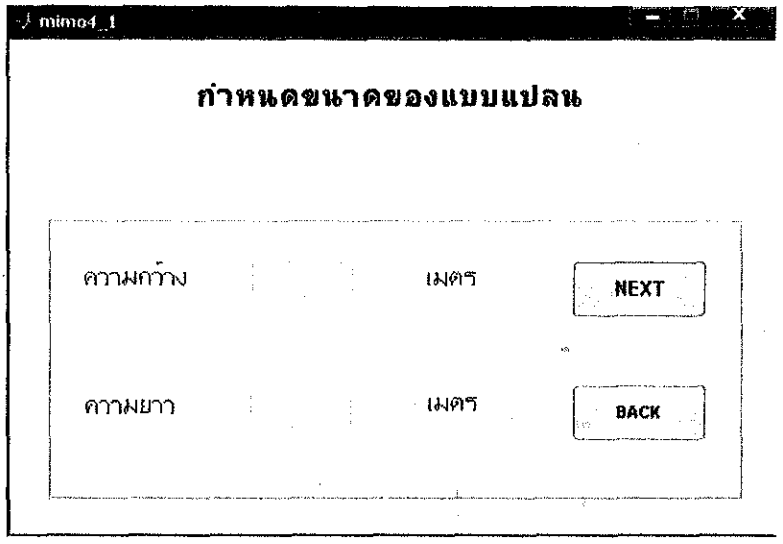


รูปที่ 14 ออกแบบโครงสร้างตามภาพ

- เลือกรูปแบบของส่วนต่างๆ ที่จะออกแบบโดยเลือกในเมนู Pop-up จะมีให้เลือก 4 ส่วนคือ ส่วนของผนังคอนกรีต (สีน้ำเงิน) ส่วนของผนังกั้นห้องที่เป็นพลาสติก (สีเขียว) ส่วนของหน้าต่าง (สีแดง) และส่วนของประตู (สีแดง)
- ลากเส้นจากจุดต้นไปยังจุดปลายเพื่อกำหนดระยะทางของเส้น
- เลือก Next เพื่อเข้าสู่ขั้นตอนถัดไป
- เลือก Back เพื่อย้อนกลับไปหน้าก่อนหน้านี้

3.3.3 เมื่อเลือกออกแบบโครงสร้างเอง

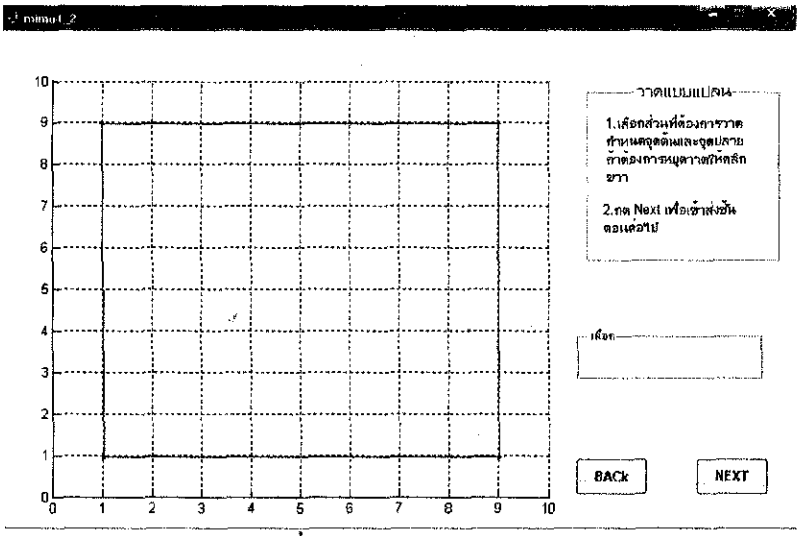
1. กำหนดขนาดของโครงสร้าง



รูปที่ 15 กำหนดขนาดของโครงสร้าง

- กำหนดขนาดของโครงสร้างทั้งความกว้างและความยาวโดยมีหน่วยเป็นเมตร (m)
- เลือก Next เพื่อเข้าสู่ขั้นตอนต่อไป
- เลือก Back เพื่อย้อนกลับไปหน้าก่อนหน้านี้

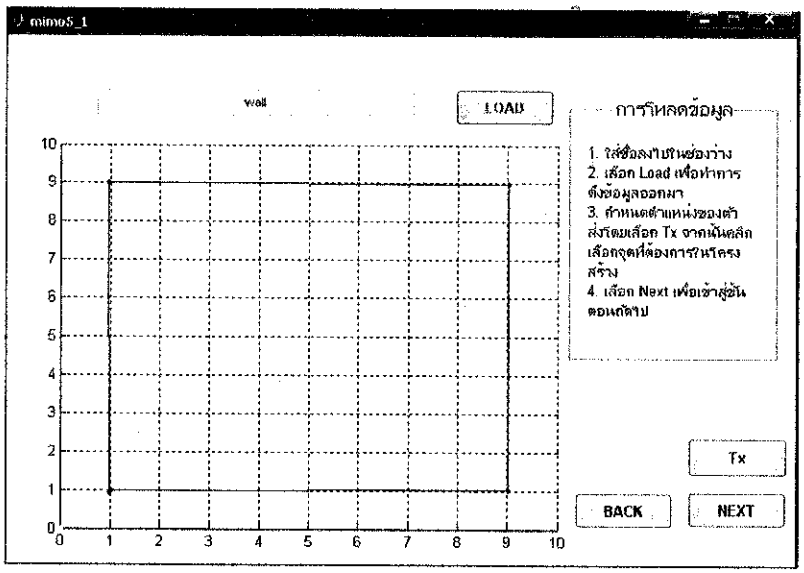
2. ออกแบบโครงสร้าง



รูปที่ 16 ออกแบบโครงสร้าง

- เลือกรูปแบบของส่วนต่างๆ ที่จะออกแบบโดยเลือกในเมนู Pop-up จะมีให้เลือก 4 ส่วนคือ ส่วนของผนังคอนกรีต (สีน้ำเงิน) ส่วนของผนังกั้นห้องที่เป็นพลาสติก (สีเขียว) ส่วนของหน้าต่าง (สีแดง) และส่วนของประตู (สีแดง)
- ลากเส้นจากจุดต้นไปยังจุดปลายเพื่อกำหนดระยะทางของเส้น
- เลือก Next เพื่อเข้าสู่ขั้นตอนถัดไป
- เลือก Back เพื่อย้อนกลับไปหน้าก่อนหน้านี้

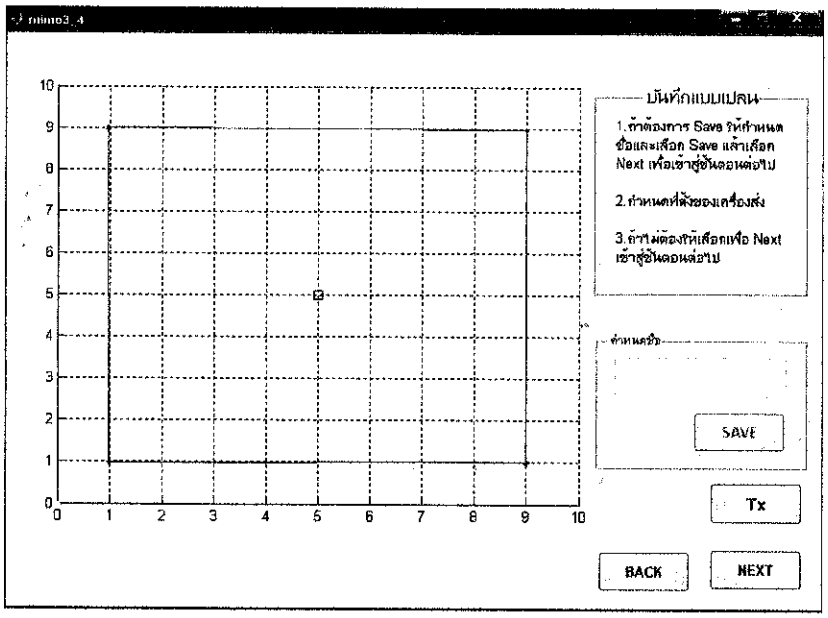
3.3.4 เมื่อเลือกโครงสร้างที่ออกแบบไว้แล้ว



รูปที่ 17 โหลดโครงสร้างที่สร้างไว้แล้ว

- ใส่ชื่อลงในช่องว่างโดยชื่อที่ใส่ต้องมีอยู่ในฐานข้อมูล
- เลือก Load เพื่อทำดึงข้อมูลที่บันทึกไว้ออกมาใช้
- เลือก Next เพื่อเข้าสู่ขั้นตอนถัดไป
- เลือก Back เพื่อย้อนกลับไปหน้าก่อนหน้านี้

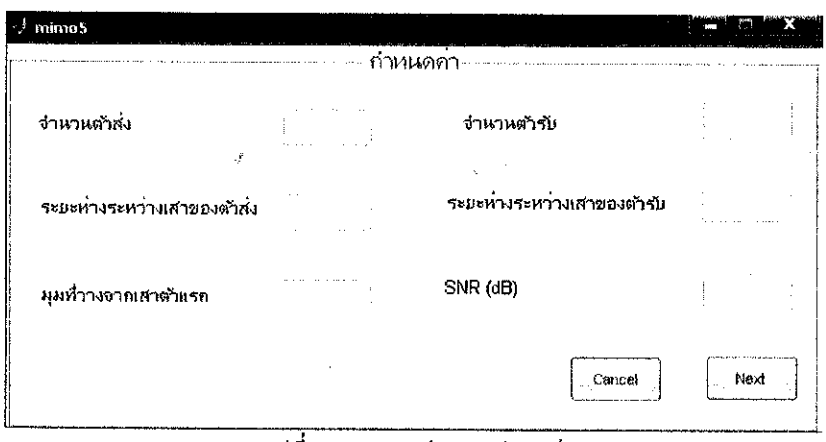
3.3.5 บันทึกโครงสร้างและกำหนดตำแหน่งของตัวส่ง



รูปที่ 18 บันทึกและกำหนดตำแหน่ง

- ถ้าต้องการบันทึกโครงสร้างที่ออกแบบมาให้ใส่ชื่อลงในช่องว่างแล้วเลือก save
- เลือก Tx เพื่อกำหนดตำแหน่งของตัวส่งโดยคลิกจุดที่ต้องลงในโครงสร้าง
- เลือก Next เพื่อเข้าสู่ขั้นตอนถัดไป
- เลือก Back เพื่อย้อนกลับไปหน้าก่อนหน้านี้

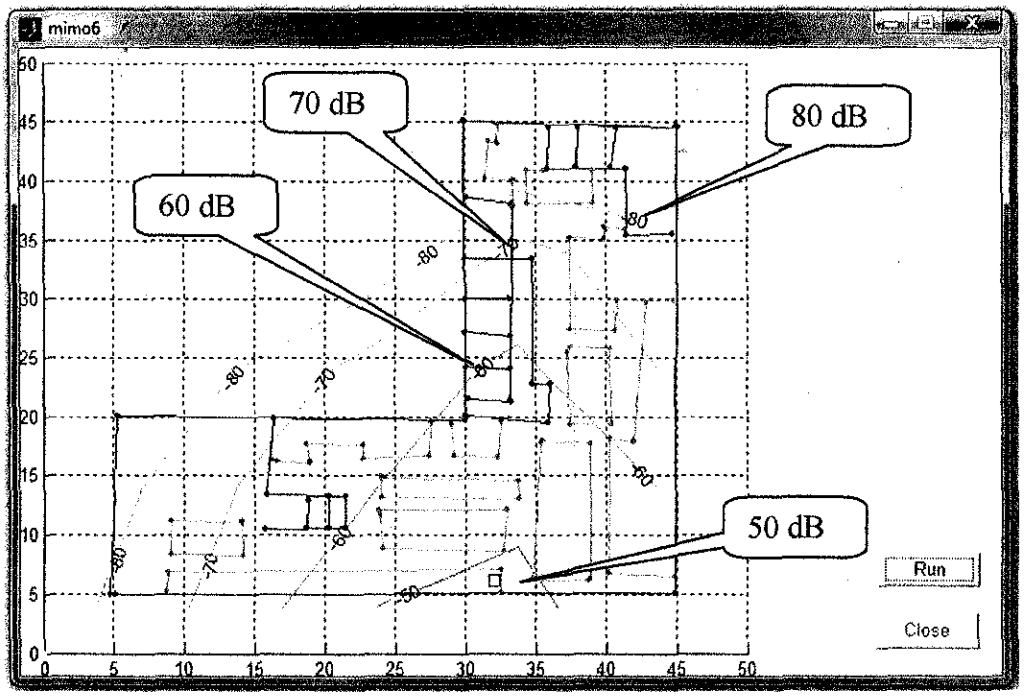
3.3.6 กำหนดค่าพารามิเตอร์



รูปที่ 19 กำหนดค่าพารามิเตอร์

- กำหนดค่าพารามิเตอร์ต่างๆ ในโปรแกรมเพื่อใช้ประมวลผล
- เลือก Next เพื่อเข้าสู่ขั้นตอนถัดไป
- เลือก Cancel เพื่อยกเลิกการใช้งานโปรแกรม

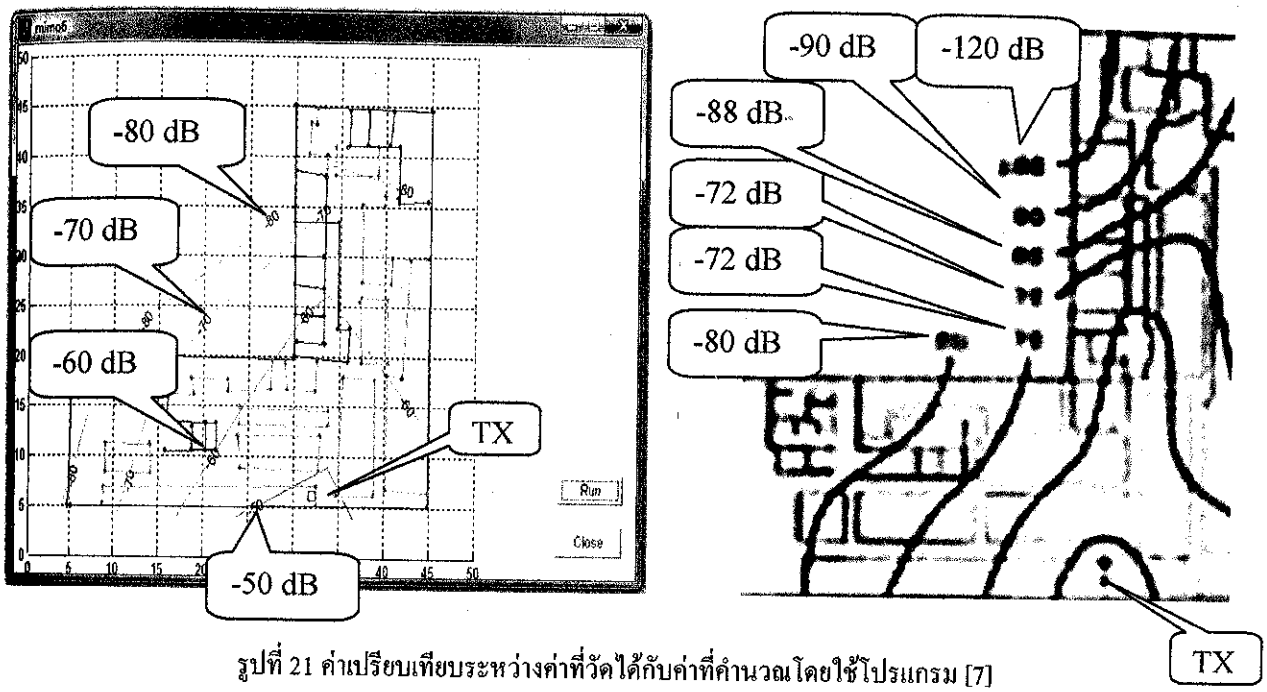
3.3.7 Plot ค่า ความแรงของสัญญาณ



รูปที่ 20 Plot ค่า ความแรงของสัญญาณ

- เลือก Run เพื่อสั่งทำงานโปรแกรมขั้นสุดท้าย
- เลือก Close เพื่อปิดการทำงานของโปรแกรม

บทที่ 4
วิเคราะห์ผลการทดลอง

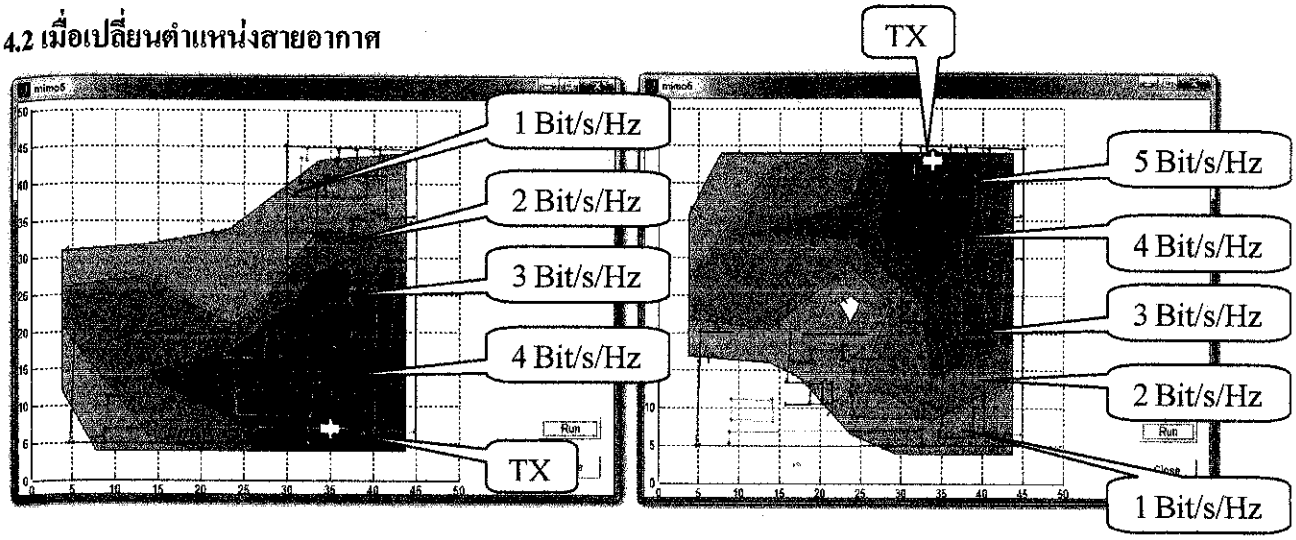


รูปที่ 21 ค่าเปรียบเทียบระหว่างค่าที่วัดได้กับค่าที่คำนวณโดยใช้โปรแกรม [7]

4.1 ผลการเปรียบเทียบผลการทดลอง

เมื่อเปรียบเทียบการแผ่กระจายของสัญญาณที่ได้จากการคำนวณและสร้างจากโปรแกรมที่สร้างขึ้นมา จะเห็นว่ามีลักษณะที่คล้ายกัน แสดงว่าการคำนวณโดยใช้โปรแกรมนั้นมีความถูกต้องใกล้เคียงกับผลที่ได้จากการวัดจริง แต่มีบางส่วนที่มีลักษณะที่ไม่เหมือนกัน เป็นเพราะว่า จากรูปที่ได้จากวัดจริง นั้นจะหาค่าการสูญเสียทุกๆ 5 เมตร ส่วนในโปรแกรมจะหาค่าทุกๆ 90 เมตร จึงทำให้ความละเอียดต่างกัน เป็นผลให้ลักษณะของ การแผ่กระจายของสัญญาณจากการวัดจริง มีลักษณะละเอียดกว่าและไม่เหมือนกับการแผ่กระจายของสัญญาณที่ได้จากโปรแกรม ระยะเวลาที่ใช้ในหาการรันโปรแกรมประมาณ 1 ชั่วโมง 20 นาที ส่วนสาเหตุที่ไม่สามารถทำการประมวลผลทุกๆ 5 เมตรได้ เนื่องจากจะต้องใช้ระยะเวลานาน 540 ชั่วโมง หรือประมาณ 22.5 วัน

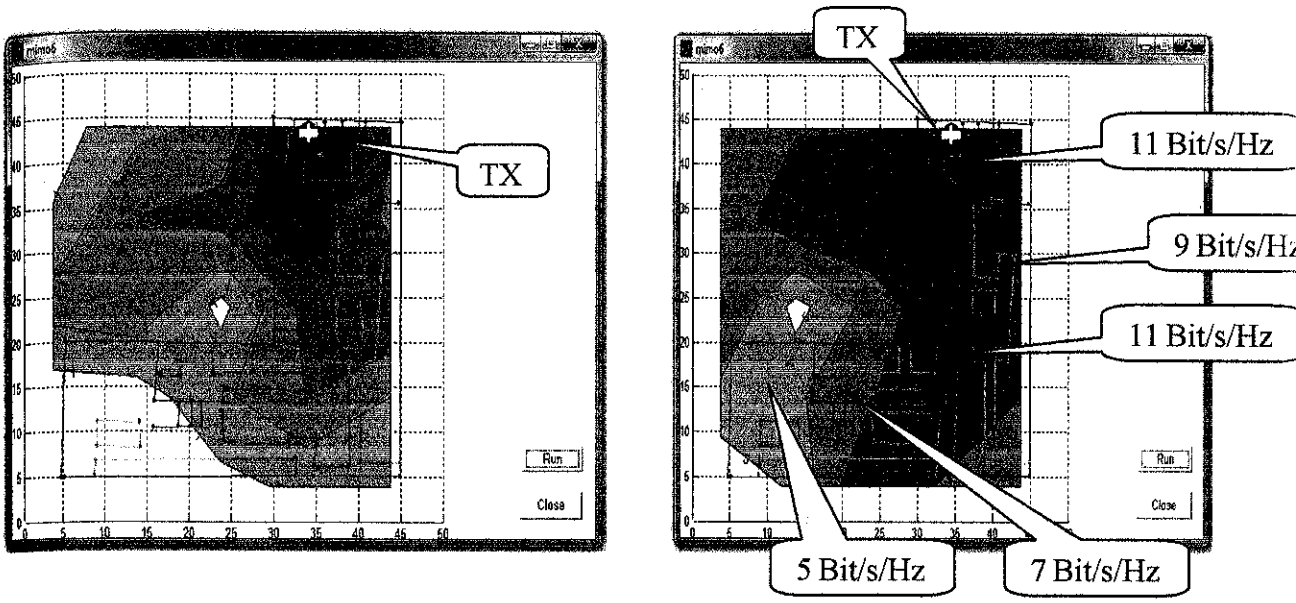
4.2 เมื่อเปลี่ยนตำแหน่งสายอากาศ



รูปที่ 22 ผลที่ได้เมื่อเปลี่ยนตำแหน่งสายอากาศ

เปลี่ยนตำแหน่งของตัวส่งสัญญาณ จะพบว่าลักษณะของสัญญาณที่แผ่ออกมาจะมีความจุของช่องสัญญาณมากที่สุดที่บริเวณใกล้ๆ กับตัวส่ง และจะน้อยลงเรื่อยตามระยะทางเมื่อสิ่งกีดขวางที่เพิ่มมากขึ้นและสามารถหาตำแหน่งภายในอาคารที่ดีที่สุดเพื่อครอบคลุมพื้นที่ทั้งอาคาร คล้ายกับตำแหน่งแรกที่ทำการทดสอบ

4.3 เมื่อเพิ่มจำนวนสายอากาศ



รูปที่ 23 เมื่อเพิ่มจำนวนสายอากาศ

เมื่อเพิ่มจำนวนสายอากาศจาก 2 ตัวเป็น 4 ตัว จะทำให้การแผ่กระจายของสัญญาณมีพื้นที่ครอบคลุมและมีความจุของช่องสัญญาณมากขึ้น เนื่องจากมีการส่งสัญญาณออกมาหลายทิศทางและมีการถ่ายโอนข้อมูลได้เร็วจากสายอากาศที่มีหลายต้น ทำให้ตัวรับสัญญาณสามารถรับสัญญาณได้มากขึ้นตามไปด้วย

บทที่ 5

สรุปผลการทดลอง

ข้อดี

1. โปรแกรมสามารถใช้ในการออกแบบระบบ MIMO ที่ใช้ภายในอาคาร
2. โปรแกรมใช้ศึกษาแบบจำลองของระบบ MIMO ภายในอาคาร สามารถทดสอบและใช้งานได้จริง
3. การเพิ่มจำนวนของสายอากาศทำให้สามารถครอบคลุมพื้นที่การส่งสัญญาณและความจุของช่องสัญญาณเพิ่มมากขึ้น
4. เรานำเทคโนโลยี MIMO มาใช้ในการรับส่งสัญญาณ จะช่วยให้ระบบการส่งสัญญาณไร้สายมีประสิทธิภาพเพิ่มมากขึ้น

ข้อเสีย

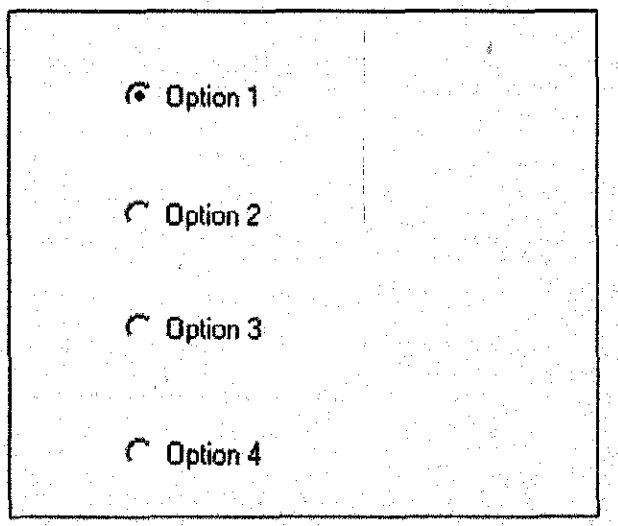
1. โปรแกรมจำลองระบบนี้สามารถใช้ได้เพียงภายในอาคารเท่านั้น
2. โปรแกรมจำลองระบบออกแบบให้ใช้งานในรูปแบบ 2 มิติเท่านั้นจึงไม่สามารถทราบได้ว่าผลของความสูงของสายอากาศหรือวัสดุมีผลอย่างไรกับความจุของสัญญาณ

ภาคผนวก ก

วิธีใช้งาน GUIDE

1. ตัวอย่างการใช้ Radio Button

ต่อไปเราจะลองสร้าง GUI แบบง่ายๆขึ้นโดยใช้ GUIDE เป็นเครื่องมือช่วย ซึ่งตัวอย่างหลายๆ ตัวอย่างต่อไปนี้จะช่วยให้เราเข้าใจการทำงานและวิธีการแก้ไข application M-file ได้ดียิ่งขึ้นในตัวอย่างนี้จะเป็นการสร้าง GUI ที่ประกอบด้วย Radio Button หลายๆ ตัวเพื่อให้ผู้ใช้ได้ทำการเลือก จากนั้นเราจะแสดงผลการเลือกให้ผู้ใช้ได้ทราบ ลักษณะของ GUI จะมีลักษณะโดยโครงร่างดังนี้



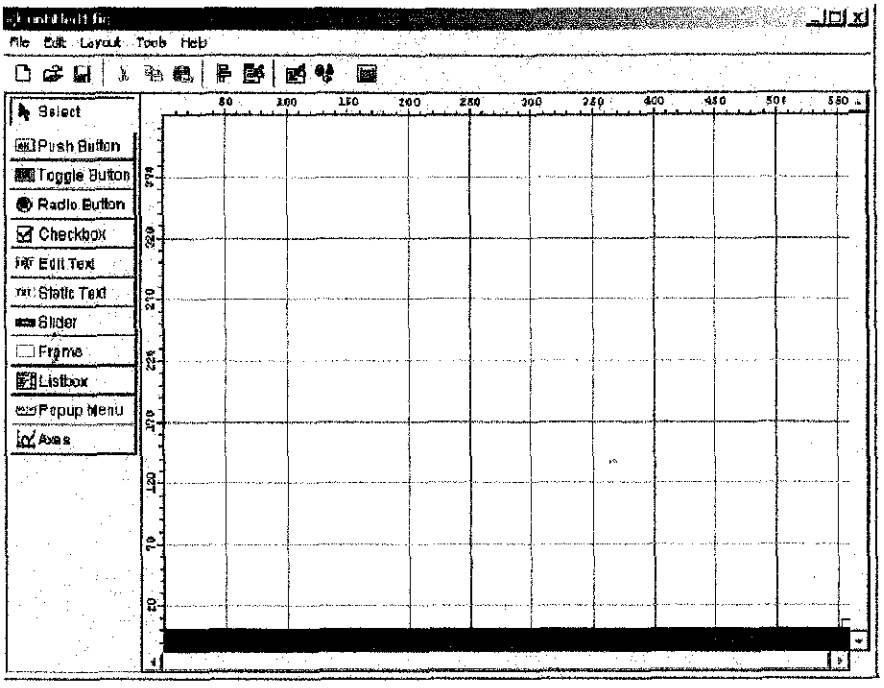
นั่นคือเราจะมีตัวเลือก 4 ตัวเลือก ให้ผู้ใช้เลือก และเมื่อผู้ใช้เลือกแล้ว จะมีข้อความบอกว่าผู้ใช้เลือกตัวเลือกใด และตัวเลือกเหล่านี้จะเลือกได้เพียงตัวเลือกเดียว เพราะปกติการเลือกที่เลือกได้ตัวเลือกเดียวเราจะนิยมใช้ radio button แต่ถ้าต้องการให้สามารถเลือกได้หลายตัวเลือก เราจะนิยมใช้ check box ซึ่งเราก็คุ้นเคยกับตัวเลือกประเภทนี้อยู่แล้ว ขั้นตอนการสร้าง GUI มีดังต่อไปนี้

วางรูปแบบใน GUIDE Layout Editor

ขั้นตอนแรกนี้เราจะเรียก GUIDE ขึ้นมาใช้งาน ใน Command Windows โดยใช้คำสั่ง

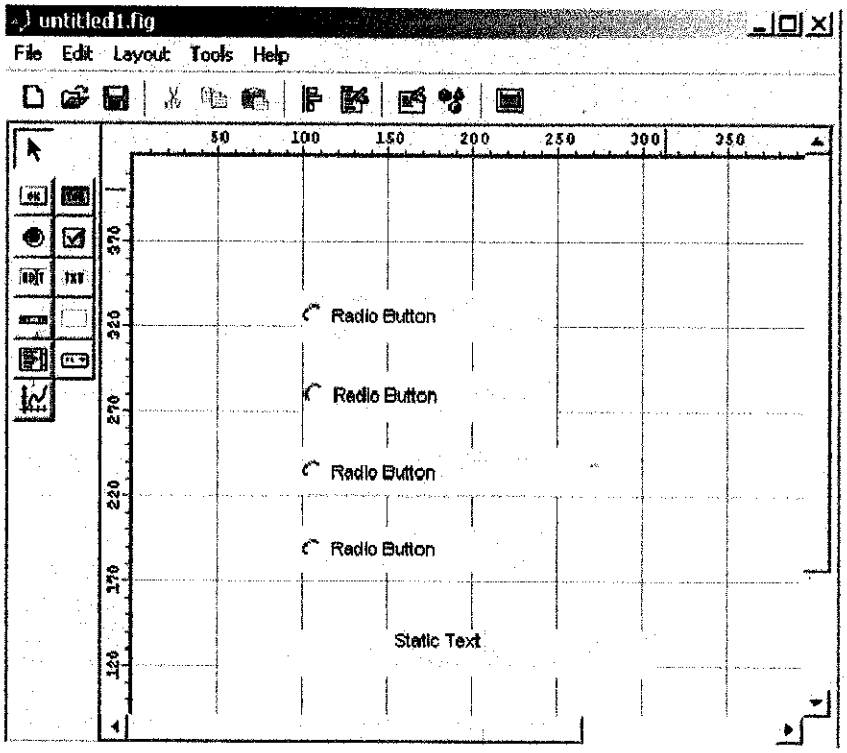
»GUIDE

ซึ่งเราก็จะได้หน้าต่างของ GUIDE Layout Editor ปรากฏขึ้นลักษณะดังรูป



ขั้นแรกเราควรจะกำหนดตัวเลือกต่างๆ ขึ้นมาก่อนโดยเลือก Allocation Option จากเมนู Tool แล้วเลือกตัวเลือกต่างๆ ตามที่เราต้องการ สำหรับในกรณีนี้เราจะไม่ปรับค่าใดๆ ที่ MATLAB ตั้งเป็นค่าเริ่มต้นมาให้

จากนั้นวางวัตถุ uicontrol ตามที่กำหนดลงไปใน GUI โดยเริ่มจากการเลือก Radio Button มาวางลงไป 4 อัน เลือก Radio Button ใน Tool Pallet ทางด้านซ้ายมือจากนั้นนำเมาส์เข้ามาในเนื้อที่หน้าต่างด้านขวามือ กดเมาส์ปุ่มซ้ายเพื่อกำหนดขอบบนซ้ายของวัตถุ กดเมาส์ค้างแล้วเลื่อนเมาส์ไปเพื่อเลือกขนาดของวัตถุ และปล่อยที่ตำแหน่งที่ต้องการให้เป็นมุมขวาล่างของวัตถุ สร้าง Radio Button ทีละอันจนครบทั้ง 4 อัน และจัดลำดับการวางโดยอันแรกให้อยู่ด้านบนสุด และอันต่อมาก็วางใต้อันแรกลดลั่นกันลงมาเพื่อความสะดวกในการกำหนดคุณสมบัติต่อไปสำหรับขนาด และการวางนั้นเรายังไม่ต้องสนใจมากในขณะนี้ก็ได้ เพราะเราสามารถปรับตำแหน่งและขนาดได้ภายหลัง และทำนองเดียวกันกับ Static Textซึ่งจะทำให้เราได้รูปแบบคร่าวๆของ GUI ในขณะนี้เป็น

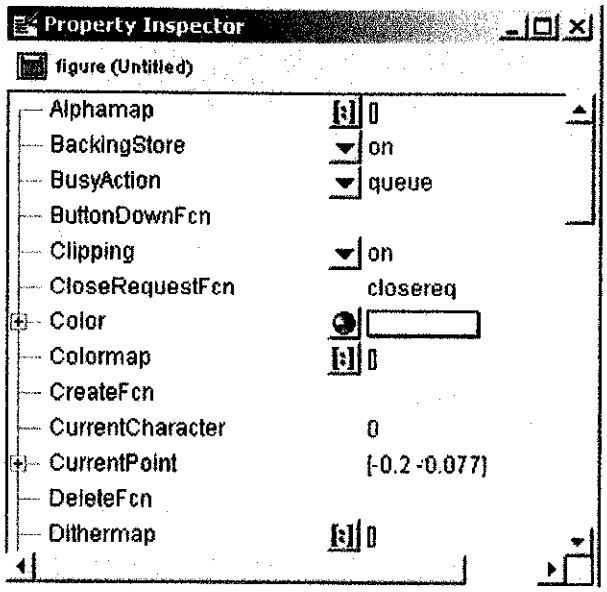


ขั้นต่อไปเพื่อความสวยงามเราอาจจะเพิ่ม Frame เพื่อจัดหมวดหมู่ใน GUI ของเรา เลือก Frame จาก Tool Pallet แล้วนำมาวางลงบน Radio Button ทั้งสี่ แต่เนื่องจากเราวาด Frame ขึ้นที่หลัง Radio Buttonจึงทำให้ Frame นั้นวางอยู่ข้างบนซึ่งทำให้มองไม่เห็น Radio Button ทั้งสี่

เราสามารถแก้ไขได้โดยใช้เมาส์ปุ่มขวากดลงไปที่ Frame จากนั้นจะปรากฏ Context Menu ขึ้น เราเลือก Send to Back เพื่อส่ง Frame นี้ไปอยู่ด้านหลังของ Radio Button ซึ่งเราก็จะได้รูปตามต้องการสุดท้ายของการจัดวัตถุบน GUI เราอาจจะใช้ Alignment Tool เพื่อช่วยในการจัดเรียงวัตถุเหล่านั้นให้มีระเบียบยิ่งขึ้นก็ได้

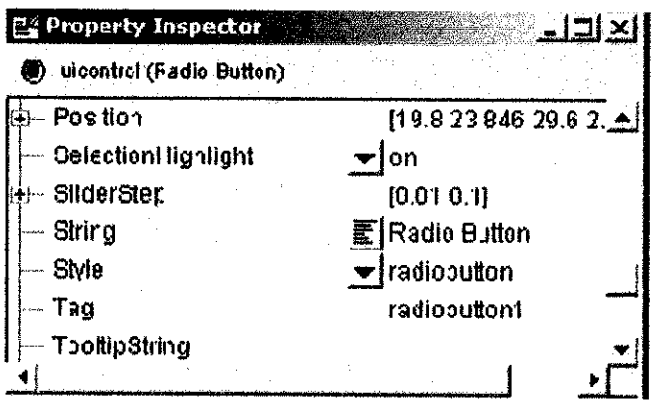
กำหนดชื่อวัตถุโดย Tag และข้อความที่จะปรากฏขึ้น

ต่อไปจะเป็นการกำหนดคุณสมบัติของวัตถุต่างๆ ขั้นแรกให้เราเปิด Property Inspector ขึ้นมาโดยการกดเมาส์ที่ปุ่ม  ซึ่งเป็นการเปิดหน้าต่าง Property Inspector ขึ้นมาซึ่งมีลักษณะดังรูป



ซึ่งเราจะเห็นจากรูปว่าในขณะนี้เป็นการแสดงคุณสมบัติของ Figure อยู่ และเมื่อเราเปลี่ยนตัวเลือก คือใช้เมาส์เลือกวัตถุใน Layout Editor หน้าต่าง Property Inspector ก็จะแสดงคุณสมบัติของวัตถุที่เราเลือก

อันดับแรกให้เราเลือกที่ Radio Button อันบนสุด ซึ่งเป็นอันแรกที่เรานำมาวางไว้ในรูป จากนั้นพิจารณาคุณสมบัติใน Property Inspector เราจะพบว่ามีคุณสมบัติ Tag และ String ดังรูป



ซึ่งคุณสมบัติ Tag จะมีค่าเป็น radiobutton1 และมีค่า String เป็น Radio Button ซึ่งเป็นค่า default ของวัตถุนี้ เราจะสามารถเปลี่ยนคุณสมบัติดังกล่าวได้ โดยการใช้เมาส์ไปกดลงที่ค่าคุณสมบัติเหล่านั้น ในกรณีนี้ให้เราเปลี่ยนคุณสมบัติ Tag ให้เป็น Option1 และเปลี่ยน String เป็น Option 1 ซึ่งเราจะสังเกตเห็นว่าเมื่อเราเปลี่ยนค่า String ไปแล้ว ข้อความที่ปรากฏบน Radio Button นี้จะเปลี่ยนไปเป็น Option 1

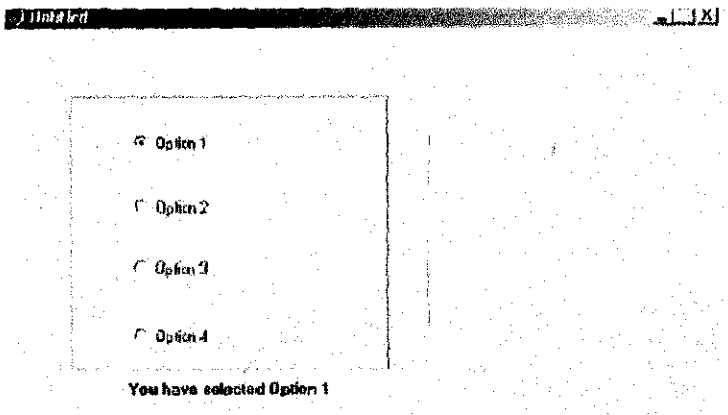
จากนั้นให้เราเปลี่ยนค่าคุณสมบัติ Tag และ String ของ Radio Button ที่เหลือให้เป็น Option2, Option3, Option4 และ Option 2, Option 3, Option 4 ตามลำดับ

ขั้นต่อไปให้เราเปลี่ยนคุณสมบัติ Value ของ Radio Button Option 1 จากค่าเดิมที่เป็น 0.0 ให้เป็น1.0 ในส่วนนี้ก็เพื่อกำหนดว่าตัวเลือกนี้เป็นตัวเลือก default นั่นเอง

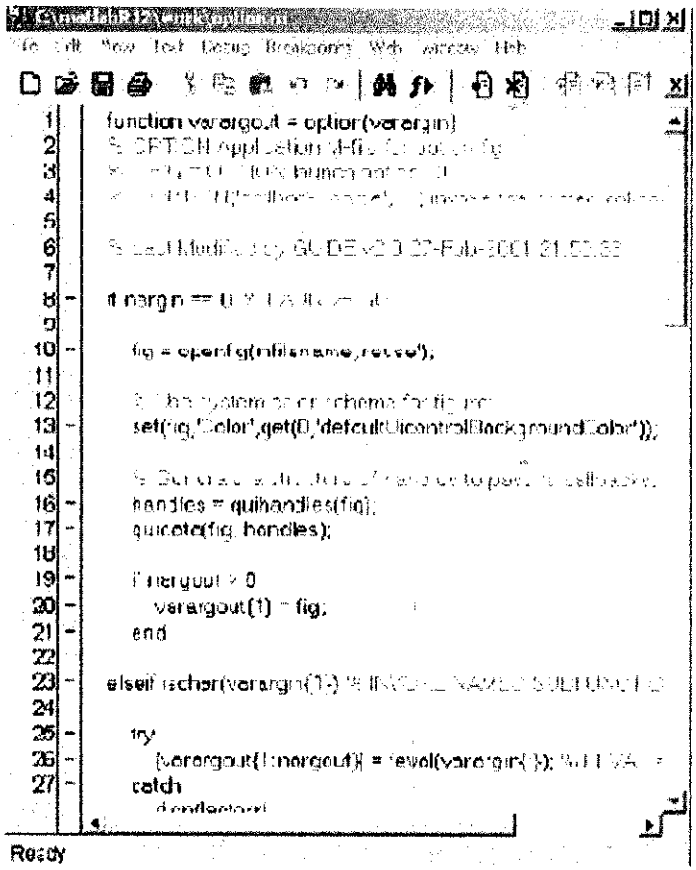
จากนั้นให้เลือก Static Text แล้วดูคุณสมบัติ String ซึ่งจะมีค่า Default เป็น Static Text ให้เราเปลี่ยนค่านี้เป็น You have select Option 1 เพื่อให้เป็นไปตาม default ที่เราจะกำหนดให้ผู้ใช้

Save และ Activate GUI

เมื่อเราปรับปรุงคุณสมบัติของวัตถุต่างๆ ตามที่เราต้องการเรียบร้อยแล้วให้ทำการ Save GUI โดยเลือก Save จากเมนู File หรือใช้เมาส์กดที่ปุ่ม  ก็ได้ จากนั้น MATLAB จะถามชื่อไฟล์ที่ต้องการเก็บซึ่งในที่นี้จะตั้งชื่อว่า option แล้ว Click OK จากนั้น MATLAB ก็จะทำการเก็บไฟล์เป็นสองไฟล์คือไฟล์ข้อมูลที่เกี่ยวกับการวางรูปแบบ วัตถุและคุณสมบัติของวัตถุต่างๆ จะเก็บไว้ในไฟล์ที่ชื่อ option.figและ Application M-file ที่ชื่อ option.m
จากนั้น MATLAB ก็จะเปิดหน้าต่างของ GUI ของเราขึ้นมา ดังรูป



และ Editor/Debugger ก็จะถูกเปิดขึ้นมาพร้อมกับเปิดไฟล์ option.m ดังแสดงในรูป



ตอนนี้ถ้าหากว่าเราลองเลือกปุ่ม Option อื่นๆ ใน GUI จะมีข้อความปรากฏที่ Command Windows ดังนี้

Option3 Callback not implemented yet.

สาเหตุก็เพราะเรายังไม่ได้ทำการปรับปรุง callback ซึ่งจะเป็น subfunction ที่อยู่ในไฟล์ option.m

ปรับปรุง และเพิ่มเติมชุดคำสั่งให้ GUI ทำงานตามที่ต้องการ

ในการปรับปรุงการทำงานของโปรแกรมนี้จะมีขั้นตอนหลายขั้นตอน แต่ในขั้นตอนแรกเราจะมองในภาพรวมของ Radio Button ทั้งสี่ก่อน เมื่อเราต้องการให้ผู้ใช้เลือกสามารถเลือกตัวเลือกได้เพียงครั้งละหนึ่งตัวเลือกเท่านั้น ดังนั้นหากผู้ใช้เลือก Option 2 ตัวเลือก Option 1 ที่เลือกไว้เดิมจะต้องถูกปิดโดยอัตโนมัติ และก็จะเป็นทำนองเดียวกันกับตัวเลือกอื่นๆ ดังนั้นเราควรจะเขียนฟังก์ชันเป็น subfunction เพิ่มเติมขึ้นมาสำหรับกรณีเช่นนี้ โดยเขียนไว้ที่ส่วนท้ายของโปรแกรม มีลักษณะดังนี้

```
% --------------------------------------------------------------------
function Turn_Off(off)
set(off,'Value',0);
% --------------------------------------------------------------------
```

Subfunction นี้ชื่อ Turn_Off จะรับข้อกำหนดเป็นค่า off จากการเรียกใช้คำสั่ง ซึ่งในที่นี้ off จะเป็นvector ของ handle ของตัวเลือกต่างๆ ที่เราต้องการปิด จากนั้นทำการตั้งค่าคุณสมบัติ value ของ handleต่างๆ ที่ส่งมากับข้อกำหนด off นี้ให้มีค่าเป็นศูนย์ ซึ่งก็หมายถึง radio button ตัวนั้นไม่ได้เลือกหรือถูกปิดนั่นเอง ในส่วนการเขียน subfunction เพื่อปิดตัวเลือกต่างๆ ก็จะเสร็จสิ้นลงเท่านี้

ขั้นต่อไปเราจะทำการเขียน subfunction ให้กับ callback ของ uicontrol ต่างๆ อันดับแรกให้ไปที่subfunction ของตัวเลือก Option 1 นั่นคือ

```
function varargout=Option1_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.Option1.
disp('Option1 Callback not implemented yet.')
```

ขั้นแรกให้ทำการลบบรรทัด disp('...') นี้ออกไปก่อน เราไม่จำเป็นต้องใช้อีกแล้ว จากนั้นพิมพ์ชุดคำสั่งต่อไปนี้ลงไปแทน

```
off = [handles.Option2 handles.Option3 handles.Option4];
turn_Off(off);
set(h,'Value',1);
set(handles.text1,'String','You have selected Option 1');
```

คำสั่งกำหนดค่า off ในบรรทัดแรก จะเป็นการสร้าง vector บรรจุค่าของ handle ของ Radio Button อีกสามอันที่เหลือที่ผู้ใช้ไม่ได้เลือก ในบรรทัดที่สองเราจะส่งค่า off นี้ไปให้ function Turn_Off ทำการเปลี่ยนค่า Value ของปุ่มทั้งสามนี้ให้เป็นศูนย์ สำหรับบรรทัดที่ 3 เรากำหนดให้ค่า Value ของวัตถุนี้ (hเป็นค่า handle ของวัตถุที่ถูกเรียก Callback) มีค่าเป็น 1 เพื่อป้องกันผู้ใช้กดปุ่มตัวเลือกที่เลือกอยู่แล้วซ้ำค่าจะได้ไม่กลับเป็นศูนย์ (การทำงานโดยปกติของ Radio Button นี้ถ้าผู้ใช้กดปุ่มเลือกขณะที่มันถูกเลือก(value = 1) อยู่เดิมแล้ว ค่าจะกลับเป็นไม่มีการเลือก(value = 0) และถ้ามีค่าเป็นไม่มีการเลือก (value = 0)อยู่ก่อนเมื่อผู้ใช้กดปุ่มเลือกก็จะเปลี่ยนเป็นมีการเลือก (value = 1)

ส่วนในบรรทัดสุดท้ายจะเป็นการกำหนดค่า String ของ Static Text ซึ่งมีค่า handles เป็น handles.text1 ให้มีค่าเป็น You have selected Option 1 ซึ่งเป็นการแสดงผลให้ผู้ใช้ทราบว่าเลือกตัวเลือกที่ 1 ในทำนองเดียวกันให้เราเปลี่ยน subfunction ที่เหลือให้มีลักษณะดังนี้

```
% --------------------------------------------------------------------
function varargout=Option2_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.Option2.
off =[handles.Option1 handles.Option3 handles.Option4];
Turn_Off(off);
set(h,'Value',1);
set(handles.text1,'String','You have selected Option 2');
% --------------------------------------------------------------------
```

```
function varargout=Option3_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.Option3.
off =[handles.Option1 handles.Option2 handles.Option4];
Turn_Off(off);
set(h,'Value',1);
set(handles.text1,'String','You have selected Option 3');
% --------------------------------------------------------------------
function varargout=Option4_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.Option4.
off =[handles.Option1 handles.Option2 handles.Option3];
Turn_Off(off);
set(h,'Value',1);
set(handles.text1,'String','You have selected Option 4');
```

ซึ่งเราจะเห็นว่ามีข้อแตกต่างในรายละเอียดของ Subfunction ต่างๆก็เพียงค่าตัวแปร off ที่จะทำการเปลี่ยนตัวเลือกอื่นให้มี Value = 0 และข้อความว่าขณะนี้ผู้ใช้ได้เลือกตัวเลือกใด เมื่อได้แก้ไขและเขียนคำสั่งให้กับ GUI เรียบร้อยแล้ว ให้ Save ไฟล์นี้ก่อนที่จะใช้งานมันต่อไป

ทดสอบการทำงานของ GUI

หลังจากที่เราเขียนคำสั่งขึ้นมาเรียบร้อยแล้ว ขั้นต่อไปจะเป็นการทดสอบ GUI ของเรา ถ้าหน้าต่าง GUI ปิดอยู่เราสามารถจะเปิดหน้าต่างขึ้นมาใหม่ได้ โดยที่ไปที่ Command Windows แล้วสั่ง

»option

จากนั้นลองทดสอบการทำงานของ GUI ของเราดู ซึ่งถ้าหากทำตามขั้นตอนที่กล่าวมาแล้วอย่างครบถ้วน ก็ไม่น่าจะมีปัญหาอะไร

ปรับปรุงข้อผิดพลาด

ในส่วนนี้เป็นขั้นตอนปกติของการเขียนโปรแกรม ซึ่งอาจมีข้อผิดพลาด หรือเราอาจจะยังไม่ใส่ใจในรายละเอียดในตอนแรก เมื่อได้ทดลองใช้แล้วอาจจะต้องมีการปรับปรุงการทำงานให้ดียิ่งขึ้น หรือรูปแบบของ GUI ให้มีความสวยงามและเหมาะสมยิ่งขึ้น

ถ้าเราดูที่หน้าต่าง GUI ของเราในตอนนี้เราจะพบว่าที่ title นั้นจะเป็นชื่อ Untitled อยู่เพราะยังไม่ได้กำหนดชื่อของ GUI ที่จะให้ปรากฏในระหว่างใช้งาน ซึ่งเราก็สามารถที่จะแก้ไขได้ โดยไปที่

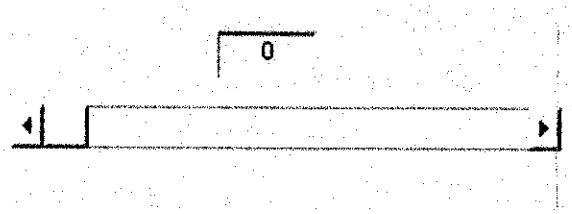
Layout Editor เลือก figure คือกดเมาส์ลงในที่ว่างตรงใดก็ได้ จากนั้นไปที่ Property Inspector แล้วเปลี่ยนค่าคุณสมบัติ Name จาก Untitled ให้เป็น My First GUI: Option (หรือจะเป็นชื่ออื่นก็ได้ ตามที่เราต้องการ)จากนั้นสั่ง Save ที่ Layout Editor อย่างไรก็ดีถ้าเรามีหน้าต่างเก่าเราอยู่ เราจะเห็นว่าชื่อนั้นยังไม่เปลี่ยนแปลง สาเหตุเพราะเรายังไม่ได้มีการ update ข้อมูลส่งไปให้ GUI

เพื่อเป็นการ update ข้อมูลให้กับ GUI ของเราให้เราสั่ง Activate GUI ของเราอีกครั้งหนึ่งก่อน ซึ่งก็จะทำให้เราเห็นการเปลี่ยนแปลงที่เกิดขึ้นกับ GUI ของเรา

ตัวอย่างที่ผ่านมานี้ แสดงขั้นตอนการเขียน GUI ให้เราได้ทำความเข้าใจ แม้ว่า GUI ที่ยกตัวอย่างนี้จะเป็นตัวอย่างง่ายๆ แต่ขั้นตอนการเขียน GUI นี้สามารถนำไปใช้กับการเขียน GUI ได้แทบทุกระดับความยาก เพราะความยุ่งยากจริงๆ อยู่ที่การเขียน Subfunction ซึ่งเปรียบเสมือนเป็น Callback Functionนั่นเอง ความยากง่ายก็มักจะขึ้นอยู่กับขั้นตอนการคำนวณ การกำหนดคุณสมบัติ การปรับปรุงค่าคุณสมบัติของวัตถุ และการส่งผ่านข้อมูลเหมือนกับการเขียน Function-file ทั่วๆ ไปนั่นเอง

2. ตัวอย่าง Slider และ Editable Text

ตัวอย่างนี้เราจะลองใช้ uicontrol ประเภทอื่นดูบ้าง ซึ่งเราจะสร้าง GUI ที่มีลักษณะดังรูป



คุณสมบัติคร่าวๆ ของ Slider ก็คือจะเป็นมีค่าคุณสมบัติ Value สัมพันธ์กับระยะเลื่อนของตัวเลื่อนบน Slider โดยถ้าตัวเลื่อนอยู่ที่ตำแหน่ง ซ้ายสุดก็จะให้ค่าต่ำสุดและเมื่อตัวเลื่อนอยู่ที่ตำแหน่งขวาสุดก็จะให้ค่าสูงสุด และค่าระหว่างนั้นจะเป็นการเปลี่ยนแปลงเชิงเส้นจากค่าน้อยสุดถึงมาที่สุดสัมพันธ์กับระยะเลื่อนของตัวเลื่อน ค่าต่ำสุดและสูงสุดจะถูกกำหนดโดยคุณสมบัติ Max และ Minตามลำดับ ส่วนค่าที่ได้ในขณะนั้นจะแสดงทางคุณสมบัติ Value

สำหรับ Editable Text นั้นเป็นการแสดงผลที่เป็นตัวหนังสือที่ผู้ใช้สามารถกำหนดลงไปได้ ค่าที่แสดงผลนั้นเป็นไปตามคุณสมบัติ String ของ Editable Text ดังนั้นไม่ว่าตัวอักษรหรือตัวเลขที่แสดงอยู่นั้นจะเป็นตัวแปรประเภท String ไม่ใช่ตัวแปร Numeric

ในตัวอย่างนี้เราจะมีข้อกำหนดในการทำงาน GUI ดังนี้คือ

- ถ้าเราปรับค่าที่ Slider จะทำให้ค่าตัวเลขที่แสดงใน Editable Text นี้เปลี่ยนไปตามค่าของ Slider ที่เปลี่ยนไป
- ถ้าเราปรับค่าตัวเลขที่ Editable Text จะทำให้ค่าที่แสดงใน Slider นี้เปลี่ยนไปตามค่าของที่ผู้ใช้ใส่ไป
- ค่าสูงสุดและต่ำสุดของ Slider นี้จะเป็น 10 และ 0 ตามลำดับ ถ้าหากว่าผู้ใช้กำหนดค่าทาง Editable Text ให้มีค่ามากกว่าหรือน้อยกว่าค่านี้ ค่าจะปรับเป็นค่าสูงสุดหรือต่ำสุดโดยอัตโนมัติ

วางรูปแบบใน GUIDE Layout Editor

ขั้นตอนแรกนี้เราจะเรียก GUIDE ขึ้นมาใช้งาน ดังนั้นใน Command Windows ให้เราสั่ง

»GUIDE

ซึ่งเราก็จะได้หน้าต่างของ Layout Editor ปรากฏขึ้น จากนั้นวาง uicontrol ทั้งสองตามรูปที่แสดงข้างบนคือวาง Editable Text ไว้เหนือ Slider ขั้นตอนการวางจะเหมือนที่กล่าวมาแล้วในตัวอย่างที่ผ่านมา

กำหนดชื่อวัตถุโดย Tag และข้อความที่จะปรากฏขึ้น

ขั้นต่อไปจะเป็นการกำหนดคุณสมบัติของวัตถุต่างๆ ขั้นแรกให้เราเปิด Property Inspector ขึ้นมาโดยการกดเมาส์ที่ปุ่ม  ซึ่งเป็นการเปิดหน้าต่าง Property Inspector ขึ้นมา จากนั้นให้ดูคุณสมบัติ Tagของวัตถุทั้งสอง โดย Editable Text จะมี Tag เป็น edit1 และ Slider จะมี Tag เป็น slider1 เราจะใช้ค่านี้ในการเขียนคำสั่งโดยไม่เปลี่ยนแปลง

คุณสมบัติที่เราจะต้องทำการเปลี่ยนแปลงในที่นี้อันแรกคือค่าต่ำสุดของ.Slider เพราะโดย Default แล้วค่าต่ำสุดจะเป็น 0 ส่วนค่าสูงสุดจะเป็น 1 ดังนั้นในที่นี้เราจะเปลี่ยนค่าคุณสมบัติ Max ให้เป็น 10 และให้แน่ใจว่าค่า Value ของ Slider นั้นมีค่าเท่ากับ 0 ซึ่งเป็น default

คุณสมบัติของ Editable Text ที่เราจะเปลี่ยนก็คือค่า String ซึ่งเราจะเปลี่ยนจากค่าเดิมให้เป็นตัวเลข 0 ซึ่งต้องทำความเข้าใจก่อนว่าค่าคุณสมบัติตัวนี้จะเป็น string ดังนั้นจะไม่มีค่าในเชิงคณิตศาสตร์นั่นคือเราจะนำไปกำหนดค่าตัวอื่นไม่ได้ ตัวเลข 0 ในที่นี้ไม่ได้มีผลทางคำนวณแตกต่างจากตัวอักษร a เลย

Save และ Activate GUI

ขั้นต่อไปให้เรา Save GUI ของเราโดยเลือก Save จากเมนู File หรือใช้เมาส์กดที่ปุ่ม ก็ได้ จากนั้น MATLAB จะถามชื่อไฟล์ที่เราต้องการเก็บ ซึ่งในที่นี้เราจะตั้งชื่อว่า numerical ซึ่งเมื่อเราเลือก OKแล้ว MATLAB ก็จะทำการเก็บเป็นสองไฟล์คือ numerical.fig และ numerical.m

เมื่อทำการ Activate GUI จะทำให้ MATLAB เปิดหน้าต่างของ GUI ของเราขึ้นมาและEditor/Debugger ก็จะถูกเปิดขึ้นมาพร้อมกับเปิดไฟล์ numerical.m ขึ้น

ปรับปรุง และเพิ่มเติมชุดคำสั่งให้ GUI ทำงานตามที่ต้องการ

เนื่องจากค่าที่ได้จาก Slider มีค่าเป็นตัวเลข ส่วนค่าที่ได้จาก Editable Text มีค่าเป็น String ทำให้เราไม่สามารถที่จะใช้ค่าของ Editable Text กับ Slider ได้โดยตรง จะต้องทำการเปลี่ยนคุณสมบัติก่อน ซึ่งจะทำได้โดยอาศัยฟังก์ชัน 2 ฟังก์ชันคือ

>> **str2num(X)** ทำการเปลี่ยน String X ให้มีค่าเป็นตัวเลข ในกรณีที่ X ไม่ใช่ตัวอักษณที่เป็นตัวเลข เราจะได้ค่าว่าง [ ] ออกมา

>>**num2str(X)** ทำการเปลี่ยนตัวเลข X ให้มีค่าเป็น String

ดังนั้นสำหรับ Slider ซึ่งมี callback ชื่อ slider1_Callback จะมีลักษณะของคำสั่งเป็น

```
nvalue = get(h,'Value');
svalue = num2str(nvalue);
set(handles.edit1,'String',svalue);
```

โดยบรรทัดแรกจะเป็นการอ่านค่าคุณสมบัติ Value ซึ่งได้ค่ามาเป็นตัวเลขเก็บไว้ในตัวแปรชื่อ nvalue ในบรรทัดที่สองจะเปลี่ยนค่า nvalue ที่เป็นตัวเลขนี้ให้มีค่าเป็น String และในบรรทัดสุดท้ายให้ปรับค่าคุณสมบัติ String ของ Editable Text ซึ่งมี

Tag เป็น edit1 ให้มีค่าเท่ากับ svalue ในที่นี้เราไม่ต้องกังวลเรื่องค่าต่ำสุดหรือสูงสุดเพราะผู้ใช้ไม่สามารถกำหนดค่าทาง slider ให้สูงกว่าหรือต่ำกว่าค่าสูงสุดหรือต่ำสุดของ slider ได้อยู่แล้ว

สำหรับ Editable Text ซึ่งมี callback ชื่อ edit1_Callback จะมีลักษณะคำสั่งดังนี้

```
svalue = get(h,'String');
nvalue = str2num(svalue)
if nvalue > 10
        nvalue = 10;
        set(h,'String','10');
elseif nvalue < 0
        nvalue = 0
        set(h,'String','0');
elseif nvalue == [ ]
        nvalue = 0
        set(h,'String','0');
end
set(handles.slider1,'Value', nvalue);
```

ในบรรทัดแรกจะเป็นการอ่านค่าคุณสมบัติ String จาก Editable Text แล้วในบรรทัดที่สองก็จะเป็นการเปลี่ยนค่า String ที่กำหนดให้เป็นตัวเลขเก็บค่าไว้ในตัวแปร nvalue ส่วนในประโยค if ถัดมาเป็นการตรวจสอบว่า nvalue มากกว่าหรือน้อยกว่าขีดจำกัดหรือไม่ ถ้ามากกว่าก็จะปรับค่าให้เท่ากับค่าสูงสุดแล้วกำหนดคุณสมบัติ String ของ Editable Text นี้ให้เป็นค่าสูงสุดด้วย (ค่า '10' เป็นการกำหนด String) และในส่วนที่เหลือก็จะมีลักษณะในทำนองเดียวกันคือเป็นการปรับค่าให้อยู่ในช่วงที่กำหนดและในบรรทัดสุดท้ายก็จะเป็นการกำหนดค่า Value ให้กับ Slider ตามที่ผู้ใช้ป้อนผ่านทาง Editable Text

ทดสอบการทำงานของ GUI

ขั้นสุดท้ายเป็นการทดสอบการทำงานของ GUI ให้เราทำการ save file ที่ Editor/Debugger เสียก่อน จากนั้น Activate GUI ของเรา แล้วทดสอบการทำงาน ถ้าทำตามขั้นตอนที่กล่าวมาแล้ว GUI ไม่น่าจะมีปัญหาในการทำงาน

3 ตัวอย่าง Check Box, List Box และ Popup Menu

ตัวอย่างต่อไปเราจะใช้ Check Box, List Box และ Popup Menu ในการสร้าง GUI ที่ต้องการให้ผู้ใช้เลือกตัวเลือกผ่านอุปกรณ์เหล่านี้ อันดับแรกจะเป็นการอธิบายคุณสมบัติของ Check Box, List Box และPopup Menu ดังนี้

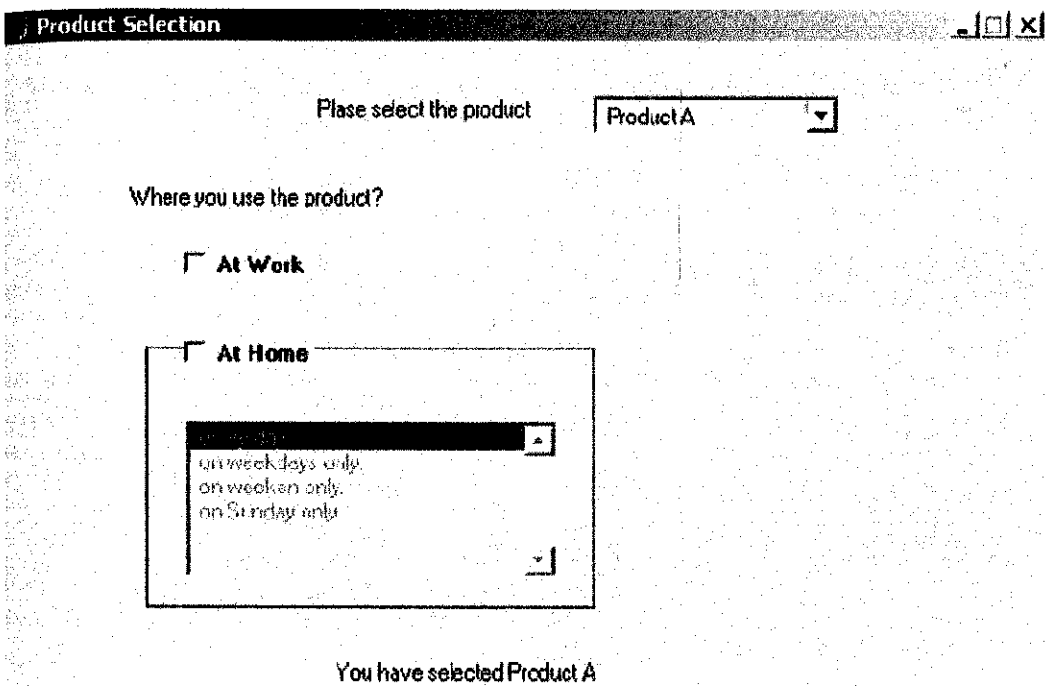
Check Box จะพิจารณาค่าคุณสมบัติ Value เพื่อแสดงว่าผู้ใช้เลือกหรือไม่ โดยเมื่อผู้ใช้เลือกจะมีค่าValue เป็น 1 และเมื่อไม่เลือกจะมีค่า Value เป็น 0

List Box จะพิจารณาค่าคุณสมบัติ Value ตามลำดับการเลือกตัวเลือกต่างๆ โดยถ้าผู้ใช้เลือกตัวเลือกที่ 1 ก็จะมีค่า Value เป็น 1 ถ้าผู้ใช้เลือกตัวเลือกที่ 2 ก็จะมีค่า Value เป็น 2 เป็นลำดับเช่นนี้เรื่อยไป การ

กำหนดตัวเลือกเหล่านั้นในคุณสมบัติ String ของ List Box นี้ จะทำได้โดยกำหนดค่าคุณสมบัติ String นี้เป็น Vector มีมิติเท่ากับจำนวนตัวเลือกที่ต้องการ

Popup Menu จะพิจารณาค่าคุณสมบัติ Value ตามลำดับการเลือกตัวเลือกต่างๆ เหมือนกับ List boxการกำหนดตัวเลือกเหล่านั้นให้กับคุณสมบัติ String ของ Popup Menu ก็จะกำหนดเป็น Vector มีมิติเท่ากับจำนวนตัวเลือกที่ต้องการ ด้วยเช่นกัน

ดังนั้นโดยการทำงานแล้ว List Box กับ Popup Menu นี้จะมีลักษณะเหมือนกัน เพียงแต่ว่าเรานิยมใช้ List Box เพื่อแสดงตัวเลือกทั้งหลายนี้พร้อมๆ กัน เพื่อให้ผู้ใช้ได้เห็นตลอดเวลา แต่มีข้อเสียที่เปลืองเนื้อที่บน GUI ซึ่ง Popup Menu จะใช้เนื้อที่บน GUI น้อยกว่า แต่ผู้ใช้จะเห็นตัวเลือกได้ค่าเดียว นอกจากจะกดเลือกเพื่อจะดูรายการที่มีอยู่ ลักษณะของ GUI ที่เราจะสร้างมีลักษณะดังรูปต่อไปนี้



ลักษณะการทำงานก็คือ

>> ผู้ใช้จะเลือกรายการ Product จาก Popup Menu

>> ผู้ใช้สามารถเลือกว่าใช้ผลิตภัณฑ์นั้นที่ใด คือที่ทำงาน และ/หรือ ที่บ้าน แต่ถ้าเลือกที่บ้านจะมีตัวเลือกให้เลือกว่าใช้ผลิตภัณฑ์นั้นบ่อยเพียงใด และถ้าหากไม่ได้เลือกตัวเลือกใช้ผลิตภัณฑ์นี้ที่บ้าน ให้ตัวเลือกใน List Box นั้นไม่สามารถเลือกได้

>> แสดงผลด้วย Static Text ข้างล่าง

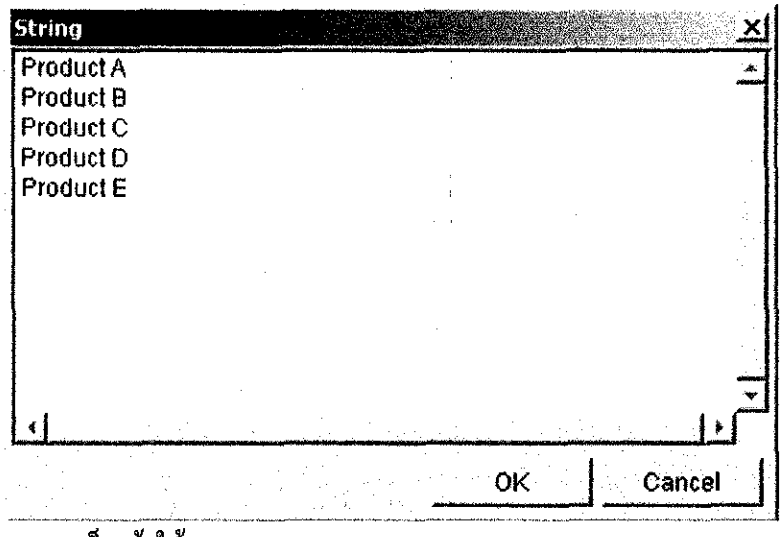
วางรูปแบบใน GUIDE Layout Editor

เริ่มด้วยการวัตถุตามรูปที่แสดงข้างบนนี้ ซึ่งเหมือนกับตัวอย่างที่ผ่านมาจึงไม่ขอกล่าวถึงในรายละเอียดในที่นี้อีก

กำหนดชื่อวัตถุโดย Tag และข้อความที่จะปรากฏขึ้น

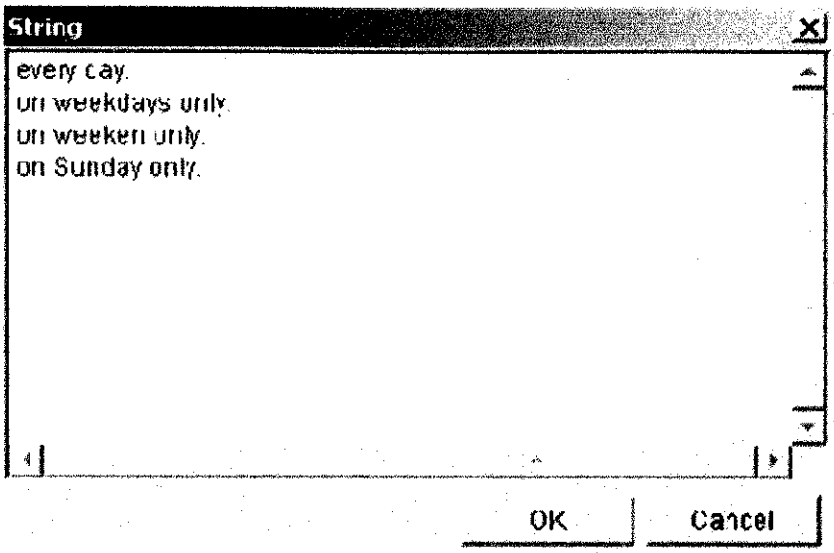
จากรูปข้างบนนี้ให้กำหนดคุณสมบัติของวัตถุต่างๆ ต่อไปนี้ สำหรับที่ไม่ได้กล่าวถึงหมายความว่าไม่จำเป็นต้องมีการปรับเปลี่ยนคุณสมบัติ

- String ของ Static Text ทั้งหมดดังรูป และให้แก้ไข Tag ของ Static Text ที่แสดงผล การเลือก Product ให้เป็น Result
- สำหรับ Check Box ที่มี String เป็น At Work ให้ใช้ Tag เป็น Work และให้ Value เป็น 0
- สำหรับ Check Box ที่มี String เป็น At Home ให้ใช้ Tag เป็น Home และให้ Value เป็น 0
- สำหรับ Popup Menu ให้กำหนด Tag เป็น Popup1 และให้กรอกคุณสมบัติ Sting โดยกดที่ปุ่ม หลังคุณสมบัติ String แล้วจะปรากฏหน้าต่างดังรูป ให้กรอกข้อความตามรูป และขอให้เว้นวรรคหนึ่งวรรคก่อนจะเริ่มพิมพ์ เพื่อความสะดวกในการแสดงผล



เมื่อกรอกข้อความเสร็จแล้วให้กด OK เราจะเห็นว่าค่าที่แสดงใน Property Inspector จะแสดง เฉพาะค่าแรกเท่านั้น และให้กำหนดค่า Value ให้เท่ากับ 1

- สำหรับ List Box ให้กำหนด Tag เป็น HomeList และให้กรอกคุณสมบัติ Sting โดยกดที่ปุ่ม หลังคุณสมบัติ String แล้วจะปรากฏหน้าต่างดังรูป ให้กรอกข้อความตามรูป และขอให้เว้นวรรคหนึ่งวรรคก่อนจะเริ่มพิมพ์ เพื่อความสะดวกในการแสดงผล



เมื่อกรอกข้อความเสร็จแล้วให้กด OK เราจะเห็นว่าค่าที่แสดงใน Property Inspector จะแสดงเฉพาะค่าแรกเท่านั้น กำหนดค่าคุณสมบัติ Value ให้เท่ากับ 1 และกำหนดค่าคุณสมบัติ Enable ให้เป็น off

- สำหรับ Figure นั้นให้กำหนดคุณสมบัติ Name เป็น Product Selection

ซึ่งเมื่อกำหนดค่าตามที่กล่าวแล้วก็ถือว่าเสร็จสิ้นการกำหนดคุณสมบัติ สำหรับคุณสมบัติอื่นๆ เช่นลักษณะตัวอักษร ขนาดต่างๆ เราไม่ถือว่าเป็นสาระสำคัญในที่นี้

Save และ Activate GUI

ขั้นต่อไปให้เรา Save GUI ของเราโดยเลือก Save จากเมนู File หรือใช้เมาส์กดที่ปุ่ม ก็ได้จากนั้น MATLAB จะถามชื่อไฟล์ที่เราต้องการเก็บ ซึ่งในที่นี้เราจะตั้งชื่อว่า product ซึ่งเมื่อเราเลือก OKแล้ว MATLAB ก็จะทำการเก็บเป็นสองไฟล์คือ product.fig และ product.m

จากนั้นเมื่อเรา activate GUI จะทำให้ MATLAB เปิดหน้าต่างของ GUI ของเราขึ้นมา และEditor/Debugger ก็จะถูกเปิดขึ้นมาพร้อมกับเปิดไฟล์ product.m ขึ้น

ปรับปรุง และเพิ่มเติมชุดคำสั่งให้ GUI ทำงานตามที่ต้องการ

ลักษณะของ GUI ที่เรากำลังจะทำนี้ จะมีลักษณะเหมือนกับการเก็บข้อมูลของวัตถุต่างๆที่อยู่ใน GUI แล้วนำมาแสดงผล ดังนั้นหากเราจะเขียนให้ทุก Callback Subfunction มีระบบการประเมินผลของตัวเองก็จะทำให้โปรแกรมมีขนาดใหญ่มากโดยไม่จำเป็น ดังนั้นเราจึงจะเขียน Subfunction ขึ้นมาอันหนึ่งเพื่อทำหน้าที่ประเมินผล ดังนั้นทุก Subfunction ที่เป็น Callback ของวัตถุต่างๆ ก็จะถูกเรียกมาที่ Subfunction นี้โดยตรง ยกเว้นว่ามีหน้าที่พิเศษที่ต้องดำเนินการก่อนนี้ เราจะเรียก Subfunction นี้ว่า manager ซึ่งให้เราเพิ่มเข้าไปในส่วนท้ายของ Application M-file product.m ดังนี้

```
%-----------------------------------------------------------

function manager(handles)
```

```
nselect = get(handles.Popup1,'Value');
pop_string = get(handles.Popup1,'String');
ProSelect = pop_string(nselect);
string_display = strcat('You have selected',ProSelect);

if get(handles.Work,'Value') == 1;
        string_display =strcat(string_display,'for use it at work');
        if get(handles.Home,'Value') ==1;
                Nselect = get(handles.HomeList,'Value');
                list_string = get(handles.HomeList,'String');
                HomeSelect = list_string(Nselect);
                string_display=strcat(string_display,'and use it at home',
                HomeSelect);
                Getting 286 tting Start with GUIDE
        end
elseif get(handles.Home,'Value') ==1;
        nselect = get(handles.HomeList,'Value');
        list_string = get(handles.HomeList,'String');
        HomeSelect = list_string(nselect);
        string_display=strcat(string_display,'and use it at home',
        HomeSelect);
end
set(handles.Result,'String',string_display);
```

ขั้นตอนการทำงานเราจะกล่าวคร่าวๆ คือ subfunction นี้ต้องการ input ตัวเดียวคือ handles ที่จะให้ค่า handle ของวัตถุต่างๆ จากนั้นขั้นแรกก็จะเริ่มพิจารณาจาก Popup Menu ว่าผู้ใช้เลือกค่าใด หลังจากนั้นก็จะดึง String ที่เลือกมาเก็บไว้ แล้วนำมาต่อกับข้อความใหม่เพื่อสร้างเป็นประโยคขึ้น สำหรับคำสั่ง strcat เป็นคำสั่งที่ใช้รวมตัวแปรประเภท String ตั้งแต่สองตัวขึ้นไปเข้าด้วยกัน โดยจะตัดช่องว่างท้ายตัวแปร string นั้นออกไป ทำให้การกรอกคุณสมบัติของ Popup Menu และ List Box นั้นต้องเว้นวรรคไว้หนึ่งช่องก่อนแล้วจึงเริ่มพิมพ์ ซึ่งในกรณีอื่นอาจไม่จำเป็นต้องทำเช่นนี้

จากนั้นในส่วนของ if – elseif –end นั้นก็จะเป็น Logic ซึ่งสร้างขึ้นมาเพื่อตรวจสอบว่าผู้ใช้ได้เลือก check box ใดบ้าง และจะมีการเพิ่มข้อความต่อไปให้เป็นประโยคได้อย่างไร โดยข้อความที่เพิ่มเติมเข้าไปนั้นส่วนหนึ่งก็จะเป็นการนำเอา string ของ list box มาเพิ่มเติมเข้าไปในประโยคด้วย เราจะไม่ขอกล่าวถึงรายละเอียดมากกว่านี้ สำหรับตัว Subfuction อื่นๆ ที่มีอยู่ใน Application M-file เราก็เพียงเพิ่มคำสั่ง manager(handles) เข้าไปเท่านั้น ยกเว้นใน Subfunction ของ Check Box ที่เรียก At

Home เพราะเราต้องการเพิ่มข้อกำหนดว่าหากว่าไม่มีการเลือกใช้ให้เราทำให้ List Box ที่อยู่ข้างล่างนั้นใช้งานไม่ได้ ซึ่งจะมีคำสั่งในลักษณะต่อไปนี้

```
% --------------------------------------------------------------------
function varargout=Home_Callback(h, eventdata, handles, varargin)
% Stub for Callback of the uicontrol handles.Home.
if get(h,'Value') == 1
        set(handles.HomeList,'Enable','on');
else
        set(handles.HomeList,'Enable','off');
end
manager(handles)
% --------------------------------------------------------------------
```

ซึ่งการที่จะกำหนดว่าวัตถุนั้นใช้ได้หรือไม่ได้ก็ให้เรากำหนดค่าคุณสมบัติ Enable เป็น on หรือ off นั่นเอง

ทดสอบการทำงานของ GUI

ขั้นสุดท้ายเป็นการทดสอบการทำงานของ GUI ให้เราทำการ save file ที่ Editor/Debugger เสียก่อน จากนั้น Activate GUI ของเรา แล้วทดสอบการทำงาน ถ้าทำตามขั้นตอนที่กล่าวมาแล้ว GUI ไม่น่าจะมีปัญหาในการทำงาน

4 การเพิ่ม Object Handle หลังจากที่มีการสร้าง GUI

เมื่อเราทำการสร้าง GUI ด้วย GUIDE เรียบร้อยแล้วก็จะมีการเก็บข้อมูลลงใน FIG-file และ M-fileเราจะพบว่าได้มีการสร้างตัวแปร Structure ที่มีชื่อว่า handles โดยมีชื่อ field เป็นชื่อ Tag ของวัตถุนั้นและมีค่าของ field เป็นค่า handles ของวัตถุนั้น

อย่างไรก็ตาม การที่ MATLAB สร้างตัวแปร handles ขึ้นเป็นขั้นตอนเริ่มต้นการทำงาน ถ้าเราทำการเพิ่มวัตถุเข้าไปใน GUI เช่น ทำการเขียนกราฟขึ้น แล้วต้องการที่จะปรับปรุงคุณสมบัติของกราฟภายหลัง เราจำเป็นต้องทราบ handle ของเส้นกราฟนั้นเพื่อที่จะปรับปรุงคุณสมบัติ แต่ตัวแปร handlesที่เรามีอยู่นั้นไม่มีข้อมูลของเส้นกราฟเส้นนั้นอยู่ ทำให้เราจะต้องกำหนด ค่า handle ของวัตถุนั้นเพิ่มเข้าไป และอย่าลืมว่า ตัวแปร handles ที่ถูกสร้างขึ้นโดย GUIDE นั้นจะมีข้อมูลเฉพาะ uicontrol ที่มีอยู่ในตอนวาง layout GUI เท่านั้น จะไม่มีข้อมูลที่เกี่ยวข้องกับเส้นกราฟ หรือวัตถุอื่นๆที่สร้างขึ้นในภายหลัง ดังนั้นถ้าเราต้องการจะส่งผ่านข้อมูลอื่นๆ ที่เกิดขึ้นภายหลังทั้งที่เป็นวัตถุใน GUI หรือนำข้อมูลประเภทต่างๆไปใช้ใน Subfunction อื่นๆ เราจำเป็นที่จะต้องมีการแก้ไขและปรับปรุงตัวแปร handles แล้วนำไปเก็บไว้ในที่ซึ่งสามารถที่จะเรียกใช้ได้ในทุก Subfunction ที่ต้องการ

ตัวอย่างต่อไปนี้จะแสดงวิธีการแก้ไข เพิ่มเติม ปรับปรุง และบันทึกข้อมูลที่ต้องการส่งผ่านระหว่าง Subfunction ต่างๆ ที่มีอยู่ใน GUI เพื่อให้เรามีความคล่องตัวในการสร้าง GUI ของเรามากยิ่งขึ้น

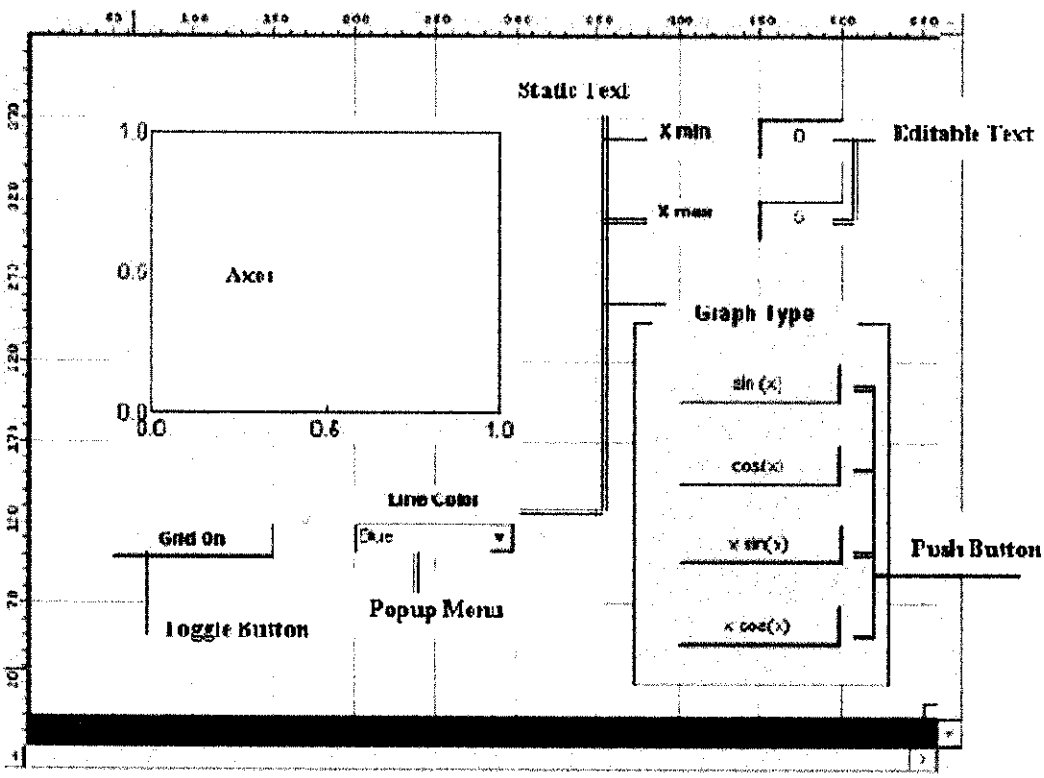
แต่ก่อนที่จะกล่าวถึงตัวอย่างเราขออธิบายคำสั่ง สองคำสั่งที่สำคัญในการสร้าง เก็บและเรียกใช้ข้อมูลใน GUI คำสั่งดังกล่าวคือ guihandles และ guidata ซึ่งมีรายละเอียดดังนี้

handles = guihandles(fig); คำสั่ง guihandles จะสร้างตัวแปรแบบ Structure จากรูปภาพที่มี handle เท่ากับ fig โดยให้ field ทั้งหมดในตัวแปรนั้นคือวัตถุต่างๆที่มีอยู่ในรูปภาพที่กำหนด ในตัวอย่างนี้ตัวแปรจะมีชื่อว่า handles สำหรับรูปที่ handle มีค่าเท่ากับ3610 . fig และชื่อ field นี้ก็จะใช้ชื่อ Tag ของวัตถุต่างๆ เหล่านั้น

guidata[fig, handles] คำสั่งนี้เป็นคำสั่งที่ให้เก็บค่าตัวแปร handles เข้าไปเป็นส่วนหนึ่งของรูปที่สร้างขึ้นคำสั่ง guidata นี้เป็นคำสั่งที่ใช้เก็บข้อมูล หรือดึงข้อมูลที่มีอยู่จากรูปออกมา การเขียนคำสั่งในลักษณะเช่นนี้จะทำให้ MATLAB ทำการเก็บตัวแปรชื่อ handles นี้เข้าไว้ให้อยู่ในรูป ซึ่งเราสามารถที่จะเรียกข้อมูลจากรูปออกมาได้ในภายหลัง ส่วนสำคัญของคำสั่งนี้ก็คือหากในภายหลังเราได้มีการเพิ่มเติมข้อมูลสำคัญเข้าไปอยู่ใน GUI และต้องการส่งต่อหรือมีการปรับเปลี่ยนค่าตัวแปร handles นี้ไม่ว่าด้วยเหตุผลใด เราจะต้องใช้คำสั่งนี้ใหม่อีกครั้งหนึ่งเพื่อให้เก็บข้อมูลที่ได้รับการปรับแก้แล้วเข้าไปอยู่ในรูป

ดังนั้นถ้าหากว่าเราต้องการที่จะเพิ่มข้อมูลให้กับรูป แล้วส่งผ่านไปยัง Subfunction อื่นๆต่อไปเราแนะนำให้คุณเก็บข้อมูลนั้นไว้ในตัวแปรที่ชื่อ handles นี้ แล้วใช้คำสั่ง guidata ทำการเก็บข้อมูลนี้เข้าไปไว้ในรูป เมื่อมีการเรียก Callback คำสั่งใน Callback จะกำหนดให้มีการดึงข้อมูลจากภายในรูปขึ้นมา แล้วส่งให้กับ Subfunction ใน Application M-file ดังนั้น Subfunction ใหม่ก็จะทราบค่าการเปลี่ยนแปลงตัวแปร handles นี้ได้

สำหรับตัวอย่างนี้เราจะสร้างกราฟขึ้นแล้วให้ผู้ใช้สามารถปรับปรุงคุณลักษณะของเส้นกราฟได้ GUI ของเราจะมีลักษณะดังนี้



วางรูปแบบใน GUIDE Layout Editor

สร้าง GUI ดังรูปข้างบน โดยใช้ Uicontrol และวัตถุอื่นๆ ตามแบบที่ได้กำหนดไว้ในรูป

กำหนดคุณสมบัติวัตถุ

กำหนดคุณสมบัติของวัตถุต่างๆ ที่มีอยู่ใน GUI ตามตารางต่อไปนี้

Uicontrol	Property name	Property value
Figure	Name	My Graph
Axes	Next Plot	Replacechildren
Editable Text (ตัวบน)	Tag	Xmin
	String	0
	Blackgroundcolor	เลือกให้เป็นสีขาว
Editable Text (ตัวล่าง)	Tag	Xmax
	String	5
	Blackgroundcolor	เลือกให้เป็นสีขาว
Push Button	Tag	Size
(sin(x))	String	sin (x)
Uicontrol	**Property name**	**Property value**
Push Button	Tag	Cos
(cos(x))	String	cos(x)
Push Button	Tag	XSine
(x sin(x))	String	x sin (x)
Push Button	Tag	XCos
(x cos(x))	String	x cos (x)
Toggle Button	Tag	GridOn
	String	Grid On
	Value	0
Popup Menu	Tag	Popupmenu2
	String	Blue, Green, Red, Yellow, Black, Pink (โดยให้แต่ละตัวอยู่คนละบรรทัด เหมือนในตัวอย่างก่อนหน้านี้)
Static Text	String	ตามรูปข้างบน

เมื่อแก้ไขคุณสมบัติต่างๆ เรียบร้อยแล้ว ก็เป็นการเสร็จสิ้นในขั้นตอนนี้

Save และ Activate GUI

ขั้นต่อไปให้เรา Save GUI ของเราโดยเลือก Save จากเมนู File หรือใช้เมาส์กดที่ปุ่ม ก็ได้จากนั้นMATLAB จะถามชื่อไฟล์ที่เราต้องการเก็บ ซึ่งในที่นี้เราจะตั้งชื่อว่า My_graph ซึ่งเมื่อเราเลือก OKแล้ว MATLAB ก็จะทำการเก็บไฟล์เป็น My_graph.fig และ My_graph.m

จากนั้นเมื่อเรา activate GUI จะทำให้ MATLAB เปิดหน้าต่างของ GUI ของเราขึ้นมา และEditor/Debugger ก็จะถูกเปิดขึ้นพร้อมกับเปิดไฟล์ My_graph.m ขึ้นมา

ปรับปรุง และเพิ่มเติมชุดคำสั่งให้ GUI ทำงานตามที่ต้องการ

ขั้นตอนต่อไปเป็นการปรับปรุงและแก้ไข subfunction ที่อยู่ภายใน application M-file ซึ่งมีสิ่งที่เราต้องทำเพิ่มเติมดังนี้

ขั้นตอนที่ 1 การเขียน subfunction ให้กับ push button ต่างๆ คือการสร้างเส้นกราฟลงไปในแกนที่เรากำหนดไว้ สำหรับปุ่ม Sine จะมีลักษณะของ code เป็นดังนี้

```
1 function varargout=Sine_Callback(h,eventdata,handles,varargin)
2 % Stub for Callback of the uicontrol handles.Sine.
3 set(handles.figure1,'HandleVisibility','on');
4 xmin = str2num(get(handles.Xmin,'String'));
5 xmax = str2num(get(handles.Xmax,'String'));
6 x=linspace(xmin,xmax,200);
7 y=sin(x);
8 LinePlot=plot(x,y);
9 handles.LineX = LinePlot;
10 guidata(gcbo, handles);
11 popupmenu2_Callback(h, eventdata, handles, varargin);
12 set(handles.figure1,'HandleVisibility','off');
```

หมายเลขข้างหน้าที่ใส่ไว้นั้นเพื่อความสะดวกในการอธิบาย เราไม่จำเป็นต้องใส่ลงไปใน Code สำหรับ บรรทัดที่ 1 เป็น function, บรรทัดที่ 3 นั้นเราจะต้องทำให้รูป GUI ของเรานั้นสามารถที่จะให้handle ของมันมองเห็นได้ เพราะก่อนหน้านี้เราสั่งให้ handle ของรูปภาพนี้มองไม่เห็นเพื่อไม่ให้สามารถที่จะ plot กราฟลงในหน้าต่าง GUI ของเราได้ ในที่นี้เราอาจจะไม่ set เป็น ON แต่ set เป็น Callback ก็ได้คือให้มองเห็นได้จากคำสั่งที่มาจาก Callback แต่จะมองไม่เห็นจากคำสั่งที่มาจาก Command windows ถ้าเรา set ไว้ที่ ON นี้จะสามารถมองเห็นได้จากทุกที่ เราเลือก On เพราะเราคาดหวังว่าในระหว่างนี้คงไม่มีการสั่ง plot จาก Command windows

บรรทัดที่ 4 และ 5 เป็นการเรียกค่า string จาก edit text ทั้งสองแล้วนำมาเปลี่ยนเป็นตัวเลข เพื่ออ่านค่าว่า

ค่าสูงสุดและต่ำสุดเป็นเท่าใด บรรทัดที่ 6, 7 และ 8 เป็นการสร้างกราฟและสั่งเขียนกราฟตามปกติ และเก็บ handle ของกราฟนั้นไว้ในชื่อ LinePlot

สำหรับบรรทัดที่ 9 นี้เป็นการสร้าง field ใหม่เพิ่มเข้าไปในตัวแปร handles โดยใช้ชื่อ field ว่า LineX และให้มีค่าเท่ากับ handle ของเส้นกราฟที่เราเพิ่งจะเขียนขึ้น

จากนั้นในบรรทัดที่ 10 เราจะทำการเก็บค่าตัวแปรนี้ไว้ในรูปภาพ สำหรับกรณีนี้เราจะใช้ gcbo ซึ่งหมายถึง get callback object ซึ่งหมายถึง push button นี้ หรือจะใช้ handles.figure1 ซึ่งเป็น handle ของหน้าต่าง GUI ของเราก็ได้ เพราะอย่างไรค่าเหล่านั้นจะถูกเก็บไว้ที่หน้าต่าง GUI เช่นเดียวกัน การเลือกให้เก็บที่ gcbo นั้นโดยแท้จริง MATLAB จะนำไปเก็บไว้ที่ parent ที่เป็นรูปที่บรรจุของวัตถุนั้นไม่ใช่เก็บไว้วัตถุนั้นแต่อย่างใด ทุกครั้งที่เราทำการเปลี่ยนแปลงค่า handles เราจะต้องทำการเก็บค่าที่เปลี่ยนแปลง เพื่อให้ Subfunction อื่นได้รับทราบการเปลี่ยนแปลงนี้ด้วย

ในบรรทัดที่ 11 จะเป็นการเรียก function ของ popup menu ที่ใช้ปรับสีของเส้นกราฟให้เป็นไปตามที่ผู้ใช้ต้องการ และในบรรทัด 12 เป็นการปิด handle ของรูปภาพไม่ให้ MATLAB สามารถมองเห็นได้

ส่วน Push Button อื่นๆ ก็จะมีลักษณะทำนองเดียวกันเพียงแต่ปรับเปลี่ยนคำสั่งที่ใช้ในการ plot เส้นกราฟให้มีลักษณะที่แตกต่างกันเท่านั้น ซึ่งมี Code ดังนี้คือ

```
function varargout=Cosine_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.Cosine.
set(handles.figure1,'HandleVisibility','on');
xmin = str2num(get(handles.Xmin,'String'));
xmax = str2num(get(handles.Xmax,'String'));
x=linspace(xmin,xmax,200);
y=cos(x);
LinePlot=plot(x,y);
handles.LineX = LinePlot;
guidata(gcbo, handles);
popupmenu2_Callback(h, eventdata, handles, varargin);
set(handles.figure1,'HandleVisibility','off');
% --------------------------------------------------------------------
function varargout=Xsine_Callback(h, eventdata, handles, varargin)
% Stub for Callback of the uicontrol handles.Xsine.
set(handles.figure1,'HandleVisibility','on');
xmin = str2num(get(handles.Xmin,'String'));
xmax = str2num(get(handles.Xmax,'String'));
x=linspace(xmin,xmax,200);
y=x.*sin(x);
```

```
LinePlot=plot(x,y);
handles.LineX = LinePlot;
guidata(gcbo, handles);
popupmenu2_Callback(h, eventdata, handles, varargin);
set(handles.figure1,'HandleVisibility','off');
% --------------------------------------------------------------------
function varargout=Xcosine_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.Xcosine.
set(handles.figure1,'HandleVisibility','on');
xmin = str2num(get(handles.Xmin,'String'));
xmax = str2num(get(handles.Xmax,'String'));
x=linspace(xmin,xmax,200);
y=x.*cos(x);
LinePlot=plot(x,y);
handles.LineX = LinePlot;
guidata(gcbo, handles);
popupmenu2_Callback(h, eventdata, handles, varargin);
set(handles.figure1,'HandleVisibility','off');
% --------------------------------------------------------------------
```

สำหรับ Popup menu ที่ใช้เลือกสีกราฟนั้นจะมี Code ดังนี้

```
% --------------------------------------------------------------------
function varargout=popupmenu2_Callback(h,eventdata,handles,varargin)
% Stub for Callback of the uicontrol handles.popupmenu2.
Getting 292 tting Start with GUIDE
col = get(handles.popupmenu2,'Value');
LinePlot=handles.LineX;
if col == 1
        set(LinePlot,'Color',[0 0 1]');
elseif col == 2
        set(LinePlot,'Color',[0 1 0]');
elseif col == 3
        set(LinePlot,'Color',[1 0 0]');
```

```
elseif col == 4
        set(LinePlot,'Color',[1 1 0]');
elseif col == 5
        set(LinePlot,'Color',[0 0 0]');
elseif col == 6
        set(LinePlot,'Color',[1 0 1]');
end
% --------------------------------------------------------
```

เป็นการสั่งให้เปลี่ยนสีเส้นกราฟ ที่บรรจุอยู่ใน handles.LineX ให้เป็นไปตามต้องการ โดยการกำหนดสีก็จะเป็นการผสมสี RGB นั่นเอง

อันดับต่อไป Callback ของ Toggle Button จะมี Code ดังนี้

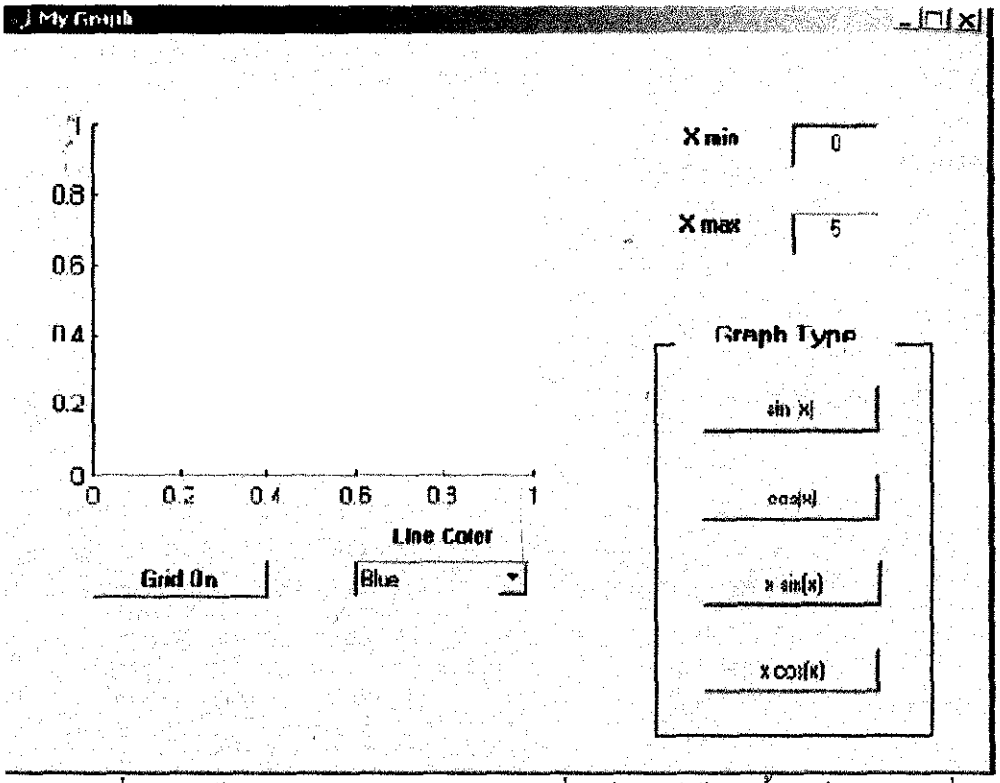
```
% --------------------------------------------------------
function varargout=GridOn_Callback(h,eventdata, handles, varargin)
% Stub for Callback of the uicontrol handles.GridOn.
set(handles.figure1,'HandleVisibility','on');
status = get(h,'Value');
if status == 0
set(h,'String','Grid On');
grid off
else
set(h,'String','Grid Off');
grid on
end
set(handles.figure1,'HandleVisibility','off');
% --------------------------------------------------------
```

ส่วนนี้ก็จะเป็นการกำหนดตัวหนังสือที่อยู่บน Toggle นี้ให้เปลี่ยนไป เมื่อมีการเลือกกดปุ่มนี้และจะเห็นว่าตอนต้นและตอนท้าย code จะมีการเปิด และ ปิด handle ของรูปภาพนี้ด้วย ส่วนคำสั่ง grid on และ grid off นั้นเป็นคำสั่งตามปกติของกราฟทั่วๆไป

สำหรับ Subfucntion ที่เป็น Callback ของ Editable Text Max และ Min นั้นไม่ต้องเขียนเพิ่มเติมและเราก็จะไม่ใช้ในที่นี้ โดยเราจะลบออกทั้งหมด หรือจะลบออกเฉพาะในส่วนที่ให้แสดงข้อความว่ายังไม่มีการกำหนด Callback ให้กับวัตถุนี้ออกไปเท่านั้นก็ได้

ทดสอบการทำงานของ GUI

จากนั้นลอง Save Application M-file แล้วลอง activate GUI ของเราดู เราจะเห็นหน้าต่างต่อไปนี้ปรากฏขึ้น



ซึ่งเราสามารถที่จะลองให้ GUI เขียนกราฟในลักษณะต่างๆ ที่เราต้องการได้ จากนั้นเราก็จะสามารถที่จะเปลี่ยนสีกราฟต่างๆ เหล่านั้นได้ ข้อเสียของ GUI นี้ก็คือถ้าให้มีการเปลี่ยนสีกราฟก่อนที่จะมีการเขียนกราฟขึ้นมันก็จะเกิดการ error ขึ้นทันที เพราะก่อนที่จะมีการสั่งให้ plot graph นั้นเรายังไม่มีการกำหนดตัวแปรชื่อhandles.LineX ลงไป ซึ่งเราอาจจะแก้ไขข้อเสียนี้ได้หลายวิธี ในที่นี้เราจะเสนอวิธีการเขียนกราฟลงไปในช่วงของการเปิด GUI นี้ขึ้นมา พร้อมทั้งมีการปรับแก้ตัวแปร handles นี้ไว้ก่อนที่ผู้ใช้จะเริ่มทำการใช้ GUI

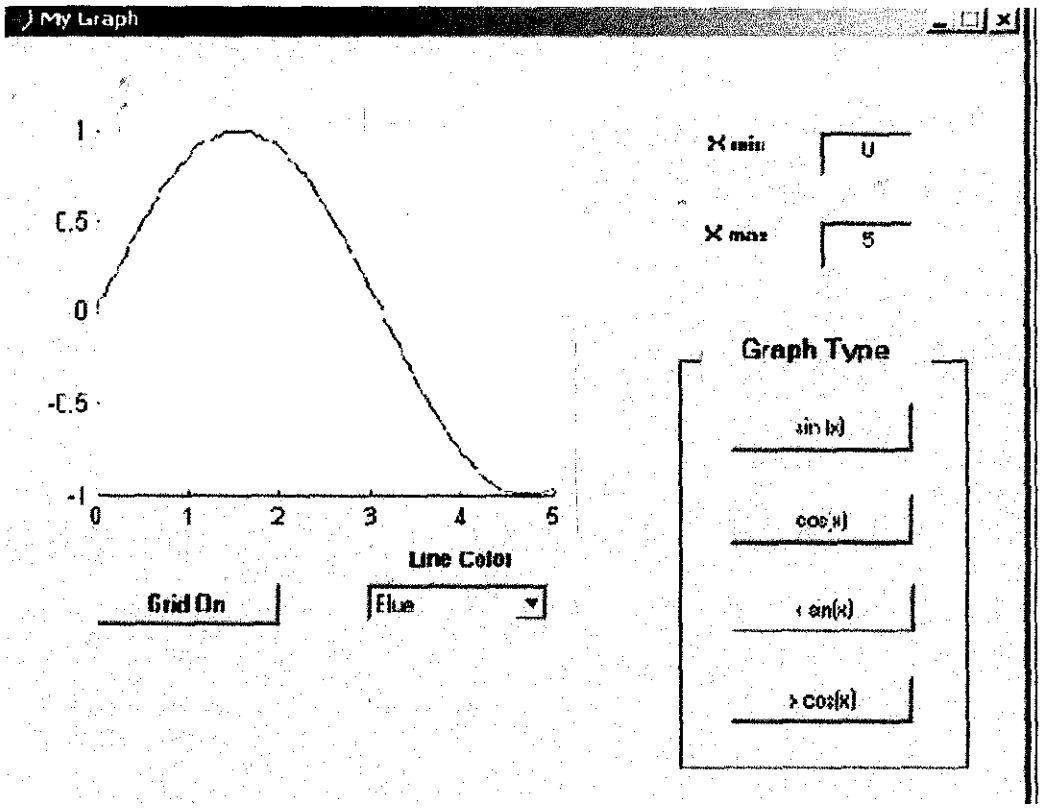
ไปที่ Application M-file ในช่วงบน แล้วมองหาคำสั่ง guidata และหลังจากคำสั่ง guidata ที่เป็นคำสั่งเดิมให้เราสั่งคำสั่งเพื่อให้มีการสร้างเส้นกราฟขึ้นแล้วเก็บค่าใหม่ ด้วยคำสั่ง guidata อีกครั้งหนึ่งลักษณะคำสั่งเป็นดังนี้

```
set(fig,'HandleVisibility','on');
xmin = str2num(get(handles.Xmin,'String'));
xmax = str2num(get(handles.Xmax,'String'));
x=linspace(xmin,xmax,200);
y=sin(x);
LinePlot=plot(x,y);
handles.LineX = LinePlot;
```

guidata(fig, handles);

set(fig,'HandleVisibility','off');

ซึ่งคำสั่งนี้จะทำให้เกิดการสร้างกราฟขึ้นก่อน พร้อมกับปรับปรุงข้อมูลของ handles ใน GUI พร้อมกันไปด้วย จากนั้น Save Application M-file แล้วทำการ activate GUI ของเราอีกครั้งหนึ่ง เราจะได้หน้าต่างต่อไปนี้



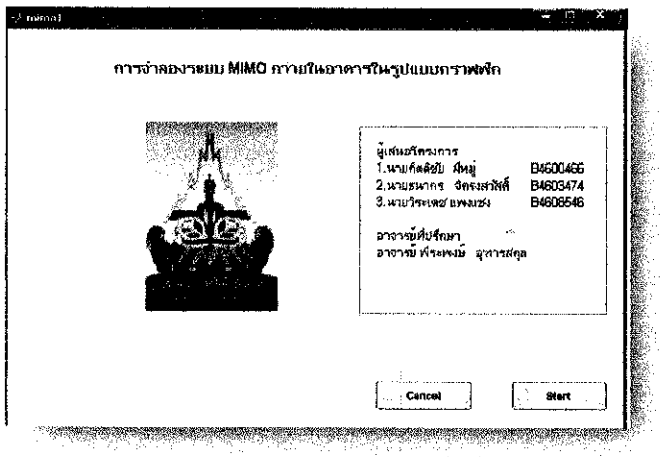
จะเห็นว่าเราสามารถแก้ไขปัญหาที่เรากล่าวถึงก่อนหน้านั้นได้ จากการแก้ปัญหาในจุดนี้เราจะเห็นว่าเราสามารถที่จะเพิ่มเติมข้อมูลต่างๆ เพื่อจะส่งผ่านไปยัง Subfunction เมื่อเริ่มต้นการทำงานของGUI ได้เช่นกัน

สำหรับการแก้ไข GUI นี้ในจุดอื่นๆ ก็สามารถทำได้เช่นกัน ทั้งนี้เป็นไปตามความต้องการของผู้ออกแบบ GUI นั่นเอง

ภาคผนวก ข

คำสั่งการทำงานของโปรแกรม GUI

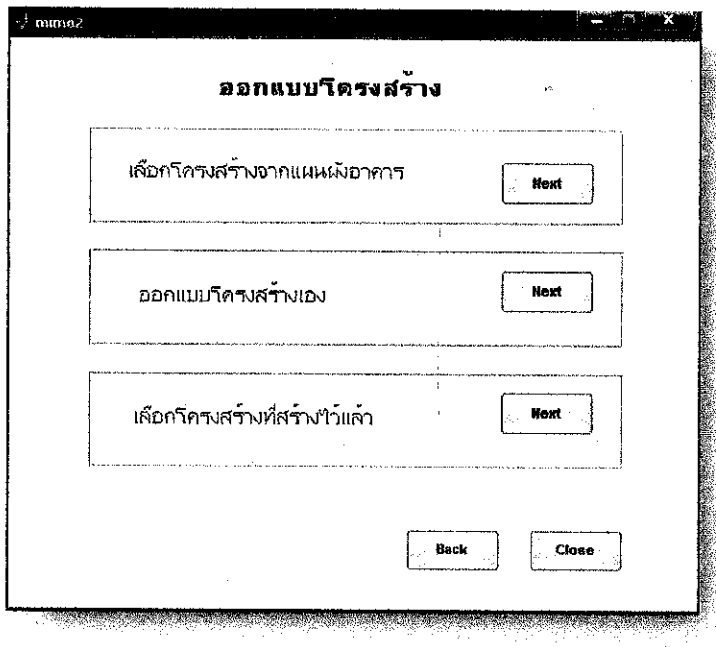
1. เริ่มใช้งานโปรแกรม




```
function varargout = mimo1(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo1_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function mimo1_OpeningFcn(hObject, eventdata, handles, varargin)
imshow('symbol.jpg');

handles.output = hObject;
guidata(hObject, handles);

function varargout = mimo1_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function pushbutton1_Callback(hObject, eventdata, handles)

 %--------------------------------------------------
close;
```

```
mimo2;
% --------------------------------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)
% --------------------------------------------------------------------
close;
% --------------------------------------------------------------------
```

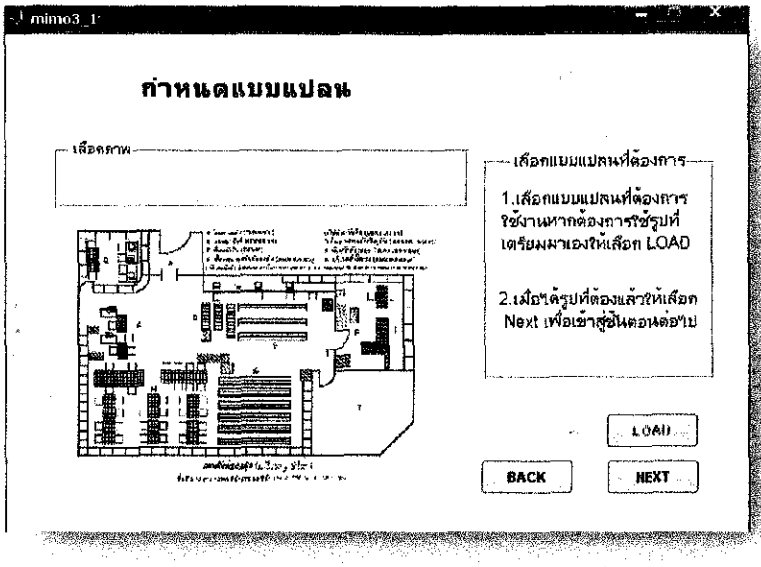
2. เลือกวิธีการออกแบบโครงสร้าง




```
function varargout = mimo2(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo2_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function mimo2_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

guidata(hObject, handles);
```

```
function varargout = mimo2_OutputFcn(hObject, eventdata, handles)

function pushbutton1_Callback(hObject, eventdata, handles)
close;
mimo3_1;

function pushbutton2_Callback(hObject, eventdata, handles)

% --------------------------------------------------

close;
mimo4_1;
% --------------------------------------------------

function pushbutton3_Callback(hObject, eventdata, handles)

% --------------------------------------------------
close;
mimo1;
% --------------------------------------------------

function pushbutton4_Callback(hObject, eventdata, handles)

% --------------------------------------------------
close;
% --------------------------------------------------

function pushbutton5_Callback(hObject, eventdata, handles)

% --------------------------------------------------
close;
mimo5_1;
% --------------------------------------------------
```

3. เลือกโครงสร้างจากแผนผังอาคาร

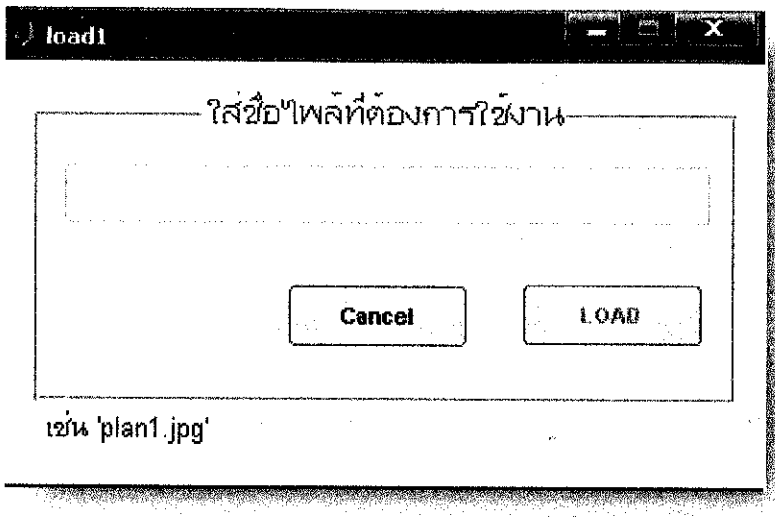



```
function varargout = mimo3_1(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo3_1_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo3_1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function mimo3_1_OpeningFcn(hObject, eventdata, handles, varargin)
1
handles.output = hObject;

guidata(hObject, handles);

function varargout = mimo3_1_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function popupmenu1_Callback(hObject, eventdata, handles)

%--------------------------

val = get(handles.popupmenu1,'Value');
```

```
cla;
switch val
   case 1
      B1=imread('Plan1.JPG');
      imshow('Plan1.JPG');
      pic=B1;
      save pic.mat pic;

   case 2
      B2=imread('Plan2.JPG');
      imshow('Plan2.JPG');
      pic=B2;
    save pic.mat pic;
    case 3
      B2=imread('floor1.JPG');
      imshow('floor1.JPG');
      pic=B2;
    save pic.mat pic;
    case 4
      B2=imread('floor2.JPG');
      imshow('floor2.JPG');
      pic=B2;
    save pic.mat pic;
    case 5
      B2=imread('floor3.JPG');
      imshow('floor3.JPG');
      pic=B2;
    save pic.mat pic;
    case 6
      B2=imread('floor4.JPG');
      imshow('floor4.JPG');
      pic=B2;
    save pic.mat pic;
  case 7
      B2=imread('F3_1st.JPG');
      imshow('F3_1st.JPG');
      pic=B2;
    save pic.mat pic;
    case 8
      B2=imread('F3_2nd.JPG');
      imshow('F3_2nd.JPG');
      pic=B2;
    save pic.mat pic;
end
%-----------------------------------
function popupmenu1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)
%-----------------------------------
close;
mimo2;
```

```
%-----------------------------
function pushbutton2_Callback(hObject, eventdata, handles)

% ---------------------------------------------------
load bwall.mat;
clear;
bwx1=0;
bwx2=0;
bwy1=0;
bwy2=0;
save bwall.mat;
%-------------------------
load gwall.mat;
clear;
gwx1=0;
gwx2=0;
gwy1=0;
gwy2=0;
save gwall.mat;
%-------------------------
load rwall.mat;
clear;
rwx1=0;
rwx2=0;
rwy1=0;
rwy2=0;
save rwall.mat;
%-------------------------
load ywall.mat;
clear;
ywx1=0;
ywx2=0;
ywy1=0;
ywy2=0;
save ywall.mat;
%-------------------------
close;
mimo3_2;
%-----------------------------------------

function pushbutton4_Callback(hObject, eventdata, handles)

%-----------------------------------------
close;
load1;
%-----------------------------------------
```

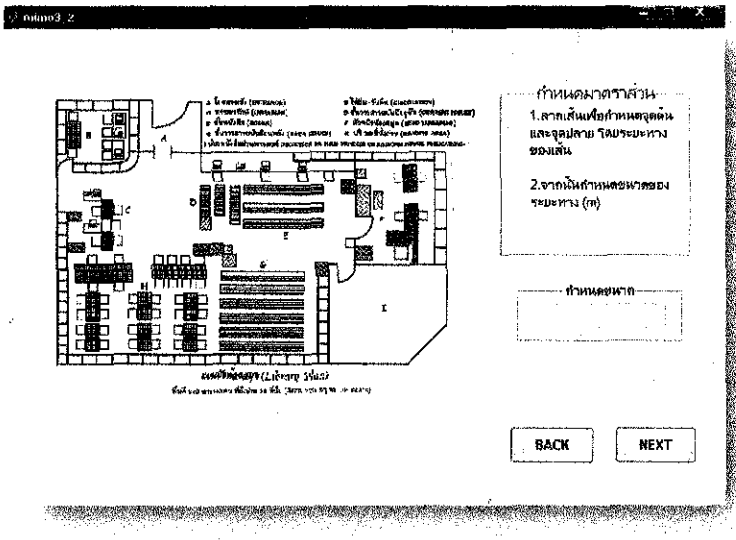
4. โหลดภาพโครงสร้าง




```
function varargout = load1(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @load1_OpeningFcn, ...
                   'gui_OutputFcn',  @load1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end


function load1_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

guidata(hObject, handles);

function varargout = load1_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function edit1_Callback(hObject, eventdata, handles)



function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end
```

```
function pushbutton1_Callback(hObject, eventdata, handles)

%--------------------------------------------------
x=get(handles.edit1,'String');
eval(['B = imread(' x ');']);
pic=B;
save pic.mat pic;
mimo3_2;
%--------------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%--------------------------------------------------
close;
mimo3_1;
%--------------------------------------------------
```

5. กำหนดมาตราส่วน

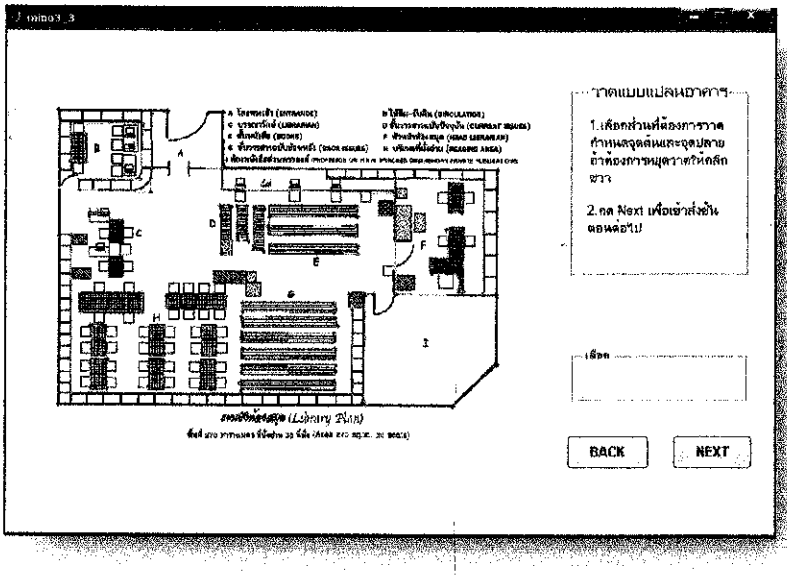



```
function varargout = mimo3_2(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo3_2_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo3_2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
  gui_mainfcn(gui_State, varargin{:});
end

function mimo3_2_OpeningFcn(hObject, eventdata, handles, varargin)

% ------------------------------------
load pic.mat;
imshow(pic);
hold on
% ------------------------------------
for i1=1:1
[x,y]=ginput(1);
lx(i1,1)=x;
ly(i1,1)=y;
[x,y]=ginput(1);
lx(i1,2)=x;
ly(i1,2)=y;
plot([lx(i1,1) lx(i1,2)],[ly(i1,1),ly(i1,2)]);
end
r=sqrt((lx(i1,2)-lx(i1,1))^2+(ly(i1,2)-ly(i1,1))^2);

save r.mat;
%------------------------------------

handles.output = hObject;
guidata(hObject, handles);


function varargout = mimo3_2_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;


function pushbutton1_Callback(hObject, eventdata, handles)

%------------------------------------
close;
mimo3_1;
%------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%------------------------------------
close;
mino3_3;
%------------------------------------

function edit2_Callback(hObject, eventdata, handles)

%------------------------------------
load r.mat
xline = str2int(get(hObject,'String'));
r=(r/xline);
save r.mat
%------------------------------------
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

6. วาดโครงสร้างตามภาพโครงสร้าง



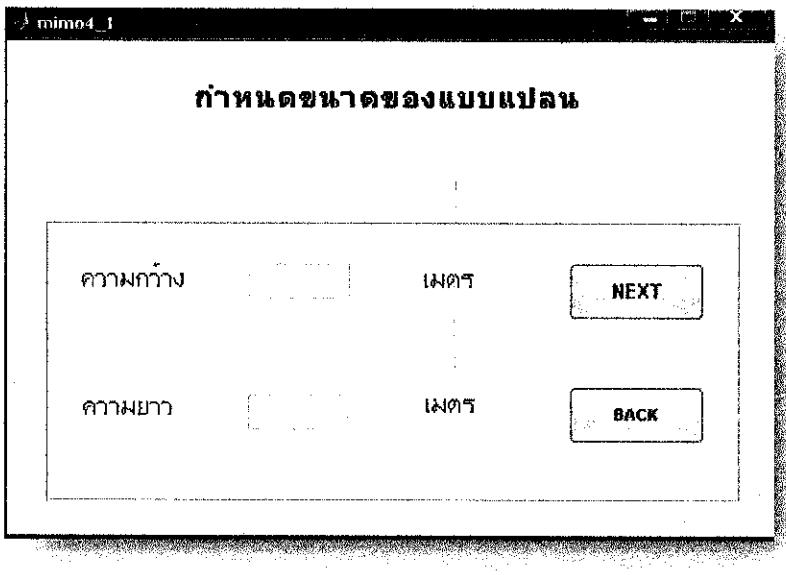

```
function varargout = mino3_3(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mino3_3_OpeningFcn, ...
                   'gui_OutputFcn',  @mino3_3_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function mino3_3_OpeningFcn(hObject, eventdata, handles, varargin)

% ---------------------------------------------

load pic.mat;
imshow(pic);
hold on;
%-----------------------------------------------
handles.output = hObject;
```

```
guidata(hObject, handles);

function varargout = mino3_3_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function popupmenu1_Callback(hObject, eventdata, handles)

%--------------------------------------------
val = get(hObject,'Value');
switch val
case 1
  hold on;


  but=1;   i=0;
  while but==1
    i=i+1;
    [x,y,but]=ginput(1);
    but
    if but==1
      bwx1(i)=x;
      bwy1(i)=y;
      [x,y,but]=ginput(1);
      bwx2(i)=x;
      bwy2(i)=y;
      plot([bwx1(i) bwx2(i)],[bwy1(i) bwy2(i)],'b.-');
      hold on;
      bwy1(i)=-1*bwy1(i);
      bwy2(i)=-1*bwy2(i);
    end

end
save bwall.mat
  case 2
    hold on;
    grid on;

    but=1;i=0;
    while but==1
      i=i+1;
      [x,y,but]=ginput(1);
      but
      if but==1
        gwx1(i)=x;
        gwy1(i)=y;
        [x,y,but]=ginput(1);
        gwx2(i)=x;
        gwy2(i)=y;
        plot([gwx1(i) gwx2(i)],[gwy1(i) gwy2(i)],'g.-');
        hold on;
        gwy1(i)=-1*gwy1(i);
        gwy2(i)=-1*gwy2(i);
      end
    end
save gwall.mat
  case 3
```

```
    hold on;
    grid on;

    but=1;i=0;
    while but==1
      i=i+1;
      [x,y,but]=ginput(1);
      but
      if but==1
        rwx1(i)=x;
        rwy1(i)=y;
        [x,y,but]=ginput(1);
        rwx2(i)=x;
        rwy2(i)=y;
        plot([rwx1(i) rwx2(i)],[rwy1(i) rwy2(i)],'r.-');
        hold on;
        rwy1(i)=-1*rwy1(i);
        rwy2(i)=-1*rwy2(i);
      end
    end
save rwall.mat
 case 4
    hold on;
    grid on;

    but=1;i=0;
    while but==1
      i=i+1;
      [x,y,but]=ginput(1);
      but
      if but==1
        ywx1(i)=x;
        ywy1(i)=y;
        [x,y,but]=ginput(1);
        ywx2(i)=x;
        ywy2(i)=y;
        plot([ywx1(i) ywx2(i)],[ywy1(i) ywy2(i)],'y.-');
        hold on;
        ywy1(i)=-1*ywy1(i);
        ywy2(i)=-1*ywy2(i);
      end
    end

     save ywall.mat
end
%--------------------------------------------------

function popupmenu1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)

%--------------------------------------------------
```

```
close;
mimo3_2;
%--------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%--------------------------------------------
mimo3_4;
%--------------------------------------------
```

7. กำหนดขนาดของโครงสร้าง

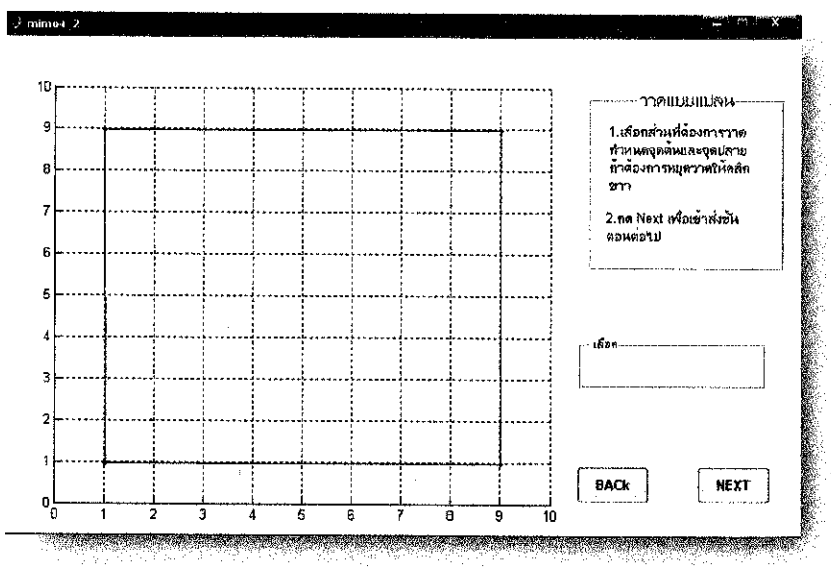



```
function varargout = mimo4_1(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo4_1_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo4_1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function mimo4_1_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

guidata(hObject, handles);
```

```
function varargout = mimo4_1_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;


function pushbutton1_Callback(hObject, eventdata, handles)
%--------------------------------------------
load bwall.mat;
clear;
bwx1=0;
bwx2=0;
bwy1=0;
bwy2=0;
save bwall.mat;
%--------------------------------------------

load gwall.mat;
clear;
gwx1=0;
gwx2=0;
gwy1=0;
gwy2=0;
save gwall.mat;
%--------------------------------------------

load rwall.mat;
clear;
rwx1=0;
rwx2=0;
rwy1=0;
rwy2=0;
save rwall.mat;
%--------------------------------------------

load ywall.mat;
clear;
ywx1=0;
ywx2=0;
ywy1=0;
ywy2=0;
save ywall.mat;
close;
mimo4_2;
%--------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%--------------------------------------------
close;
mimo2;
%--------------------------------------------

function edit2_Callback(hObject, eventdata, handles)

%--------------------------------------------
```

```
xa=str2num(get(hObject,'String'));
save xa.mat xa;
%---------------------------------------------

function edit2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit1_Callback(hObject, eventdata, handles)

%---------------------------------------------
ya=str2num(get(hObject,'String'));
save ya.mat ya;
%---------------------------------------------

function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

8. ออกแบบโครงสร้าง



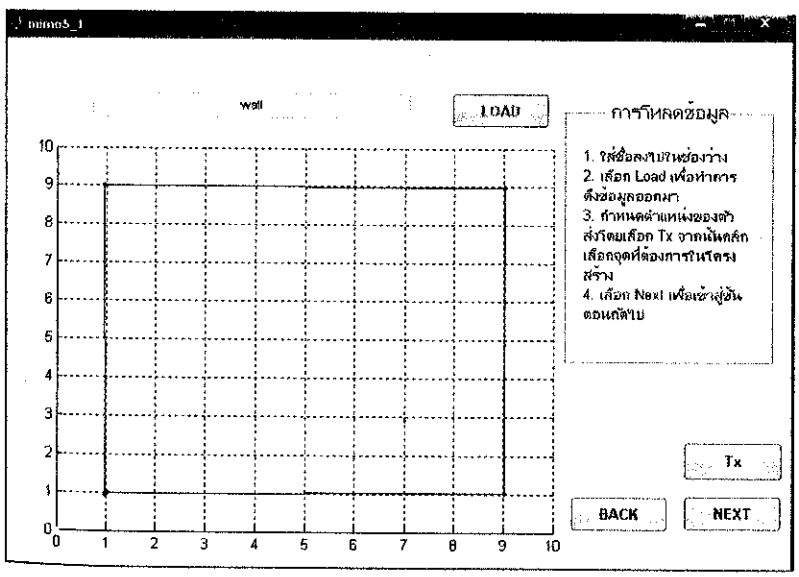

```
function varargout = mimo4_2(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo4_2_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo4_2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
```

```
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function mimo4_2_OpeningFcn(hObject, eventdata, handles, varargin)

%-----------------------------------------------
load xa.mat
load ya.mat

xmin=0;
xmax=xa;
ymin=0;
ymax=ya;
axis([xmin xmax ymin ymax]);
hold on;
%-----------------------------------------------

handles.output = hObject;

guidata(hObject, handles);

function varargout = mimo4_2_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function pushbutton1_Callback(hObject, eventdata, handles)

%-----------------------------------------------
close;
mimo4_1;
%-----------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%-----------------------------------------------
mimo3_4;
%-----------------------------------------------

function popupmenu1_Callback(hObject, eventdata, handles)

%-----------------------------------------------
val = get(hObject,'Value');
switch val
case 1
  hold on;
  grid on;

  but=1;   i=0;
  while but==1
```

```
    i=i+1;
    [x,y,but]=ginput(1);
    but
    if but==1
      bwx1(i)=x;
      bwy1(i)=y;
      [x,y,but]=ginput(1);
      bwx2(i)=x;
      bwy2(i)=y;
      plot([bwx1(i) bwx2(i)],[bwy1(i) bwy2(i)],'b.-');
      hold on;
    end

end
save bwall.mat
  case 2
    hold on;
    grid on;

    but=1;i=0;
    while but==1
      i=i+1;
      [x,y,but]=ginput(1);
      but
      if but==1
        gwx1(i)=x;
        gwy1(i)=y;
        [x,y,but]=ginput(1);
        gwx2(i)=x;
        gwy2(i)=y;
        plot([gwx1(i) gwx2(i)],[gwy1(i) gwy2(i)],'g.-');
        hold on;
      end
    end
  save gwall.mat
  case 3
    hold on;
    grid on;

    but=1;i=0;
    while but==1
      i=i+1;
      [x,y,but]=ginput(1);
      but
      if but==1
        rwx1(i)=x;
        rwy1(i)=y;
        [x,y,but]=ginput(1);
        rwx2(i)=x;
        rwy2(i)=y;
        plot([rwx1(i) rwx2(i)],[rwy1(i) rwy2(i)],'r.-');
        hold on;
      end
    end
save rwall.mat
  case 4
```

```
    hold on;
    grid on;

    but=1;i=0;
    while but==1
      i=i+1;
      [x,y,but]=ginput(1);
      but
      if but==1
        ywx1(i)=x;
        ywy1(i)=y;
        [x,y,but]=ginput(1);
        ywx2(i)=x;
        ywy2(i)=y;
        plot([ywx1(i) ywx2(i)],[ywy1(i) ywy2(i)],'y.-');
        hold on;
      end
    end

     save ywall.mat
end
%---------------------------------------------------
function popupmenu1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end
```

9. โหลดโครงที่สร้างที่สร้างไว้แล้ว

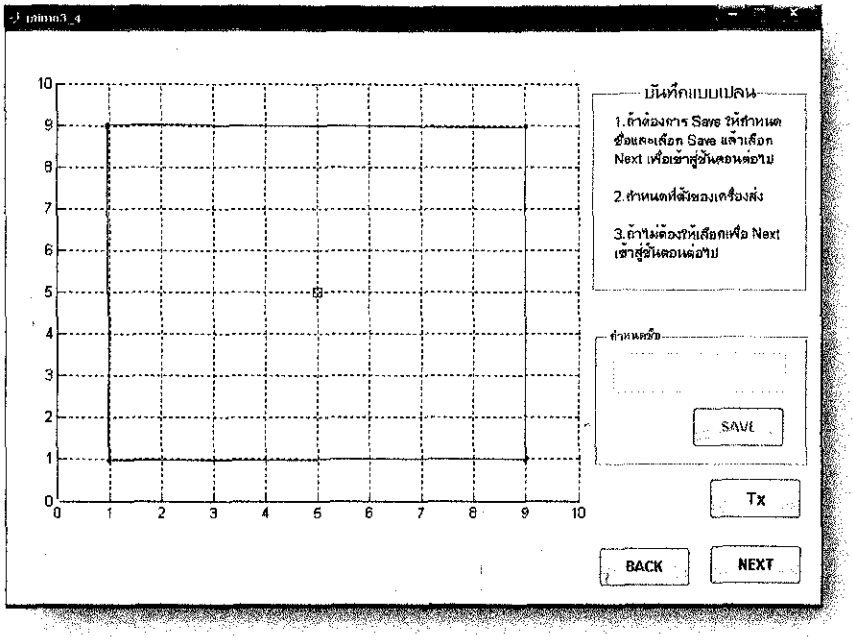



```
function varargout = mimo5_1(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo5_1_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo5_1_OutputFcn, ...
```

```
            'gui_LayoutFcn',  [] , ...
            'gui_Callback',   []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function mimo5_1_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);

function varargout = mimo5_1_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function edit1_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end


function pushbutton1_Callback(hObject, eventdata, handles)

%--------------------------------------------------
x=get(handles.edit1,'String');
eval(['load ' x '.mat;']);
hold on;
grid on;

% load bwall.mat;
  hold on;
for i=1:length(bwx1)
    plot([bwx1(i) bwx2(i)],[bwy1(i) bwy2(i)],'b.-');
  end
% load gwall.mat;
hold on;
for i=1:length(gwx1)
  plot([gwx1(i) gwx2(i)],[gwy1(i) gwy2(i)],'g.-');
end

% load rwall.mat
hold on;
for i=1:length(rwx1)
  plot([rwx1(i) rwx2(i)],[rwy1(i) rwy2(i)],'r.-');
end
hold on;
for i=1:length(ywx1)
  plot([ywx1(i) ywx2(i)],[ywy1(i) ywy2(i)],'y.-');
end
save wall.mat
```

```
%--------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%--------------------------------------------
close ;
mimo2;
%--------------------------------------------
function pushbutton3_Callback(hObject, eventdata, handles)

%--------------------------------------------
mimo5;
%--------------------------------------------

function pushbutton6_Callback(hObject, eventdata, handles)

%--------------------------------------------
[x,y]=ginput(1);
xt=x;
yt=y;
save Tx.mat;
%--------------------------------------------

function pushbutton7_Callback(hObject, eventdata, handles)

%--------------------------------------------
[x,y]=ginput(1);
xr=x;
yr=y;
save Rx.mat;
%--------------------------------------------
```

10.บันทึกโครงสร้างและกำหนดตำแหน่งของตัวส่ง

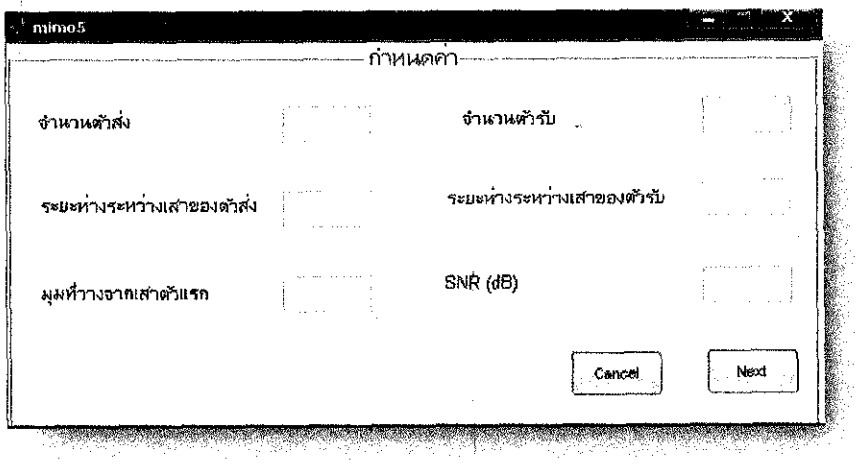



```
function varargout = mimo3_4(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo3_4_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo3_4_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end

function mimo3_4_OpeningFcn(hObject, eventdata, handles, varargin)

%--------------------------
grid on;
load bwall.mat;
hold on;

   for i=1:length(bwx1)
      plot([bwx1(i) bwx2(i)],[bwy1(i) bwy2(i)],'b.-');
   end
 load gwall.mat;
hold on;
   for i=1:length(gwx1)
```

```
  plot([gwx1(i) gwx2(i)],[gwy1(i) gwy2(i)],'g.-');
  end

load rwall.mat
hold on;
  for i=1:length(rwx1)
  plot([rwx1(i) rwx2(i)],[rwy1(i) rwy2(i)],'r.-');
  end
load ywall.mat
hold on;
  for i=1:length(ywx1)
  plot([ywx1(i) ywx2(i)],[ywy1(i) ywy2(i)],'y.-');
  end
save wall.mat
%--------------------------

handles.output = hObject;

guidata(hObject, handles);

function varargout = mimo3_4_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function edit1_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)

%---------------------------------------

aa=get(handles.edit1,'String');
load wall.mat
eval(['save ' aa '.mat']);
%---------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)

%--------------------------
close;
%--------------------------

function pushbutton3_Callback(hObject, eventdata, handles)

%--------------------------
Close all;
mimo5;
%--------------------------

function pushbutton4_Callback(hObject, eventdata, handles)

%--------------------------
[x,y]=ginput(1);
```

```
xt=x;
yt=y;
plot(xt,yt,'rs');
save Tx.mat;
%-------------------------
```

11. กำหนดค่าพารามิเตอร์

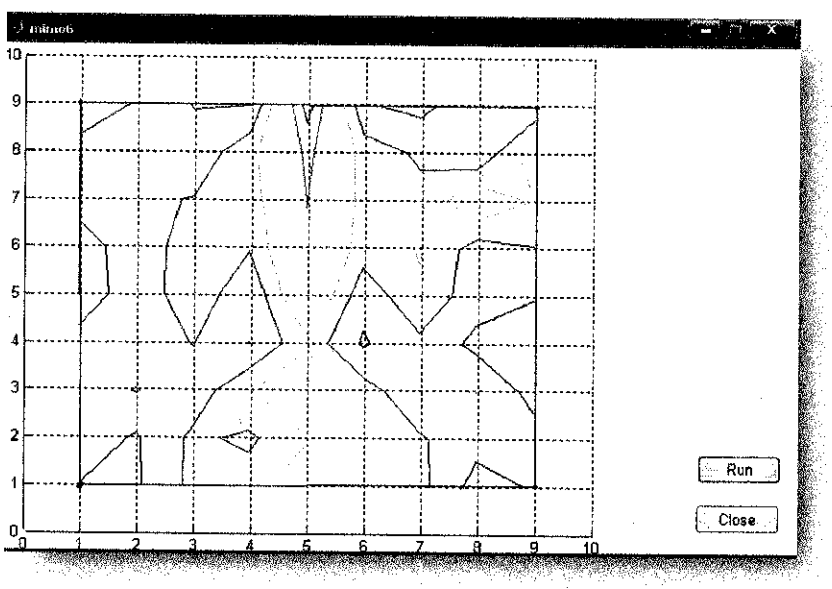



```
function varargout = mimo5(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo5_OpeningFcn, ...
                   'gui_OutputFcn',  @mimo5_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function mimo5_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);

function varargout = mimo5_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function edit1_Callback(hObject, eventdata, handles)
```

```
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end



function edit8_Callback(hObject, eventdata, handles)

function edit8_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)

function edit9_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end



function edit13_Callback(hObject, eventdata, handles)

function edit13_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)

%--------------------------------------------------------
Nt=str2num(get(handles.edit1,'String'));

Dt=str2num(get(handles.edit8,'String'));

ang=str2num(get(handles.edit9,'String'));

Snr_dB=str2num(get(handles.edit13,'String'));

Nr=str2num(get(handles.edit14,'String'));

Dr=str2num(get(handles.edit15,'String'));
save parameter.mat;

Close;
mimo6
%--------------------------------------------------------

function pushbutton2_Callback(hObject, eventdata, handles)
```

```
%--------------------------------------------------------
close;
mimo3_4;
%--------------------------------------------------------

function edit14_Callback(hObject, eventdata, handles)

function edit14_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit15_Callback(hObject, eventdata, handles)

function edit15_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```

12. Plot ค่า Capacity ของสัญญาณ




```
function varargout = mimo6(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @mimo6_OpeningFcn, ...
```

```
            'gui_OutputFcn',  @mimo6_OutputFcn, ...
            'gui_LayoutFcn',  [] , ...
            'gui_Callback',   []);
if nargin && ischar(varargin{1})
  gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
  [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
  gui_mainfcn(gui_State, varargin{:});
end

function mimo6_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = mimo6_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function pushbutton1_Callback(hObject, eventdata, handles)

%------------------------------------------------------------------
Close all;
%------------------------------------------------------------------

function edit1_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
  set(hObject,'BackgroundColor','white');
end

function pushbutton2_Callback(hObject, eventdata, handles)

%------------------------------------------------------------------
load wall.mat
load Tx.mat
hold on;
grid on;

[xc yc capa]=main()
%----[xc yc capa]=main_strength()              %  Use test Program

contour(xc,yc,capa);

hold on;

for i=1:length(bwx1)
    plot([bwx1(i) bwx2(i)],[bwy1(i) bwy2(i)],'b.-');
  end
hold on;
for i=1:length(gwx1)
  plot([gwx1(i) gwx2(i)],[gwy1(i) gwy2(i)],'g.-');
```

```
end

hold on;
for i=1:length(rwx1)
   plot([rwx1(i) rwx2(i)],[rwy1(i) rwy2(i)],'r.-');
end

hold on;
for i=1:length(ywx1)
   plot([ywx1(i) ywx2(i)],[ywy1(i) ywy2(i)],'y.-');
end
plot(xt,yt,'rs');
save wall.mat
%-----------------------------------------------------------------
```

ภาคผนวก ค

คำสั่งการทำงานของโปรแกรม Raytracing

Raytracing

```
Function H=RayTracing(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1
,ywy2,Tx,Rx)
 [Et Tcoef]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,
Tx,Rx);
Er=reflec(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Rx);
E=Er+Et;
H=E;
```

Reflec

```
function
Er=reflec(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Rx)
E0=1;
Lamda=(3*10^8)/(2.45*10^9);
di=sqrt((Rx(1)-Tx(1))^2+(Rx(2)-Tx(2))^2);
bEr=0;
gEr=0;
rEr=0;
yEr=0;
%---------- blue ----------
for i=2:length(bwx1)
  if abs(Rx(1)-Tx(1))<0.1 & (bwx2(i)~=bwx1(i))          %m1 = infinity
    m2=(bwy2(i)-bwy1(i))/(bwx2(i)-bwx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[bwy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[bwy1(i) Tx(1)];
        z2=[bwy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %========== find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        bEr=bEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    else
```

```
        c2=bwy1(i)-(m2*bwx1(i));
        Z(1)=m2*Rx(1)+c2;
        Z(2)=Rx(1);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
           z1=[bwy1(i) Tx(1)];
           z2=[bwy1(i) Rx(1)];
           d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
           h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
           h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
           r=sqrt((h1+h2)^2+d^2);
           D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
          Omaga=(2*pi*(r-di)*10/Lamda);
           ep=5-j*60*0.005*Lamda;
           Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
           %================ find transmission in reflection ==========
           [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
           [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
           bEr=bEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
        else
           Rf=1;
           Tr1=0;
           Tr2=0;
        end
     end
  elseif (Rx(1)~=Tx(1)) & abs(bwx2(i)-bwx1(i))<0.1          %m2 = infinity
     m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
     if abs(m1)<0.1
        Z=[Rx(2) bwx1(i)];
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
           z1=[Tx(2) bwx1(i)];
           z2=[Rx(2) bwx2(i)];
           d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
           h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
           h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
           r=sqrt((h1+h2)^2+d^2);
           D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
          Omaga=(2*pi*(r-di)*10/Lamda);
           ep=5-j*60*0.005*Lamda;
           Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
           %================ find transmission in reflection ==========
           [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
           [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
           bEr=bEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
        else
           Rf=1;
           Tr1=0;
           Tr2=0;
        end
     else
        c1=Tx(2)-(m1*Tx(1));
        Z(1)=m1*bwx2(i)+c1;
        Z(2)=bwx1(i);
```

```
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          z1=[Tx(2) bwx1(i)];
          z2=[Rx(2) bwx1(i)];
          d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
          h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
          h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
          r=sqrt((h1+h2)^2+d^2);
          D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
          Omaga=(2*pi*(r-di)*10/Lamda);
          ep=5-j*60*0.005*Lamda;
          Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
          %========== find transmission in reflection ==========
          [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
          [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
          bEr=bEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
        else
          Rf=1;
          Tr1=0;
          Tr2=0;
        end
      end
    elseif abs(Rx(1)-Tx(1))<0.1 & abs(bwx2(i)-bwx1(i))<0.1          %infi and infi
      Rf=1;
      Tr1=0;
      Tr2=0;
    elseif abs(Rx(2)-Tx(2))<0.1 & abs(bwy2(i)-bwy1(i))<0.1          %zero and infi
      Rf=1;
      Tr1=0;
      Tr2=0;
    else
      m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
      m2=(bwy2(i)-bwy1(i))/(bwx2(i)-bwx1(i));        %slope of wall
      c1=Tx(2)-(m1*Tx(1));
      c2=bwy1(i)-(m2*bwx1(i));
      if  abs(m1-m2)<0.1
        Rf=1;
        Tr1=0;
        Tr2=0;
      else
        if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
        [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
        Z(1)= eval(yy);
        Z(2)= eval(xx);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          if abs(m2)<0.1
            z1=[bwy1(i) Tx(1)];
            z2=[bwy1(i) Rx(1)];
            d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
            h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
            h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
            r=sqrt((h1+h2)^2+d^2);
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
```

```
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
          [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
          [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
          bEr=bEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
        else
          m3=(-1)/m2 ;                    %slope line tanchak
          c3=Tx(2)-m3*Tx(1);              %line1 equation constant from Tx
          c4=Rx(2)-m3*Rx(1);              %line1 equation constant from Rx
          A1=[1 -(m2);1 -(m3)];
          A2=[1 -(m2);1 -(m3)];
          b1=[c2;c3];
          b2=[c2;c4];
          z1=inv(A1)*b1;
          z2=inv(A2)*b2;
          d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
          h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
          h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
          r=sqrt((h1+h2)^2+d^2);
          D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
          Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
          x=(h1*d)/(h1+h2);               %distance from z1 to reflection point
          Xa=z1(2)+(x/d)*(z2(2)-z1(2));
          Ya=z1(1)+(x/d)*(z2(1)-z1(1));
          A=[Xa Ya];
          [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,A);
          [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,A,Rx) ;
          bEr=bEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
        end

      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    end
  end
end
%---------------- green --------------------
for i=2:length(gwx1)
  if abs(Rx(1)-Tx(1))<0.1 & (gwx2(i)~=gwx1(i))        %m1 = infinity
    m2=(gwy2(i)-gwy1(i))/(gwx2(i)-gwx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[gwy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))     %check
        z1=[gwy1(i) Tx(1)];
        z2=[gwy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
```

```
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================== find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        gEr=gEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    else
      c2=gwy1(i)-(m2*gwx1(i));
      Z(1)=m2*Rx(1)+c2;
      Z(2)=Rx(1);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[gwy1(i) Tx(1)];
        z2=[gwy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================== find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        gEr=gEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    end
  elseif (Rx(1)~=Tx(1)) & abs(gwx2(i)-gwx1(i))<0.1          %m2 = infinity
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
    if abs(m1)<0.1
      Z=[Rx(2) gwx1(i)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[Tx(2) gwx1(i)];
        z2=[Rx(2) gwx2(i)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
```

```
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        gEr=gEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    else
      c1=Tx(2)-(m1*Tx(1));
      Z(1)=m1*gwx2(i)+c1;
      Z(2)=gwx1(i);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[Tx(2) gwx1(i)];
        z2=[Rx(2) gwx1(i)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        gEr=gEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    end
  elseif abs(Rx(1)-Tx(1))<0.1 & abs(gwx2(i)-gwx1(i))<0.1          %infi and infi
    Rf=1;
    Tr1=0;
    Tr2=0;
  elseif abs(Rx(2)-Tx(2))<0.1 & abs(gwy2(i)-gwy1(i))<0.1          %zero and infi
    Rf=1;
    Tr1=0;
    Tr2=0;
  else
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
    m2=(gwy2(i)-gwy1(i))/(gwx2(i)-gwx1(i));      %slope of wall
    c1=Tx(2)-(m1*Tx(1));
    c2=gwy1(i)-(m2*gwx1(i));
    if abs(m1-m2)<0.1
```

```
        Rf=1;
        Tr1=0;
        Tr2=0;
      else
        if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
        [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
        Z(1)= eval(yy);
        Z(2)= eval(xx);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          if abs(m2)<0.1
            z1=[gwy1(i) Tx(1)];
            z2=[gwy1(i) Rx(1)];
            d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
            h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
            h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
            r=sqrt((h1+h2)^2+d^2);
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
            ep=5-j*60*0.005*Lamda;
            Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
            %========== find transmission in reflection ==========
            [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
            [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
            gEr=gEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
          else
            m3=(-1)/m2 ;                    %slope line tanchak
            c3=Tx(2)-m3*Tx(1);              %line1 equation constant from Tx
            c4=Rx(2)-m3*Rx(1);              %line1 equation constant from Rx
            A1=[1 -(m2);1 -(m3)];
            A2=[1 -(m2);1 -(m3)];
            b1=[c2;c3];
            b2=[c2;c4];
            z1=inv(A1)*b1;
            z2=inv(A2)*b2;
            d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
            h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
            h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
            r=sqrt((h1+h2)^2+d^2);
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
            ep=5-j*60*0.005*Lamda;
            Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
            %========== find transmission in reflection ==========
            x=(h1*d)/(h1+h2);               %distance from z1 to reflection point
            Xa=z1(2)+(x/d)*(z2(2)-z1(2));
            Ya=z1(1)+(x/d)*(z2(1)-z1(1));
            A=[Xa Ya];
            [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,A);
            [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,A,Rx) ;
            gEr=gEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
          end
```

```
        else
          Rf=1;
          Tr1=0;
          Tr2=0;
        end

      end
    end
end
%------------------ red ---------------------------
for i=2:length(rwx1)
  if abs(Rx(1)-Tx(1))<0.1 & (rwx2(i)~=rwx1(i))          %m1 = infinity
    m2=(rwy2(i)-rwy1(i))/(rwx2(i)-rwx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[rwy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[rwy1(i) Tx(1)];
        z2=[rwy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================== find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        rEr=rEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    else
      c2=rwy1(i)-(m2*rwx1(i));
      Z(1)=m2*Rx(1)+c2;
      Z(2)=Rx(1);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[rwy1(i) Tx(1)];
        z2=[rwy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================== find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
```

```
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        rEr=rEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    end
  elseif (Rx(1)~=Tx(1)) & abs(rwx2(i)-rwx1(i))<0.1          %m2 = infinity
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
    if abs(m1)<0.1
      Z=[Rx(2) rwx1(i)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[Tx(2) rwx1(i)];
        z2=[Rx(2) rwx2(i)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        rEr=rEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    else
      c1=Tx(2)-(m1*Tx(1));
      Z(1)=m1*rwx2(i)+c1;
      Z(2)=rwx1(i);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[Tx(2) rwx1(i)];
        z2=[Rx(2) rwx1(i)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        rEr=rEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
```

```
        else
          Rf=1;
          Tr1=0;
          Tr2=0;
        end
      end
    elseif abs(Rx(1)-Tx(1))<0.1 & abs(rwx2(i)-rwx1(i))<0.1          %infi and infi
      Rf=1;
      Tr1=0;
      Tr2=0;
    elseif abs(Rx(2)-Tx(2))<0.1 & abs(rwy2(i)-rwy1(i))<0.1          %zero and infi
      Rf=1;
      Tr1=0;
      Tr2=0;
    else
      m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
      m2=(rwy2(i)-rwy1(i))/(rwx2(i)-rwx1(i));       %slope of wall
      c1=Tx(2)-(m1*Tx(1));
      c2=rwy1(i)-(m2*rwx1(i));
      if abs(m1-m2)<0.1
        Rf=1;
        Tr1=0;
        Tr2=0;
      else
        if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
        [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
        Z(1)= eval(yy);
        Z(2)= eval(xx);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          if abs(m2)<0.1
            z1=[rwy1(i) Tx(1)];
            z2=[rwy1(i) Rx(1)];
            d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
            h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
            h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
            r=sqrt((h1+h2)^2+d^2);
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
          ep=5-j*60*0.005*Lamda;
          Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
          %================ find transmission in reflection ==========
            [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
            [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
            rEr=rEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
          else
            m3=(-1)/m2 ;                          %slope line tanchak
            c3=Tx(2)-m3*Tx(1);                    %line1 equation constant from Tx
            c4=Rx(2)-m3*Rx(1);                    %line1 equation constant from Rx
            A1=[1 -(m2);1 -(m3)];
            A2=[1 -(m2);1 -(m3)];
            b1=[c2;c3];
            b2=[c2;c4];
            z1=inv(A1)*b1;
```

```
            z2=inv(A2)*b2;
            d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
            h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
            h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
            r=sqrt((h1+h2)^2+d^2);
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
          ep=5-j*60*0.005*Lamda;
          Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
          %================ find transmission in reflection ==========
            x=(h1*d)/(h1+h2);                  %distance from z1 to reflection point
            Xa=z1(2)+(x/d)*(z2(2)-z1(2));
            Ya=z1(1)+(x/d)*(z2(1)-z1(1));
            A=[Xa Ya];
            [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,A);
            [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,A,Rx) ;
            rEr=rEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
          end
        else
          Rf=1;
          Tr1=0;
          Tr2=0;
        end
      end
    end
end
%------------------- yellow --------------------
for i=2:length(ywx1)
  if abs(Rx(1)-Tx(1))<0.1 & (ywx2(i)~=ywx1(i))          %m1 = infinity
    m2=(ywy2(i)-ywy1(i))/(ywx2(i)-ywx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[ywy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[ywy1(i) Tx(1)];
        z2=[ywy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        yEr=yEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
```

```
    else
      c2=ywy1(i)-(m2*ywx1(i));
      Z(1)=m2*Rx(1)+c2;
      Z(2)=Rx(1);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[ywy1(i) Tx(1)];
        z2=[ywy1(i) Rx(1)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        yEr=yEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    end
  elseif (Rx(1)~=Tx(1)) & abs(ywx2(i)-ywx1(i))<0.1          %m2 = infinity
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
    if abs(m1)<0.1
      Z=[Rx(2) ywx1(i)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        z1=[Tx(2) ywx1(i)];
        z2=[Rx(2) ywx2(i)];
        d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
        h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
        h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
        r=sqrt((h1+h2)^2+d^2);
        D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);      %D=reflec-direct wave
        Omaga=(2*pi*(r-di)*10/Lamda);
        ep=5-j*60*0.005*Lamda;
        Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
        %================ find transmission in reflection ==========
        [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
        [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
        yEr=yEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
      else
        Rf=1;
        Tr1=0;
        Tr2=0;
      end
    else
      c1=Tx(2)-(m1*Tx(1));
      Z(1)=m1*ywx2(i)+c1;
```

```
        Z(2)=ywx1(i);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
           z1=[Tx(2) ywx1(i)];
           z2=[Rx(2) ywx1(i)];
           d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
           h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
           h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
           r=sqrt((h1+h2)^2+d^2);
           D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
           Omaga=(2*pi*(r-di)*10/Lamda);
           ep=5-j*60*0.005*Lamda;
           Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
           %================ find transmission in reflection ==========
           [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
           [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
           yEr=yEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
        else
           Rf=1;
           Tr1=0;
           Tr2=0;
        end
     end
  elseif abs(Rx(1)-Tx(1))<0.1 & abs(ywx2(i)-ywx1(i))<0.1          %infi and infi
     Rf=1;
     Tr1=0;
     Tr2=0;
  elseif abs(Rx(2)-Tx(2))<0.1 & abs(ywy2(i)-ywy1(i))<0.1          %zero and infi
     Rf=1;
     Tr1=0;
     Tr2=0;
  else
     m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
     m2=(ywy2(i)-ywy1(i))/(ywx2(i)-ywx1(i));        %slope of wall
     c1=Tx(2)-(m1*Tx(1));
     c2=ywy1(i)-(m2*ywx1(i));
     if  abs(m1-m2)<0.1
        Rf=1;
        Tr1=0;
        Tr2=0;
     else
        if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
        [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
        Z(1)= eval(yy);
        Z(2)= eval(xx);

        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
           if abs(m2)<0.1
              z1=[ywy1(i) Tx(1)];
              z2=[ywy1(i) Rx(1)];
              d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
              h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
              h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
              r=sqrt((h1+h2)^2+d^2);
```

```
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
          ep=5-j*60*0.005*Lamda;
          Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
          %================== find transmission in reflection ==========
            [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,Z);
            [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Z,Rx) ;
            yEr=yEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
          else
            m3=(-1)/m2 ;                          %slope line tanchak
            c3=Tx(2)-m3*Tx(1);                    %line1 equation constant from Tx
            c4=Rx(2)-m3*Rx(1);                    %line1 equation constant from Rx
            A1=[1 -(m2);1 -(m3)];
            A2=[1 -(m2);1 -(m3)];
            b1=[c2;c3];
            b2=[c2;c4];
            z1=inv(A1)*b1;
            z2=inv(A2)*b2;
            d=sqrt((z2(1)-z1(1))^2+(z2(2)-z1(2))^2);
            h1=sqrt((Tx(1)-z1(2))^2+(Tx(2)-z1(1))^2);
            h2=sqrt((Rx(1)-z2(2))^2+(Rx(2)-z2(1))^2);
            r=sqrt((h1+h2)^2+d^2);
            D=sqrt((h1+h2)^2+d^2)-sqrt((h1+h2)^2-d^2);        %D=reflec-direct wave
            Omaga=(2*pi*(r-di)*10/Lamda);
          ep=5-j*60*0.005*Lamda;
          Rf=(cos(Omaga)-(ep-(sin(Omaga))^2)^(1/2))/(cos(Omaga)+(ep-(sin(Omaga))^2)^(1/2));
          %================== find transmission in reflection ==========
            x=(h1*d)/(h1+h2);                     %distance from z1 to reflection point
            Xa=z1(2)+(x/d)*(z2(2)-z1(2));
            Ya=z1(1)+(x/d)*(z2(1)-z1(1));
            A=[Xa Ya];
            [Et1
Tr1]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,Tx,A);
            [Et2
Tr2]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,ywy2,A,Rx) ;
            yEr=yEr+(E0*(Lamda/(10*4*pi*D))*exp(-j*10*2*pi*D/Lamda)*Rf*Tr1*Tr2);
          end
        else
          Rf=1;
          Tr1=0;
          Tr2=0;
        end
      end
    end
  end
%================================ total
Er=bEr+yEr+rEr+gEr;
```

Tran

```
function [Et Tcoef]=tran(bwx1,bwx2,bwy1,bwy2,gwx1,gwx2,gwy1,gwy2,rwx1,rwx2,rwy1,rwy2,ywx1,ywx2,ywy1,
ywy2,Tx,Rx)
Lamda=(3*10^8)/(2.45*10^9);
E0=1;
Bule=1;
Green=1;
Red=1;
Yellow=1;
%---------------------------------- Blue ----------------------------------------------------------
for i=2:length(bwx1)
  bTr=1;
  if abs(Rx(1)-Tx(1))<0.1 & (bwx2(i)~=bwx1(i))          %infi and x
    m2=(bwy2(i)-bwy1(i))/(bwx2(i)-bwx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[bwy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        bTr=0.5;
      end
    else
      c2=bwy1(i)-(m2*bwx1(i));
      Z(1)=m2*Rx(1)+c2;
      Z(2)=Rx(1);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        bTr=0.5;
      end
    end

  elseif (Rx(1)~=Tx(1)) & abs(bwx2(i)-bwx1(i))<0.1          %m2 = infinity
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
    if abs(m1)<0.1
      Z=[Rx(2) bwx1(i)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        bTr=0.5;
      end
    else
      c1=Tx(2)-(m1*Tx(1));
      Z(1)=m1*bwx2(i)+c1;
      Z(2)=bwx1(i);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        bTr=0.5;
      end
    end
  elseif abs(Rx(1)-Tx(1))<0.1 & abs(bwx2(i)-bwx1(i))<0.1          %infi and infi
    bTr=1;
  elseif abs(Rx(2)-Tx(2))<0.1 & abs(bwy2(i)-bwy1(i))<0.1          %zero and infi
    bTr=1;
  else
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
    m2=(bwy2(i)-bwy1(i))/(bwx2(i)-bwx1(i));        %slope of wall
    c1=Tx(2)-(m1*Tx(1));
```

```
    c2=bwy1(i)-(m2*bwx1(i));
    if abs(m1-m2)<0.1
      bTr=1;
    else
      if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
      [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
%       ['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')]
%       ['yy=' num2str(m2, '%5.3f') '*xx' sc2 num2str(c2, '%5.3f')]
      Z(1)= eval(yy);
      Z(2)= eval(xx);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        bTr=0.5;
      end
    end
  end
  Bule=Bule*bTr;
end

%================================ Green ================================
for i=2:length(gwx1)
  gTr=1;
  if abs(Rx(1)-Tx(1))<0.1 & (gwx2(i)~=gwx1(i))          %infi and x
    m2=(gwy2(i)-gwy1(i))/(gwx2(i)-gwx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[gwy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        gTr=0.63;
      end
    else
      c2=gwy1(i)-(m2*gwx1(i));
      Z(1)=m2*Rx(1)+c2;
      Z(2)=Rx(1);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        gTr=0.63;
      end
    end

  elseif (Rx(1)~=Tx(1)) & abs(gwx2(i)-gwx1(i))<0.1          %m2 = infinity
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
    if abs(m1)<0.1
      Z=[Rx(2) gwx1(i)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        gTr=0.63;
      end
    else
      c1=Tx(2)-(m1*Tx(1));
      Z(1)=m1*gwx2(i)+c1;
      Z(2)=gwx1(i);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        gTr=0.63;
      end
    end
  elseif abs(Rx(1)-Tx(1))<0.1 & abs(gwx2(i)-gwx1(i))<0.1          %infi and infi
    gTr=1;
  elseif abs(Rx(2)-Tx(2))<0.1 & abs(gwy2(i)-gwy1(i))<0.1          %zero and infi
```

```
      gTr=1;
    else
      m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %slope of distance Tx to Rx
      m2=(gwy2(i)-gwy1(i))/(gwx2(i)-gwx1(i));        %slope of wall
      c1=Tx(2)-(m1*Tx(1));
      c2=gwy1(i)-(m2*gwx1(i));
      if  abs(m1-m2)<0.1
        gTr=1;
      else
        if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
        [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
        Z(1)= eval(yy);
        Z(2)= eval(xx);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          gTr=0.63;
        end
      end
    end
    Green=Green*gTr;
end

%================================ Red ================================================
for i=2:length(rwx1)
    rTr=1;
    if abs(Rx(1)-Tx(1))<0.1 & (rwx2(i)~=rwx1(i))              %infi and x
      m2=(rwy2(i)-rwy1(i))/(rwx2(i)-rwx1(i));          %slope of wall
      if abs(m2)<0.1
        Z=[rwy1(i) Rx(1)];
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          rTr=0.63;
        end
      else
        c2=rwy1(i)-(m2*rwx1(i));
        Z(1)=m2*Rx(1)+c2;
        Z(2)=Rx(1);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          rTr=0.63;
        end
      end

    elseif (Rx(1)~=Tx(1)) & abs(rwx2(i)-rwx1(i))<0.1          %m2 = infinity
      m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
      if abs(m1)<0.1
        Z=[Rx(2) rwx1(i)];
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          rTr=0.63;
        end
      else
        c1=Tx(2)-(m1*Tx(1));
        Z(1)=m1*rwx2(i)+c1;
        Z(2)=rwx1(i);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          rTr=0.63;
        end
      end
```

```
  elseif abs(Rx(1)-Tx(1))<0.1 & abs(rwx2(i)-rwx1(i))<0.1          %infi and infi
    rTr=1;
  elseif abs(Rx(2)-Tx(2))<0.1 & abs(rwy2(i)-rwy1(i))<0.1          %zero and infi
    rTr=1;
  else
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
    m2=(rwy2(i)-rwy1(i))/(rwx2(i)-rwx1(i));       %slope of wall
    c1=Tx(2)-(m1*Tx(1));
    c2=rwy1(i)-(m2*rwx1(i));
    if  abs(m1-m2)<0.1
      rTr=1;
    else
      if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
      [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
      Z(1)= eval(yy);
      Z(2)= eval(xx);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        rTr=0.63;
      end
    end
  end
  Red=Red*rTr;
end
%================================ Yellow ================================
for i=2:length(ywx1)
  yTr=1;
  if abs(Rx(1)-Tx(1))<0.1 & (ywx2(i)~=ywx1(i))            %infi and x
    m2=(ywy2(i)-ywy1(i))/(ywx2(i)-ywx1(i));          %slope of wall
    if abs(m2)<0.1
      Z=[ywy1(i) Rx(1)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        yTr=0.63;
      end
    else
      c2=ywy1(i)-(m2*ywx1(i));
      Z(1)=m2*Rx(1)+c2;
      Z(2)=Rx(1);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        yTr=0.63;
      end
    end

  elseif (Rx(1)~=Tx(1)) & abs(ywx2(i)-ywx1(i))<0.1          %m2 = infinity
    m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));                %
    if abs(m1)<0.1
      Z=[Rx(2) ywx1(i)];
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        yTr=0.63;
      end
    else
      c1=Tx(2)-(m1*Tx(1));
      Z(1)=m1*ywx2(i)+c1;
      Z(2)=ywx1(i);
      if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
        yTr=0.63;
```

```
        end
      end
    elseif abs(Rx(1)-Tx(1))<0.1 & abs(ywx2(i)-ywx1(i))<0.1          %infi and infi
      yTr=1;
    elseif abs(Rx(2)-Tx(2))<0.1 & abs(ywy2(i)-ywy1(i))<0.1          %zero and infi
      yTr=1;
    else
      m1=(Rx(2)-Tx(2))/(Rx(1)-Tx(1));              %slope of distance Tx to Rx
      m2=(ywy2(i)-ywy1(i))/(ywx2(i)-ywx1(i));        %slope of wall
      c1=Tx(2)-(m1*Tx(1));
      c2=ywy1(i)-(m2*ywx1(i));
      if  abs(m1-m2)<0.1
        yTr=1;
      else
        if c1<0; sc1='';else;sc1='+';end;if c2<0; sc2='';else;sc2='+';end;
        [xx yy]=solve(['yy=' num2str(m1, '%5.3f') '*xx' sc1 num2str(c1, '%5.3f')],['yy=' num2str(m2, '%5.3f') '*xx'
sc2 num2str(c2, '%5.3f')]);
        Z(1)= eval(yy);
        Z(2)= eval(xx);
        if (Tx(1)<=Z(2)<=Rx(1)) | (Rx(1)<=Z(2)<=Tx(1))      %check
          yTr=0.63;
        end
      end
    end
    Yellow=Yellow*yTr;
end
%=============================================
di=sqrt((Rx(1)-Tx(1))^2+(Rx(2)-Tx(2))^2);
Et=E0*(Lamda/(10*4*pi*di))*Bule*Green*Red*Yellow*exp(-j*2*pi*di*10/Lamda);
Tcoef=Bule*Green*Red*Yellow;
```

เอกสารอ้างอิง

[1] MIMO หัวใจสู่การสื่อสารไร้สายความเร็วสูง ,ผศ. ดร.บงการ หอมนาน,ภาควิชาวิศวกรรมโทรคมนาคม มหาวิทยาลัยธุรกิจบัณฑิตย์ URL: http://te.eng.dpu.ac.th E-mail: bongkarn@dpu.ac.th

[2] Downlink Capacity of Interference-Limited MIMO Systems With Joint Detection Huaiyu Dai, *Member, IEEE*, Andreas F. Molisch, *Senior Member, IEEE*, and H. Vincent Poor, *Fellow, IEEE*,IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, VOL. 3, NO. 2, MARCH 2004

[3] Katsuyoshi Sato, Takeshi Manabe, *Member, IEEE*, Toshio Ihara, *Member, IEEE*, Hiroshi Saito, Shigeru Ito,Tetsu Tanaka, *Member, IEEE*, Kazuyoshi Sugai, Norichika Ohmi, Yasushi Murakami, *Member, IEEE*,Masanori Shibayama, Yoshihiko Konishi, *Senior Member, IEEE*, and Tsuneto Kimura, Measurements of Reflection and Transmission Characteristics of Interior Structures of Office Building in the 60-GHz Band, IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, VOL. 45, NO. 12, DECEMBER 1997

[4] อ.ดร.พีระพงษ์ อุฑารสกุล, เอกสารประกอบการเรียนวิชา Mobile Communicatuion System 1/2550

[5] Sakda Naruniranat, Yi Huang, and David Parsons Department of Electrical Engineering and Electronics University of Liverpool, L69 3GJ, UK,A three-dimensional Image Ray Tracing (3D-IRT) Method for Indoor Wireless Channel Characterisation,1999 High Frequency Postgraduate Student Colloquium*I?* September *1999,* Universip *of* Lee*&*

[6] Jacques Beneat and Nathan Bailey Norwich University Northfield, Vermont,OPTIMIZATION OF BUILDING MATERIAL PROPERTIES FOR ACCURATE INDOOR RAY TRACING MODELS

[7] นายพีระศักดิ์ ค้าพลอย, โครงงานวิชา 427499 โครงงานวิศวกรรมโทรคมนาคม, โครงงานโปรแกรมคำนวณจุดติดตั้งสถานีฐานภายในอาคาร, มหาวิทยาลัยเทคโนโลยีสุรนารี, ปีการศึกษา 3/2544