รายการอ้างอิง

รัชทิน จันทร์เจริญ วัชระ เลิศพิริยสุวัฒน์ (2001). การควบคุมหุ่นยนต์อุตสาหกรรมแบบปรับเปลี่ยนได้โดยการประมาณค่าความเสียดทานในเวลาจริง, การประชุมวิชาการเครือข่ายเครื่องกลแห่งประเทศไทยครั้งที่ 15, พศ. 2544

รัชทิน จันทร์เจริญ วัชระ เลิศพิริยสุวัฒน์ สิรณัฐ ขันธจีรวัฒน์ ภิญโญ ศรีคำแหง(2002). การชดเชยแรงเสียดทานในเวลาจริงทางระบบกล, การประชุมวิชาการเครือข่ายเครื่องกลแห่งประเทศไทยครั้งที่ 16, พศ. 2545

J. Rice and A. Ruina(1983). Stability of steady frictional slipping, Journal of Applied Mechanics, Vol. 50 , June 1983, pp. 343-349

C. Canudas , K. Astrom and K. Braun(1986). Adaptive friction compensation in DC motor drives, Proceeding of the IEEE International Conference on Robotics and Automation, pp 1556-61, San Francisco, April, 1986, CA

Naomi Elizabeth Ehrich and P.S. Krishnaprasad(1991). An Investigation of Control Strategies for Friction Compensation, available from http://www.isr.umd.Edu/TechReports /ISR/1991/MS_91-4/MS_91-4.phtml

Brian Armstrong-Hélouvry and Pierre Dupont (1993). Friction Modeling for Control, Proceeding of the American Control Conference, pp. 1905-9, San Francisco, June 1993, California.

Brian Armstrong-Hélouvry (1993). Stick Slip and Control in Low-Speed Motion, IEEE Transactions on Automatic Control, Vol. 38, No.10, pp. 1483-96, October 1993.

C. Canudas de wit, H. Olsson, K. J. Astrom and P. Lischinsky (1995). A New Model for Control of System with Friction, IEEE Transactions on Automatic Control, Vol. 40, No.3, pp. 419-25, Mar 1995.

Pierre E.Dupont and Eric P Dunlap (1993). Friction Modeling and Control in Boundary Lubrication. Proceedingof the American Control Conferrence, pp.1911-1914, San Francisco, June 1993, California.

Pierre E.Dupont (1994). Avoid Sick-Slip Through PD Control. IEEE Transactions on Automatic Control, Vol. 39, No.5, May 1994. pp. 1094-1097.

Brian Armstrong-Helouvry, Pierre Dupont and Carlos Canudas De Wit(1994). A Servay of Models, Analysis Tools and Compensation Methods for the Control of Mechines with Friction, Automation, Vol. 30, No.7, pp.1083-1138

E. Dupont and Eric P. Dunlap (1995). Friction Modeling and PD Compensation at Very Low Velocities, ASME Transactions of Dynamic systems Measurement and Control , Vol. 117, March 1995, pp.8-14

Susan L. Ipri and Haruhiko Asada (1995). Tuned Dither for Friction Suppression During Force-Guided Robotic Assembly, IEEE conference on Robotic and Automation, pp. 310-5, 1995.

Laura R. Ray and Jennifer S Remine (1998). Machine Friction Estimation for Modeling Diagnostics and Control, Proceeding of the American Control Conference, pp.2731-2741, Philadelphia, June 1998, Pensylvania

Karl J. Astrom (1999). Control of System with Friction. available from : http://www.control.lth.se /articles/article.pike?artkey=ast98movic

J.L. Meriam &L.G. Kraige, Engineering Mechanics Vol.2,John Wiley and sons, 1987

Steven C. Chapra & Raymond P. Canale, Numerical methods for engineers: with software and programming applications, McGraw-Hill, 1990.

John J. Craig Introuction to Robotics mechanics and control, Addison-Wesley, 1955.

Lorenzo Sciavicco & Bruno Siciliano, Modeling and Control of Robot mainpulators, McGraw-Hill, 1996.

ภาคผนวก

ภาคผนวก ก.

โปรแกรม Matlab®- Simulink®, xPC target

ก.1 นำเรื่อง

โปรแกรม Matlab เป็นโปรแกรมคำนวณเชิงตัวเลขที่ใช้ในการสนับสนุนการวิจัยต่างๆ ทั้งงานวิศวกรรม และในงานสาขาอื่นๆ เนื่องโปรแกรม Matlab® เป็นโปรแกรมที่ง่ายต่อการศึกษา มีฟังค์ชั่นเฉพาะมากมาย และการทำงานบนระบบปฏิบัติการ Windows® ซึ่งเป็นส่วนหนึ่งที่สามารถทำให้โปรแกรม Matlab สามารถใช้ร่วมกับโปรแกรมอื่นเพื่อเพิ่มศักยภาพในการทำงาน

Simulink® Toolbox เป็นเครื่องมือหนึ่งในโปรแกรม Matlab ที่ใช้ในการจำลอง ทดสอบ และวิเคราะห์การทำงานของระบบพลศาสตร์ในเชิงเวลาได้อย่างง่ายดายภายใต้การทำงานในหน้าต่างการเชื่อมต่อแบบรูปภาพ คือการนำ Block Diagram ใน Library Simulink แต่ละอันมาต่อกันเพื่อแทนระบบพลศาสตร์ที่สนใจ สามารถใช้ได้ทั้งในระบบเชิงเส้นและไม่เชิงเส้น ระบบเวลาต่อเนื่องและไม่ต่อเนื่อง โดยในวิทยานิพนธ์ใช้โปรแกรมนี้ในการจำลองระบบต่างๆ รวมไปถึงใช้ในการทำตัวควบคุม และเก็บข้อมูลสำหรับการควบคุมหุ่นยนต์ CRS Robotics

xPC Target เป็นเครื่องมือหนึ่งในโปรแกรม Matlab ที่ใช้ในการสร้างต้นแบบ ควบคุม และทำการทดสอบระบบ ในเวลาจริง ที่ทำงานอยู่บน Target PC ใช้ระบบปฏิบัติการ xPC kernel โดยอุปกรณ์ติดต่อต่างๆ ที่จะต้องใช้เช่น AD, DA หรือการ์ดติดต่อต่างๆ จะต้องติดตั้งอยู่บนเครื่อง Target PC

ข้อดีที่สำคัญของระบบ xPC คือการแยกระหว่างเครื่องต้นแบบและเครื่องเป้าหมาย โดยเครื่องต้นแบบใช้งานบนระบบปฏิบัติการ Windows ทำให้สามารถนำความสามารถและส่วนสนับสนุนที่มีอยู่มากมายในระบบปฏิบัติการ Windows มาใช้ในการพัฒนาระบบได้ และเครื่องเป็นหมายที่ใช้ในการควบคุมระบบ ทำงานบนระบบปฏิบัติการของ xPC Real-time สามารถใช้ในการควบคุม และทดสอบงานที่ต้องการความแม่นยำสูงได้

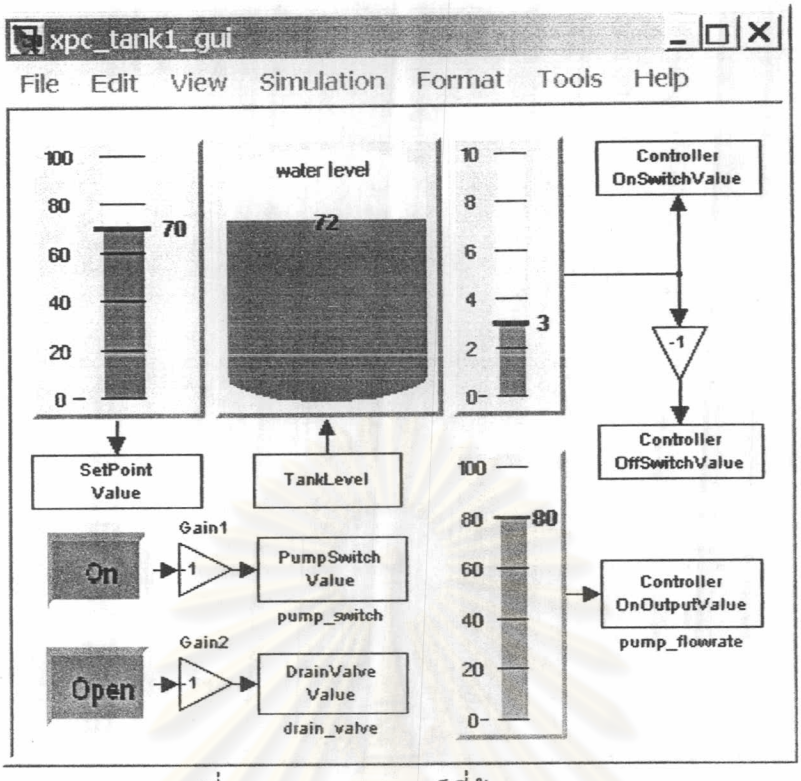
ระบบนี้ต้องการคอมพิวเตอร์ 2 เครื่องในการทำงาน ประกอบด้วย Host PC ที่มีโปรแกรม Matlab® Simulink® เพื่อใช้ในการสร้าง Block Diagram โดย Simulink Toolbox และเมื่อจำลองระบบด้วย Block Diagram แล้วระบบสามารถทดสอบได้บน Host PC เมื่อได้ระบบที่จำลองด้วย Block Diagram แล้ว Host PC จะทำการเปลี่ยนแปลงจาก Block Diagram เป็นภาษาที่สามารถใช้ใน Target PC ได้โดยใช้โปรแกรม real-time Workshop® และ C/C++

ในการสร้างโค้ดใหม่ และทำการส่งผ่านข้อมูลจากเครื่อง Host PC ไปสู่ Target PC ดังแสดงตัวอย่างระบบควบคุมระดับน้ำในรูปที่ ก.1 ก.2 ก.3

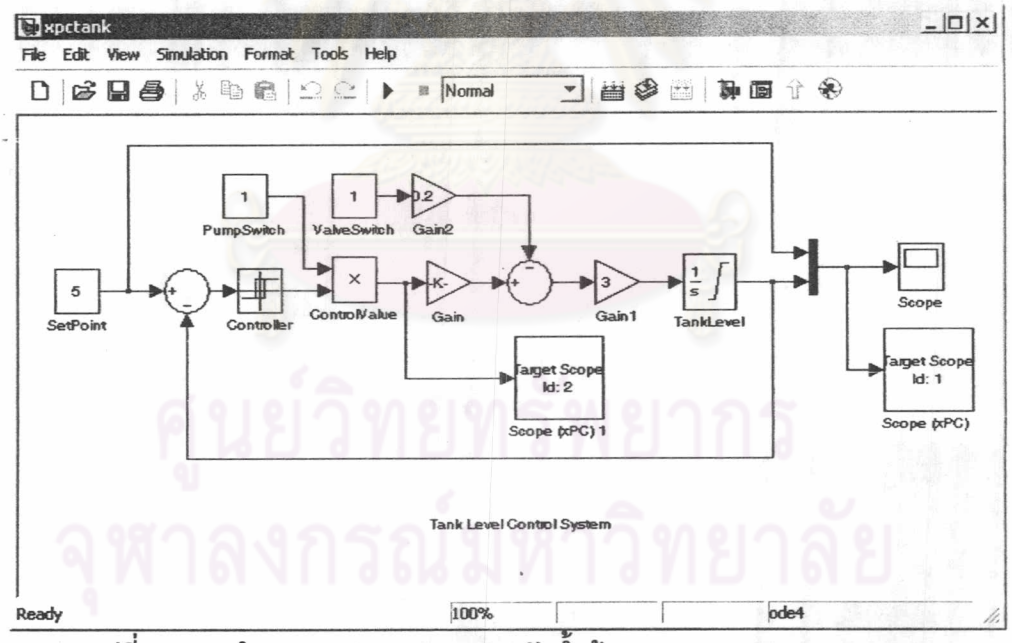
ก.2 ความรู้เบื้องต้นก่อนการใช้งาน

ก่อนการใช้งานของระบบ xPC เครื่อง Host PC จะต้องติดตั้งระบบปฏิบัติการ Window และโปรแกรมต่างๆ ดังนี้

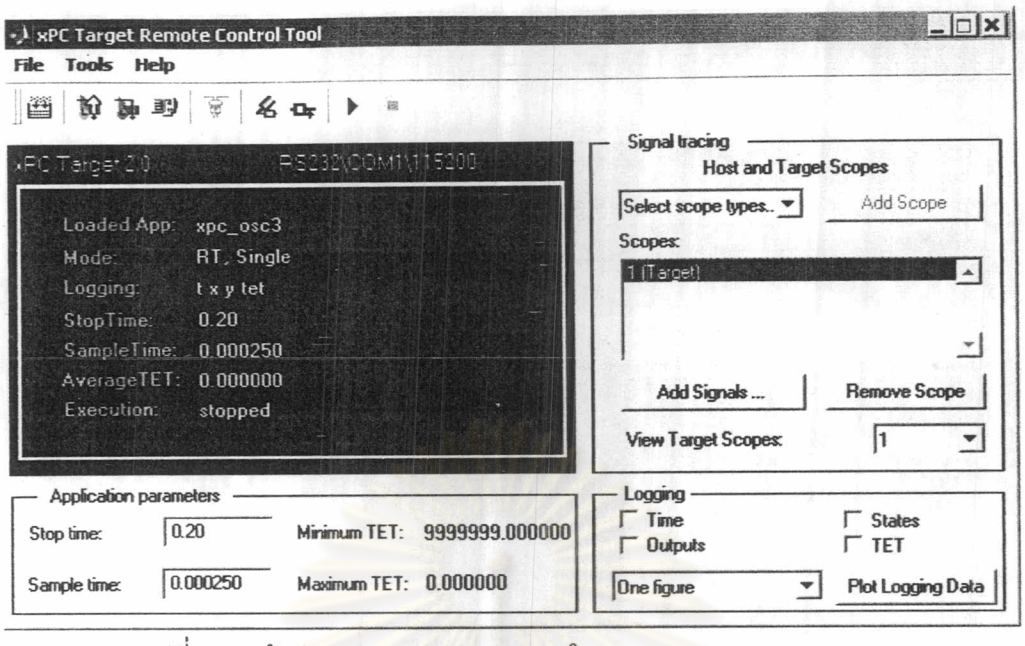
- Matlab® ใช้ในการควบคุมและติดต่อกับระบบ xPC โดยผ่านบรรทัดคำสั่ง หรือหน้าต่างควบคุม สามารถใช้ในการ บันทึกข้อมูลจาก Target PC สั่งเริ่มและหยุดการทำงานของ Target เปลี่ยนแปลงค่าตัวแปรที่ใช้ รวมไปถึงการรวบรวมและวิเคราะห์ข้อมูลที่ได้จาก Target PC ดังรูปที่ ก.3
- Simulink Library ใช้ในการสร้าง Block Diagram เพื่อควบคุมและจำลองระบบพลศาสตร์ที่สนใจดังที่กล่าวมาแล้วข้างต้น (รูปที่ ก.1 และ ก.2) และสามารถสร้าง Block ที่มีคุณลักษณะเฉพาะตามที่ต้องการ เพิ่มเติมจากที่ Simulink Library มีอยู่ โดยการใช้ C-Code S-Function เพื่อขยายความสามารถของโปรแกรมออกไป และอีกหนึ่งคุณลักษณะที่น่าสนใจใน Simulink Library คือ IO Diver Block Library ซึ่งเป็นการจัดเตรียม Diver สำหรับการติดต่อ IO ชนิดต่างๆ ที่นิยมใช้มากกว่า 400 ชนิดดังแสดงในรูปที่ ก.4 เป็นต้น
- Real-Time Workshop ใช้ในการเปลี่ยนรูปแบบข้อมูลจาก Block Diagram ไปสู่รหัสภาษา C ด้วยคำสั่ง Build ดังแสดงในรูปที่ ก.5
- C Compiler ใช้ในการสร้าง code เพื่อใช้ในการปฏิบัติการของ xPC kernel ในการใช้งานต้องติดตั้งที่ Host PC โดยโปรแกรมที่สามารถใช้ได้คือ Microsoft Visual C++ version 5 6 หรือ 7 Watcom C/C++ version 1.6 หรือ 11



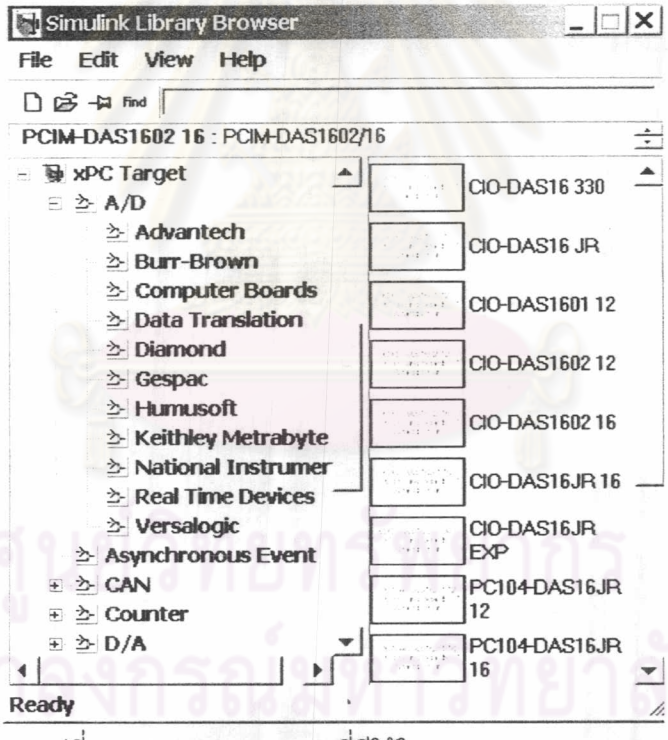
รูปที่ ก.1 ระบบพลศาสตร์ ที่ต้องการควบคุม



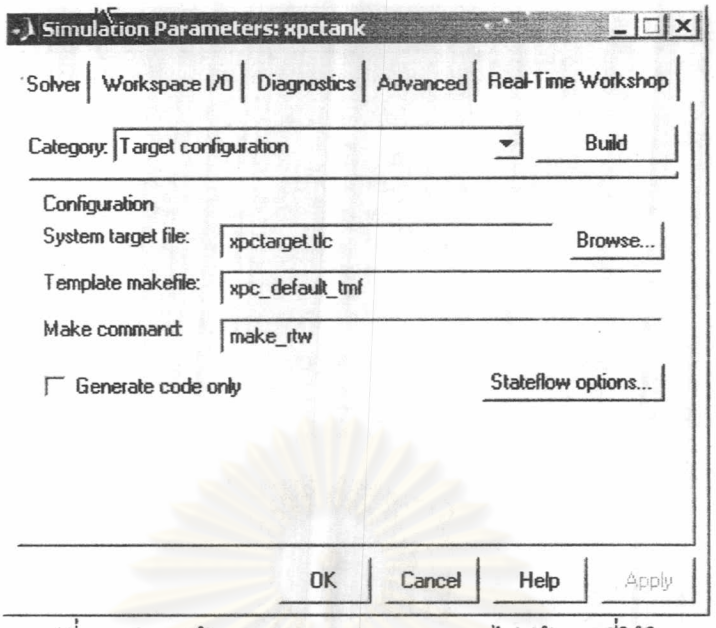
รูปที่ ก.2 การจำลองระบบการควบคุมระดับน้ำด้วย Simulink Block Diagram



รูปที่ ก.3 หน้าต่างแสดงการควบคุมการทำงานของ xPC บน Host PC



รูปที่ ก.4 IO Block Library ที่มีใช้ใน Simulink Library



รูปที่ ก.5 การเปลี่ยนรูปแบบข้อมูลจาก Block Diagram ไปสู่ข้อมูลที่ใช้ใน xPC Target

ก.3 สมบัติหลักที่ของ xPC

3.1 สามารถใช้งานโปรแกรมหรือระบบที่จำลองด้วย Simulink® บน xPC kernel ซึ่งเป็นระบบปฏิบัติการที่ทำงานบนเวลาจริงในเครื่อง PC

3.2 สนับสนุนการทำงานบน PC, PC/104, Compact PCI, industrial PC หรือ Single board ให้สามารถใช้งานในระบบเวลาจริง

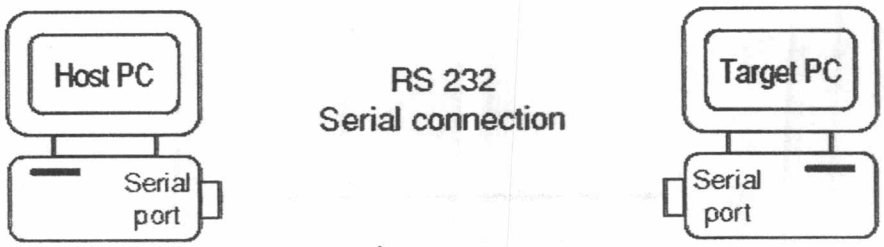
3.3 สามารถทำงานที่ความถี่ Sampling Rate สูงถึง 100 kHz ได้ โดยขึ้นอยู่กับประสิทธิภาพของตัวประมวลผล (Micro Processor)

3.4 สนับสนุนการใช้ IO board แบบมาตรฐานมากกว่า 150 ชนิด ซึ่งจัดเตรียมไว้ใน IO Device driver library ดังแสดงตัวอย่างในรูปที่ ก.4

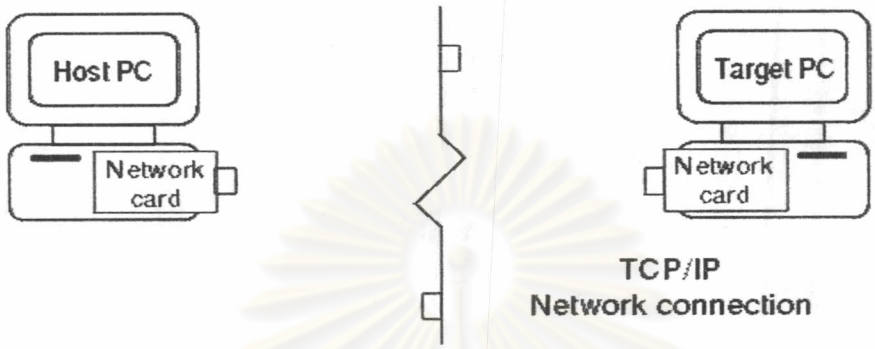
3.5 สามารถประมวลผลข้อมูล เปลี่ยนแปลงค่าตัวแปรต่างๆ และแสดงผลของระบบได้จากทั้งที่ Host และ Target PC

3.6 สามารถเชื่อมต่อระหว่าง Host และ Target ผ่านทาง RS232 หรือ TCP/IP protocol ดังรูปที่ ก.6

3.7 สนับสนุนการพัฒนา GUI (graphic user interface) เพื่อเข้าสู่การปรับปรุงสัญญาณและค่าตัวแปรของระบบ



ก.การเชื่อมต่อผ่านระบบ RS 232



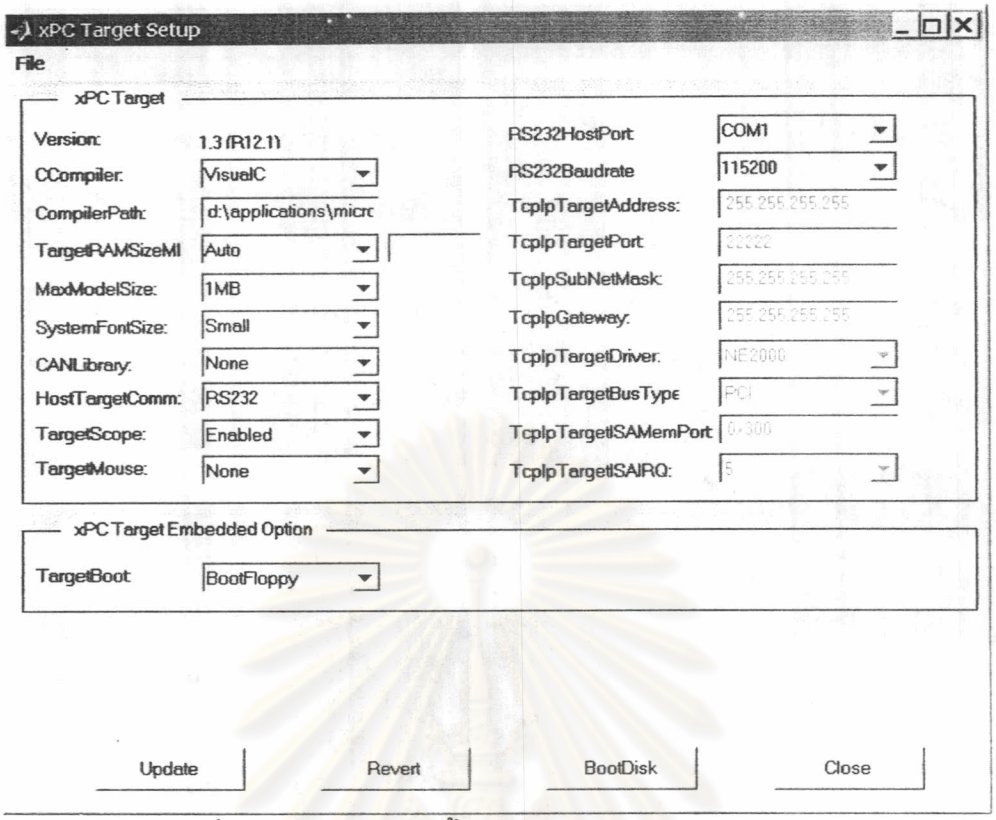
ข.การเชื่อมต่อผ่าน TCP/IP Protocol

รูปที่ ก.6 การเชื่อมต่อระหว่าง Host PC และ Target PC

ก.4 การใช้งาน xPC target

เมื่อพัฒนาหรือจำลองระบบบน Host PC แล้ว จะสามารถสร้างโปรแกรมที่จะทำงานบนเวลาจริงเพื่อนำไปใช้ใน Target PC โดยเริ่มจากการบูท Target PC ด้วยแผ่นพิเศษที่สร้างขึ้นมา (รูปที่ ก.7) เพื่อติดตั้งข้อมูล real time Kernel สู่ระบบบน Target PC หลังจากบูทเครื่อง Target PC แล้วจึงจะสามารถสร้างหรือถ่ายโอนข้อมูลที่พัฒนาไว้สู่ Target PCได้

xPC Target real-time kernel เป็นระบบปฏิบัติการที่ทำงานในเวลาจริง ซึ่งสามารถควบคุมจากเครื่อง Host PC ทั้งบน Matlab Command line หรือ Standard toolbox ของ Matlab หรือจะควบคุมผ่านทาง standard Internet browser และ the target PC command-line interface ในขณะที่ Target PC กำลังทำงาน สามารถเปลี่ยนแปลงค่าตัวแปรต่างๆ ของระบบ สั่งให้แสดงผลการวัดสัญญาณ ณ ขณะนั้น เก็บค่าที่แสดงอยู่เพื่อนำไปวิเคราะห์ต่อไป และสามารถแสดงสถานะของการทำงานบนจอของ Target PC ได้



รูปที่ ก.7 หน้าต่างการติดตั้งค่าตัวแปรในการใช้ xPC Target

รูปที่ ก.8 หน้าจอแสดงผลของ Target PC

การทำงานบน Target PC จะไม่มีผลใดๆ ต่อโปรแกรมต่างๆ ที่ติดตั้งไว้บนเครื่อง Target PC ดังนั้นเมื่อต้องการให้ Target PC สามารถใช้งานอย่างปรกติบนระบบปฏิบัติการ Window, Linux หรือระบบปฏิบัติการอื่นๆ สามารถทำได้เพียงแต่นำแผ่นบูทที่มีระบบปฏิบัติการ xPC Target real-time kernel ออกแล้วทำการบูทอย่างปรกติเท่านั้น

ก.5 การเชื่อมต่อระหว่าง Host และ Target PC

ดังที่ได้กล่าวมาแล้วในเบื้องต้นว่าการเชื่อมต่อระหว่าง Host และ Target PC สามารถใช้ได้ทั้ง ระบบ RS232 และ TCP/IP Protocol ซึ่งการเชื่อมต่อผ่านระบบ RS 232 โดยใช้ Port COM1 หรือ Port COM2 ในการเชื่อมต่อโดยวิธีนี้จะใช้ค่อนข้างง่ายเนื่องจากใช้เพียงอุปกรณ์พื้นฐานที่มีอยู่กับเครื่องคอมพิวเตอร์ และสนับสนุนการส่งผ่านข้อมูลด้วยความเร็วสูงสุด 115 kBaud

สำหรับการเชื่อมต่อผ่านระบบ TCP/IP Protocol นั้นจะค่อนข้างยุ่งยากกว่าเนื่องจากต้องใช้ Ether Card ในรุ่นที่ Matlab® สนับสนุน แต่สามารถส่งข้อมูลจากได้ในระยะทางที่ไกลกว่า และสามารถรับและส่งข้อมูลด้วยความเร็วสูงถึง 100 Mbit/s

ก.6 โปรแกรมที่ใช้ในการควบคุมระบบหุ่นยนต์ CRS

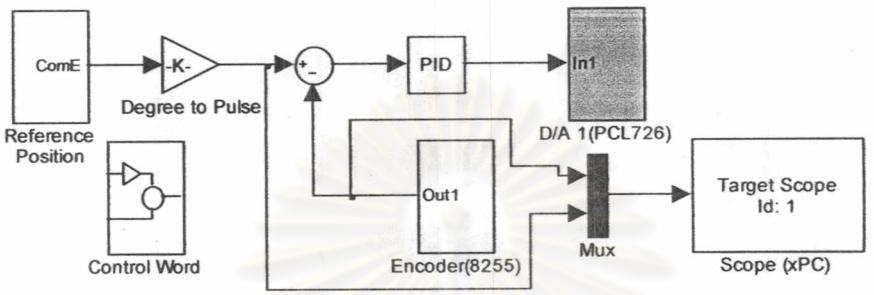
จากเหตุผลและสิ่งที่ได้กล่าวมาแล้วในเบื้องต้น ในงานวิจัยนี้จึงเลือกที่จะใช้ระบบ xPC-target เป็นโปรแกรมที่ใช้ในการพัฒนาและจำลองระบบควบคุม หุ่นยนต์ CRS โดยจะนำเสนอต่อไปนี้

ก.6.1 โปรแกรมควบคุมแบบปิด

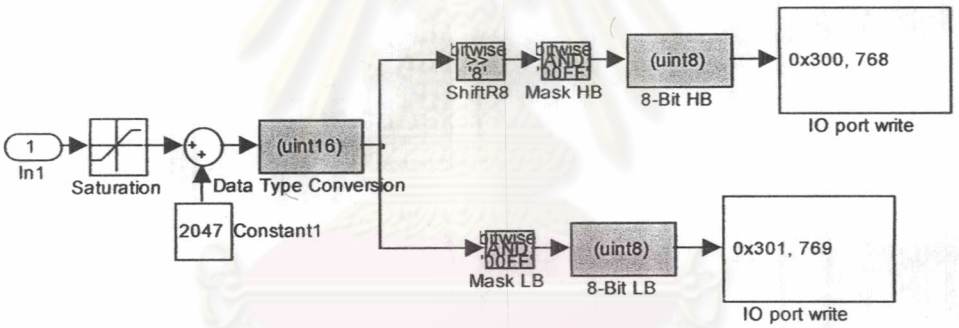
จากรูปที่ ก.9 เป็น block diagram ที่ใช้ในการควบคุมหุ่นยนต์แบบปิด ประกอบด้วย

- Reference Position Block เป็นสัญญาณอ้างอิง หรือค่าสัญญาณตำแหน่งที่ต้องการให้หุ่นยนต์เคลื่อนที่ไปถึง ในที่นี้จะใช้ค่าในหน่วยองศา
- Degree to Pulse เป็นส่วนที่เปลี่ยนค่าจากองศาเป็นจำนวน pulse เนื่องจากค่าที่วัดได้จากหุ่นยนต์เป็นค่า pulse และในระบบควบคุมแบบปิด ต้องนำค่าความผิดพลาดของสัญญาณตำแหน่งมาใช้ในการขับเคลื่อน ดังนั้นจึงต้องมีการเปลี่ยนแปลงสัญญาณให้อยู่ในหน่วยเดียวกัน
- ส่วนที่ใช้ในการควบคุม ซึ่งในที่นี้ใช้การควบคุมแบบ PD Controller

- D/A Subsystem (D/A 1 PCL726) เป็นส่วนที่ใช้ในการติดต่อกับการ์ด D/A รุ่น PCL 726 ของบริษัท Advantech ซึ่งรายละเอียดของการ์ดได้กล่าวไว้ในบทที่ 3 ในส่วนของการติดต่อกับการ์ด D/A เป็นส่วนที่ต้องพัฒนาขึ้นโดยอาศัยความรู้ความเข้าใจในการทำงานของการ์ดและฟังค์ชั่นของ Matalb® Simulink มีรายละเอียดดังแสดงไว้ในรูปที่ ก.10 คือ



รูปที่ ก.9 Block Diagram ที่ใช้ในการควบคุมหุ่นยนต์แบบปิด



รูปที่ ก.10 ส่วนควบคุมที่ใช้ในการติดต่อกับการ์ดA/D

a. Saturation Block ใช้ในการจำกัดค่าคำสั่งที่มาจากตัวควบคุมให้อยู่ในช่วง -2047 ถึง 2048 เนื่องจาก การ์ด D/A ที่นำมาใช้มีความละเอียดในการรับสัญญาณ 12 บิตในเลขฐาน 2 ซึ่งสามารถแปลงเป็นเลขฐาน 10 คือ 4096 โดยการ์ดจะส่งค่าสัญญาณ -5 โวลต์ (สามารถส่งสัญญาณได้สูงสุด ± 10 V.)เมื่อรับค่า 0 และส่งสัญญาณสูงสุด 10 โวลต์ เมื่อรับค่า 4096 จากรูปที่ ก.10 เป็นการ

จัดการกับขนาดของสัญญาณคำสั่งเพื่อให้สามารถรับค่า บวกและลบและจ่ายสัญญาณออกในรูปแบบที่สัมพันธ์กับค่าตัวเลขที่รับเข้า

b. Constant Block เป็นส่วนที่ใช้ในการจัดการกับขนาดของสัญญาณคำสั่งเพื่อให้มีความสอดคล้องกับ ค่าคำสั่งและสัญญาณที่จ่ายออกมาจากการ์ด

c. Data type Conversion Block ใช้ในการเปลี่ยนค่าที่รับจากโปรแกรมจากเลขฐาน 10 ไปเป็นเลขฐาน 2 ขนาด 16 บิตเพื่อให้สามารถนำไปใช้กับระบบส่งข้อมูลที่จะส่งค่าไปสู่การ์ดได้

d. ShiffR8, Mark LB,Mark HB Block เนื่องจากข้อมูลที่ส่งเข้าสู่การ์ดมีขนาด 16 บิต แต่ในระบบของการเป็นการรับข้อมูลเป็นการอ้างตำแหน่งการส่งข้อมูลของคอมพิวเตอร์ ซึ่งรับข้อมูลทีละ 8 บิต จึงต้องมีการจัดเรียงข้อมูลใหม่ดังรูปที่ ก.11 และ ก.12 เพื่อให้สามารถส่งข้อมูลคำสั่งเข้าสู่การ์ดได้

d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	d3	d2	d1	d0
bit 15	bit 14	bit 13	bit 12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	Bit3	bit2	bit1	bit0

⇓(Shift R8 and Mask HB)

0	0	0	0	0	0	0	0	d15	d14	d13	d12	d11	d10	d19	d8
bit 15	bit 14	bit 13	bit 12	bit11	bit10	Bit9	bit8	bit7	bit6	bit5	bit4	Bit3	bit2	bit1	bit0

รูปที่ ก.11 การจัดเรียงข้อมูลในตำแหน่ง high byte

d15	d14	d13	d12	d11	d10	d9	d8	d7	d6	d5	d4	D3	d2	d1	d0
bit 15	bit 14	bit 13	bit 12	bit11	bit10	bit9	bit8	bit7	bit6	bit5	bit4	Bit3	bit2	bit1	bit0

⇓(Shift R8 and Mask HB)

0	0	0	0	0	0	0	0	d7	d6	d5	d4	D3	d2	d1	d0
bit 15	bit 14	bit 13	bit 12	bit11	bit10	bit9	bit8	bit7	bit6	Bit5	bit4	Bit3	bit2	bit1	bit0

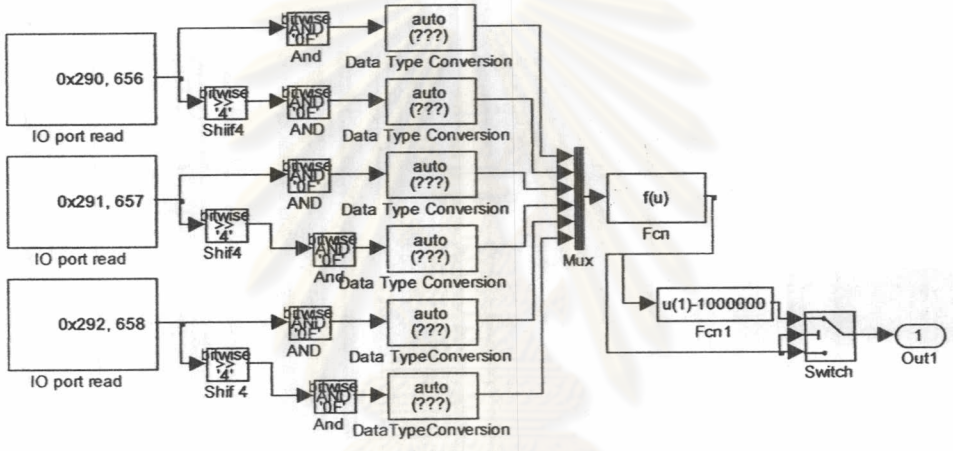
รูปที่ ก.12 การจัดเรียงข้อมูลในตำแหน่ง Low byte

e. 8 Bit HB, 8 Bit LB ใช้ในการเปลี่ยนรูปแบบข้อมูลจากขนาด 16 บิต เป็นข้อมูลขนาด 8 บิต เพื่อให้สอดคล้องกับรับแบบการรับส่งสัญญาณ ของคอมพิวเตอร์

f. IO port write block เป็นส่วนที่ใช้ในการอ้างอิงตำแหน่งของการ์ดเมื่อเทียบกับระบบส่งข้อมูลของคอมพิวเตอร์ เนื่องจากข้อมูลที่การ์ดต้องการมีขนาด 12 บิต และระบบส่งข้อมูลสามารถส่งได้

เพียง 8 บิตต่อตำแหน่งอ้างอิง ดังนั้นจึงต้องใช้ตำแหน่งอ้างอิง 2 ตำแหน่ง จากรูปที่ ก.9 จะใช้ตำแหน่ง 300_h และ 301_h

- Digital input Subsystem (Encoder 8255) เป็นส่วนที่พัฒนาขึ้นมาเพื่อใช้ในการติดต่อกับการ์ด ETT 8255 โดยรายละเอียดของการ์ดได้กล่าวไว้ในบทที่ 3 ในส่วนของการติดต่อกับการ์ด Digital Input ดังแสดงไว้ในรูปที่ ก.13 และรูปที่ ก.14 เป็นการจัดเรียงข้อมูลจาก Interface box สู่ IC 8255 แต่ละตัว โดยแถวแรกแทนค่าที่อ่านได้จาก Interface box สำหรับแถวที่ 2 จะเป็นค่า Binary Code Decimal (BCD) ที่ส่งมาสู่ IC 8255 แถวที่ 3 แทน ตำแหน่งย่อย แต่ละบิตในตำแหน่งอ้างอิงตามแถวที่ 4



รูปที่ ก.13 ส่วนควบคุมที่ใช้ในการติดต่อกับการ์ด Digital Input

1				5				3				7				2				9			
0	0	0	1	0	1	0	1	0	0	1	1	0	1	1	1	0	0	1	0	1	0	0	1
d7	d6	d5	D4	d3	d2	d1	d0	d7	d6	D5	d4	d3	d2	d1	d0	d7	d6	d5	d4	d3	d2	d1	d0
Base Address+2								Base Address+1								Base Address+0							

รูปที่ ก.14 การจัดเรียงข้อมูลที่ใน ETT 8255 Card

a. IO port read block เป็นส่วนที่ใช้ในการอ่านค่าจากการ์ด ETT 8255 เนื่องจาก IC8255 แต่ละตัวจะสามารถรับและส่งข้อมูลขนาด 24 บิต ดังนั้นจึงต้องใช้ตำแหน่งอ้างอิงจำนวน 3 ตำแหน่งในการรับค่าจาก IC 8255 จากรูปที่ ก.13 จะใช้ตำแหน่ง 290_h 291_h และ 292_h

b. Shif4, And Block เนื่องจากข้อมูลที่ส่งออกมาจาก Interface Box (รูปที่ 3.8) ส่งค่าแบบ Binary Code Decimal(BCD) ขนาด 24 บิตไปที่ตำแหน่งอ้างอิง 3 ตำแหน่งดังที่กล่าวมาแล้ว ตำแหน่งละ 8 บิต โดยทุกๆ 4 บิตจะแทนตัวเลขในแต่ละหลักของค่าที่อ่านจาก Encoder ดังรูปที่ ก.14 ดังนั้นจึงต้องมีการเปลี่ยนตำแหน่งและจัดการค่าที่อยู่ในตำแหน่งย่อยของแต่ละตำแหน่งอ้างอิงเพื่อที่จะสามารถนำไปคำนวณต่อไปได้

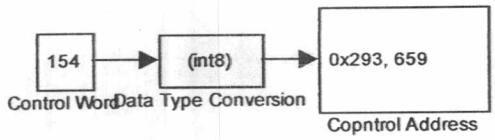
c. Data type conversion block ใช้ในการเปลี่ยนชุดข้อมูลในรูปของเลขฐาน 2 ไปเป็นเลขฐาน 10 เพื่อนำไปใช้ในการคำนวณหาตำแหน่งต่อไป

d. Fcn Block เป็นการนำเลขฐาน 10 ที่ได้จากแต่ละตำแหน่งอ้างอิงมารวมกันเพื่อให้ได้ตำแหน่งที่แท้จริงตามสูตร "100000*u(6)+10000*u(5)+1000*u(4)+100*u(3)+10*u(2)+u(1)" โดย u(.) แทนค่าที่ส่งมาจากตำแหน่งต่างๆ เรียงตำแหน่งที่ 1 จากบนลงล่าง

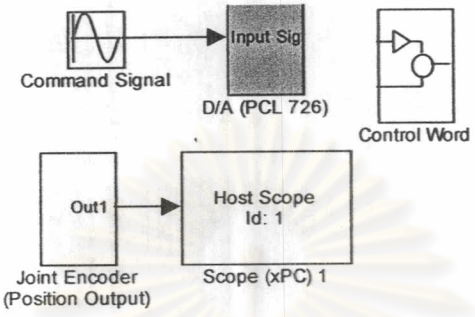
e. Fcn1 Block แสดงค่าเมื่อ ค่าที่คำนวณได้เป็นค่าด้านลบ

f. Switch Block เป็นการเลือกค่าที่จะส่งออกไปแสดงผลว่าเป็นค่าลบหรือค่าบวก

- Mux เป็นส่วนที่ใช้ในการรวมสัญญาณที่คำนวณได้และสัญญาณอ้างอิง
- xPC Scope ใช้ในการแสดงค่าออกไปสู่ Target PC โดยสามารถเลือกวิธีแสดงผลได้ทั้งในรูปแบบของ กราฟ และตัวเลข
- Control Word Block ใช้ในการระบุหน้าที่ของ IC 8255 เพื่อให้ทำหน้าที่ในการรับข้อมูลทั้ง 3 ช่องทางดังรายละเอียดในบทที่ 3 โดยการส่งค่า "10011010" ฐาน2 สู่ตำแหน่งควบคุมของ IC 8255



รูปที่ ก.15 ส่วนควบคุม IC 8255



รูปที่ ก.16 Block Diagram ที่ใช้ในระบบควบคุมแบบเปิด

ก.6.2 โปรแกรมควบคุมระบบเปิด

จากรูปที่ ก.16 ระบบควบคุมแบบเปิดประกอบด้วย

- Command Signal เป็นสัญญาณควบคุมที่ส่งเข้าไปยังการ์ด D/A เพื่อแปลงจากค่าตัวเลข เป็นสัญญาณไฟฟ้า โดยการ์ด D/A สามารถส่งสัญญาณ ไฟฟ้าแบบ Analog ได้ขนาด ± 5 โวลต์ ซึ่งจะสามารถทราบขนาดของสัญญาณไฟฟ้าได้จากสูตร

$$output = \frac{x * 5}{2047} \quad \text{volt} \qquad \text{(ก.1)}$$

เมื่อ x เป็นสัญญาณคำสั่งที่อยู่ในช่วง -2047 ถึง 2048

- D/A Subsystem เป็นส่วนที่ใช้ในการติดต่อกับการ์ด D/A รุ่น PCL 726 ของบริษัท Advantech ซึ่งได้กล่าวรายละเอียดไว้ในข้างต้น
- Digital input Subsystem เป็นส่วนที่ใช้ในการติดต่อกับการ์ด ETT 8255 ซึ่งรายละเอียดไว้ในข้างต้น
- xPC Scope ใช้ในการแสดงค่าออกไปสู่ Target PC โดยสามารถเลือกวิธีแสดงผลได้ทั้งในรูปแบบของ กราฟ และตัวเลข

- Control Word Block ใช้ในการระบุหน้าที่ของ IC 8255 เพื่อให้ทำหน้าที่ในการรับข้อมูลทั้ง 3 ช่องทางดังรายละเอียดในบทที่ 3 โดยการส่งค่า "10011010" ฐาน2 สู่ตำแหน่งควบคุมของ IC 8255

จากที่ได้อธิบายรายละเอียดของโปรแกรมที่ใช้ในการควบคุมทั้งแบบเปิดและแบบปิดนั้น แสดงให้เห็นถึงประสิทธิภาพ ความสะดวก ความยืดหยุ่นในการพัฒนาโปรแกรมควบคุม รวมไปถึงการใช้ Block Diagram ที่สามารถเข้าทำความเข้าใจเกี่ยวกับการทำงานของโปรแกรมได้ง่าย จึงคาดหวังว่า สิ่งที่ได้นำเสนอจะเป็นแนวทางที่ผู้อ่านสามารถเข้าถึงประโยชน์ของ Matlab® Simulink xPC target และสามารถพัฒนาและนำไปใช้ต่อไปได้

ภาคผนวก ข.

Numerical Differential Method

ข.1 นำเรื่อง

ในหัวข้อนี้จะนำเสนอการหาอนุพันธ์ของข้อมูลโดยใช้กรรมวิธีเชิงตัวเลขเพื่อหาค่าความเร็วและความเร่ง ซึ่งเป็นส่วนจำเป็นที่จะต้องนำไปใช้ต่อไปในการศึกษาพฤติกรรมของความเสียดทานที่เกิดขึ้นในหุ่นยนต์ CRS โดยวิธีที่จะนำเสนอต่อไปนี้ เป็นวิธีที่เป็นที่รู้จักกันอย่างแพร่หลายอันได้แก่วิธี Finite Divided Difference Approximation และวิธี 2nd order Polynomial fitting และการเพิ่มความแม่นยำของการหาอนุพันธ์ด้วยวิธี Richardson Extrapolation

ข.2 Finite Divided Difference Approximation

Finite Divided Difference Approximation เป็นวิธีที่รู้จักกันอย่างแพร่หลายในการหาหาอนุพันธ์ จากข้อมูลที่มีอยู่ โดยมีที่มาจากการดัดแปลง อนุกรม ของ Taylor เพื่อนำมาใช้ในการหาอนุพันธ์

Finite Divided Difference Approximation ที่จะนำเสนอนี้แบ่งออกเป็น 3 วิธี ขึ้นอยู่กับการนำไปใช้กับตำแหน่งของข้อมูลที่นำมาหาค่าอนุพันธ์

ข.2.1 Forward Finite Differential

$$f'(x_i) = \frac{-f(x_{i+2}) + 4f(x_{i+1}) - 3f(x_1)}{2h} \qquad \text{(ข.1)}$$

$$f''(x_i) = \frac{-f(x_{i+3}) + 4f(x_{i+2}) - 5f(x_{i+1}) + 2f(x_i)}{12h^2} \qquad \text{(ข.2)}$$

ข.2.2 Central Finite Differential

$$f'(x_i) = \frac{-f(x_{i+2}) + 8f(x_{i+1}) - 8f(x_{i-1}) + f(x_{i-2})}{12h} \qquad \text{(ข.3)}$$

$$f''(x_i) = \frac{-f(x_{i+2}) + 16f(x_{i+1}) - 30f(x_i) + 16f(x_{i-1}) - f(x_{i-2})}{12h^2} \qquad \text{(ข.4)}$$

ข.2.3 Backward Finite Differential

$$f'(x_i) = \frac{3f(x_i) - 4f(x_{i-1}) + f(x_{i-2})}{2h} \quad \text{(ข.5)}$$

$$f''(x_i) = \frac{2f(x_i) - 5f(x_{i-1}) + 4f(x_{i-2}) + 3f(x_{i-3})}{h^2} \quad \text{(ข.6)}$$

เมื่อ x_i คือเวลาที่ตำแหน่ง i

$f(x_i)$ คือข้อมูลตำแหน่งที่เวลา x_i

$f'(x_i)$ คืออนุพันธ์ลำดับที่หนึ่งของข้อมูลตำแหน่ง(ความเร็ว)ที่เวลา x_i

$f''(x_i)$ คืออนุพันธ์ลำดับที่สองของข้อมูลตำแหน่ง(ความเร่ง)ที่เวลา x_i

h คือช่วงห่างของเวลาในการบันทึกข้อมูล (Sampling Time)

ข.3 Richardson's Extapolation

กรรมวิธี Richardson's Extrapolation เป็นวิธีหนึ่งที่ใช้ในการเพิ่มความแม่นยำแก่การหาค่าอนุพันธ์ จากค่าอนุพันธ์ที่มีอยู่แล้ว โดยมีพื้นฐานอยู่บนกฎสี่เหลี่ยมคางหมู

$$I = I(h) + E(h) \quad \text{(ข.7)}$$

เมื่อ I เป็นค่าจริง

$I(h)$ เป็นค่าที่ได้จากการประมาณโดยกรรมวิธีเชิงตัวเลขโดยมี Step size h

$E(h)$ เป็นค่าความผิดพลาดที่เกิดขึ้น

และเมื่อนำแนวทางของ Richardson Extrapolation มาใช้ในการเพิ่มความแม่นยำ แก่การหาค่าอนุพันธ์สามารถใช้ได้ในหลายระดับความละเอียด โดยนำค่าที่ได้มาคำนวณตามสูตร

$$I_n = \frac{4}{3} I_n(h_2) - \frac{1}{3} I_n(h_1) \quad \text{(ข.8)}$$

เมื่อ $I_n(h_1)$ เป็นค่าอนุพันธ์ที่ได้ ที่เวลา n เมื่อค่า Step size คือ h_1

$I_n(h_2)$ เป็นค่าอนุพันธ์ที่ได้ ที่เวลา n เมื่อค่า Step size คือ h_2 โดย $h_2 = \frac{1}{2} h_1$

การเพิ่มระดับความแม่นยำโดยวิธี Richardson's Extrapolation สามารถทำได้หลายระดับ โดยนำค่าที่ได้จากการเพิ่มความแม่นยำ ตามสมการ ข.8 มาคำนวณอีกครั้งตามสมการ

$$I_n = \frac{16}{15} I_n(h_2) - \frac{1}{15} I_n(h_1) \qquad \text{(ข.9)}$$

และเมื่อมีการเพิ่มความแม่นยำในหลายๆ ระดับ รูปแบบของสมการเพิ่มความแม่นยำจะอยู่ในรูปของ

$$I_{j,k.} = \frac{4^{k-1} I_{j+1,k-1} - I_{j,k-1}}{4^{k-1} - 1} \qquad \text{(ข.10)}$$

เมื่อ j คือค่าความละเอียดของ Step size ที่ต้องการในการคำนวณ

k คือระดับความแม่นยำที่ต้องการคำนวณโดยวิธี Richardson's Extrapolation

ตารางที่ ข.1 การเพิ่มความแม่นยำด้วยวิธี Richardson's extrapolation

	Step Size	Diff. Result	$j=1$	$j=2$	$j=3$
$k=2$	$2h$	I_1	$I_{1,2} = \frac{4I_2 - I_1}{3}$		
	h	I_2			
$k=3$	$4h$	I_1	$I_{1,2} = \frac{4I_2 - I_1}{3}$	$I_{2,3} = \frac{16I_{1,3} - I_{1,2}}{15}$	
	$2h$	I_2	$I_{1,3} = \frac{4I_3 - I_2}{3}$		
	h	I_3			
$k=4$	$8h$	I_1	$I_{1,2} = \frac{4I_2 - I_1}{3}$	$I_{2,3} = \frac{16I_{1,3} - I_{1,2}}{15}$	$I_{3,4} = \frac{64I_{2,4} - I_{2,3}}{63}$
	$4h$	I_2	$I_{1,3} = \frac{4I_3 - I_2}{3}$	$I_{2,4} = \frac{16I_{1,4} - I_{1,3}}{15}$	
	$2h$	I_3	$I_{1,4} = \frac{4I_4 - I_3}{3}$		
	h	I_4			

ข.4 2nd order Polynomial Fitting

การใช้ 2nd Order Polynomial Regression เป็นอีกวิธีหนึ่งในการหาค่าอนุพันธ์ที่เข้าใจได้ง่ายและเป็นที่รู้จักกันอย่างแพร่หลาย โดยมีหลักการอยู่ที่การใช้ข้อมูลจำนวนหนึ่ง มาประมวลเข้ากับสมการ 2nd Order Polynomial และแทนค่าอนุพันธ์ของแต่ละจุดด้วยค่าความชันของสมการในจุดนั้นๆ

$$\begin{bmatrix} n & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} \sum y_i \\ \sum x_i y_i \\ \sum x_i^2 y_i \end{Bmatrix} \qquad \text{(ข.11)}$$

$$y = a_0 + a_1 x + a_2 x^2 \qquad \text{(ข.12)}$$

เมื่อ n คือจำนวนจุดที่ต้องการนำมาพิจารณา

a_0, a_1, a_2 คือสัมประสิทธิ์ของสมการ 2nd Order Polynomial Fitting โดย

a_1 เป็นค่าอนุพันธ์ลำดับที่ 1 ของชุดข้อมูล

a_2 เป็นค่าอนุพันธ์ลำดับที่ 2 ของชุดข้อมูล

x_i คือข้อมูลในแกนนอน ในที่นี้คือข้อมูลเวลา

ภาคผนวก ค

Non-linear Least Square Method

ค.1 นำเรื่อง

จากปัญหาที่ได้กล่าวไว้ในบทที่ 3 การคำนวณหาค่าแรงเสียดทานที่เกิดขึ้นในหุ่นยนต์ CRS Robotics โดยมีเพียงชุดข้อมูลตำแหน่งและชุดข้อมูลสัญญาณคำสั่ง โดยอ้างอิงแบบจำลองความเสียดทานหุ่นยนต์ Coulomb Viscous Friction และค่าโมเมนต์ความเฉื่อยของหุ่นยนต์ ซึ่งทั้งสองตัวแปรไม่สามารถคำนวณแยกจากกันได้ จึงเป็นที่มาของการนำเสนอวิธี Least square curve fitting มาใช้ในการหาค่าที่เหมาะสม เพื่อใช้ในการศึกษาพฤติกรรมของแรงเสียดทานที่เกิดขึ้นในหุ่นยนต์ CRs Robotics

ค.2 Least Square Curve Fitting

ในส่วนนี้จะนำเสนอแนวคิดเกี่ยวกับ Least Square Curve Fitting เพื่อเป็นแนวทางในการนำไปใช้และศึกษาต่อไป

รูปแบบปัญหา Least Square คือการทำให้เทอมของค่าความผิดพลาดในจากความแตกของข้อมูลที่ได้จากการวัดและค่าที่ได้จากการคำนวณมีค่าน้อยที่สุดดังสมการ ค.1

$$\min_{x\in R^n} f(x) = \frac{1}{2}\|F(x)\|_2^2 = \frac{1}{2}\sum_i F_i(x)^2 \qquad \text{(ค.1)}$$

เมื่อ $F_i(x)$ คือค่าความผิดพลาดที่เกิดจากข้อมูลจริงและฟังค์ชั่นที่ต้องการประมาณค่า

โดยปัญหาลักษณะนี้จะพบมากในงานที่ต้องการหาค่าตัวแปรในการพัฒนาแบบจำลองต่างๆ รวมถึงงานระบบควบคุมที่ต้องการติดตามระบบใดๆ ซึ่งสามารถอธิบายด้วยสมการ

$$\min_{x\in R^n} \int_1^2 [y(x,t) - \phi(t)]^2 \, dt \qquad \text{(ค.2)}$$

เมื่อ $y(x,t)$ คือรูปแบบของระบบควบคุม

$\phi(t)$ คือชุดของข้อมูลที่ต้องการให้ระบบติดตาม

และในระบบไม่ต่อเนื่องสามารถอธิบายในรูปแบบ

$$\min_{x \in R^n} \sum_{i=1}^{m} \left[\bar{y}(x,t) - \bar{\phi}(t)\right]^2 \qquad \text{(ค.3)}$$

เมื่อ $\bar{y}(x,t)$ คือรูปแบบของระบบควบคุม

$\bar{\phi}(t)$ คือชุดของข้อมูลที่ต้องการให้ระบบติดตาม

วิธีที่นิยมใช้กันมากวิธีหนึ่งในการหาค่าที่เหมาะสมของตัวแปรจาก Least Square Method คือการใช้ Gradient Method เพื่อหาค่าที่น้อยที่สุดหรือมากที่สุดจากอนุพันธ์ลำดับที่หนึ่งของชุดข้อมูล โดย Gradient Method จะใช้ความชันของข้อมูลในการคำนวณทิศทางที่จะไปสู่ค่าสูงสุดหรือต่ำสุด ซึ่งจะเป็นไปในทิศของ $-\nabla f(x)$ ปัญหาหนึ่งที่เกิดจากการใช้วิธีนี้คือ เมื่อรูปแบบของชุดข้อมูล มีจุดสูงสุดหรือจุดต่ำสุดมากกว่าหนึ่งจุด จะทำให้ไม่สามารถหาค่าสูงสุดหรือต่ำสุดที่แท้จริงได้

ในงานวิจัยนี้ใช้ฟังค์ชั่น lsqcurvefit.m ซึ่งเป็นฟังค์ชั่นใน Optimization Toolbox ของโปรแกรม Matlab® ที่ใช้สำหรับการแก้ปัญหาเกี่ยวกับการประมาณค่าตัวแปรของฟังค์ชั่นที่มาประมาณข้อมูล ในรูปแบบของ nonlinear function ซึ่งมีรูปแบบการใช้คือ

$$[x, resnorm, residual.flag] = lsqcurvefit(fun, x_0, x\_data, y\_data) \quad \text{(ค.3)}$$

เมื่อ x คือตัวแปรในฟังค์ชั่น fun ที่ได้จากการใช้ Least Square method

$resnorm = \sum (fun(x, x\_data) - y\_data)^2$ เป็นค่าความผิดพลาดที่เกิดขึ้น

$residual = fun(x, x\_data) - y\_data$ เป็นค่าความผิดพลาดที่ตำแหน่งต่างๆ ของฟังชั่น

$flag$ เป็นดัชนีที่ใช้ในการชี้บ่งว่า lsqcurvefit ลู่เข้าหรือลู่ออก ในการหาค่าตัวแปร x

โดย $flag > 0$ หมายความว่าสมการลู่เข้า สามารถหาค่าตัวแปร x ได้

$flag = 0$ หมายความว่าจำนวนครั้งในการคำนวณ มากกว่าค่าที่กำหนดไว้ โดยยังไม่สามารถสรุปค่าตัวแปร x ได้

$flag < 0$ หมายความว่าสมการลู่ออก ไม่สามารถหาค่าตัวแปร x ได้

fun คือฟังค์ชั่นที่ต้องการหาค่าตัวแปร

x_0 เป็นการกำหนดค่าเริ่มต้น ในการประมวลผลของ lsqcurvefit ซึ่งการกำหนดค่าเริ่มต้น ที่ดีจะทำให้ใช้เวลาในการคำนวณน้อย และค่าเริ่มต้นที่ไม่เหมาะสมจะทำให้เสียเวลาในการคำนวณ หรืออาจทำให้การคำนวณลู่ออก ไม่สามารถหาค่าตัวแปรได้

x_data เป็นชุดข้อมูลของตัวแปรต้น สำหรับสมการที่ต้องการหาค่าสัมประสิทธิ์

y_data เป็นชุดข้อมูลของตัวแปรตาม ที่อาจได้จากการผลการทดลอง ซึ่งใช้เป็นข้อมูลอ้างอิงในการระบุค่าผิดพลาดจากการคำนวณ

ตัวอย่างการใช้ ฟังค์ชั่น lsqcurvefit.m

```
F=inline('x(1)*xdata.^2+x(2)*sin(xdata)','x','xdata');
X=lsqcurvefit(f,x0,xdata,ydata);
```

นอกเหนือจากที่ได้กล่าวมาแล้ว ยังมีความสามารถเพิ่มเติมอีกหลายอย่างของฟังค์ชั่น lsqcurvefit.m ซึ่งสามารติดตามหรือศึกษาได้จากคู่มือของ Matalb® Optimization Toolbox หรือใน http://www.mathworks.com/access/helpdesk/help/toolbox/optim/optim.shtml

ภาคผนวก ง.

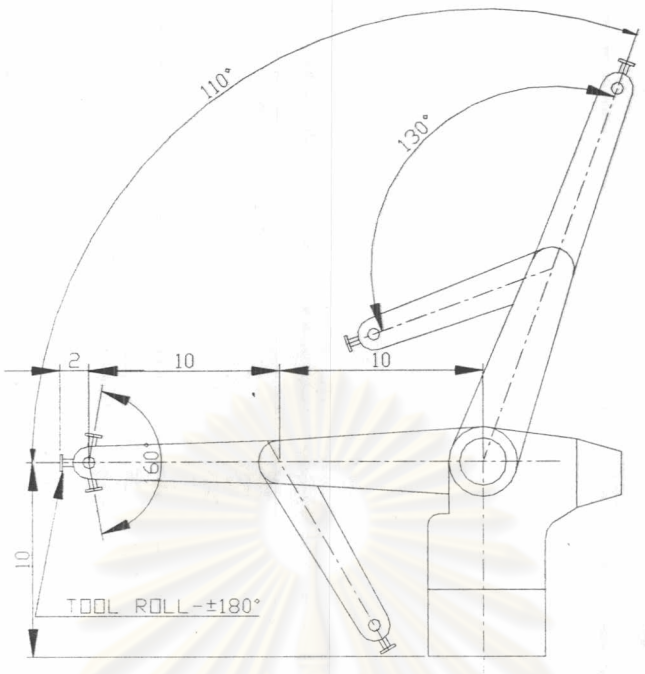
รายละเอียดของหุ่นยนต์ CRS

ตารางที่ ง.1 รายระเอียดเกี่ยวกับหุ่นยนต์ CRS

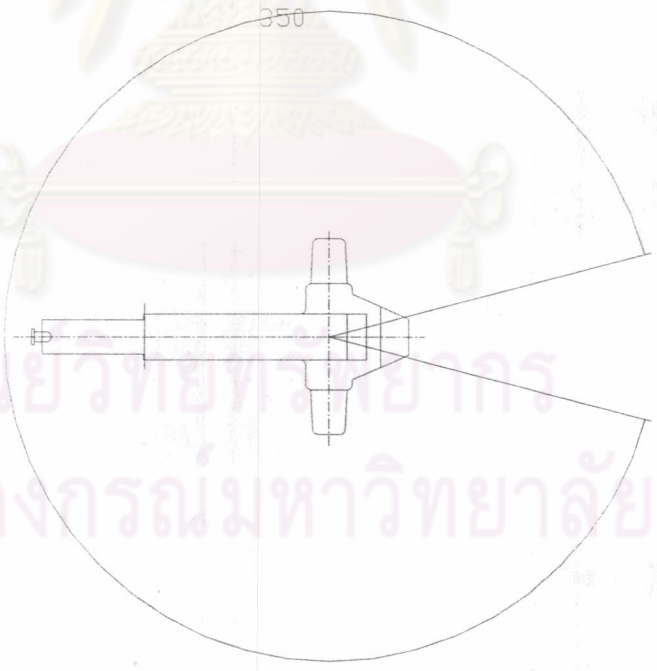
STRUCTURE		ARTICURATED 5 DOF
DRIVE MOTOR		PERMANNT MAGNET DC SERVO
	Bearings	ABEC Class -0.375" ID
	Max voltage	+/-25Vdc
	Max current	10.8 amps
	Mech. Time const	11.62 msec
	Max speed @ 25V	3600 rpm
	Peak torque	100 oz-in
	Brush life	8000 hours@1200 rpm
TRANSMISSION		
	Waist rotation	Size 20 cup type harmonic drive
	Shoulder	Size 20 cup type harmonic drive
	Elbow	Size 20 cup type harmonic drive/chain
	Wrist bend(pitch)	bevel-/spur-gear/chain
	Tool roll	bevel-/spur-gear/chain/gear
PAYLOAD		k.g.
	Maximum design	2.0
	Full speed/acc	1.0
REACH-WAIST TO TOOL FLANGE		22 inches
REACH (by link)		inches
	Base to Shoulder	10
	Shoulder to elbow	10
	Elbow to wrist pivot	10
	Wrist pivot to tool flange	2
JOINT TRAVEL RANGES		degrees

	Waist rotation	+/-175
	Shoulder	+110,0
	Elbow	+0,-130
	Wrist bend(pitch)	+/-110
	Tool roll	+/-180
JOINT SPEEDS AT 100% PROGRAM SPEED		rad/sec
	A150 Series:	
	Waist rotation	1.74
	Shoulder3	1.08
	Elbow	1.74
	Wrist bend (pitch)	3.14
	Tool roll	6.28
	A250 Series:	
	Waist rotate	3.05
	Shoulder	2.18
	Elbow	3.05
	Wrist bend (pitch)	3.14
	Tool roll	6.28
JOINT DEFAULT ACCELERATION RATES		
	A150 Series:	
	Waist rotation	5.45
	Shoulder3	5.45
	Elbow	5.45
	Wrist bend (pitch)	24.54
	Tool roll	49.09
	A250 Series:	
	Waist rotate	12.93
	Shoulder	12.93
	Elbow	12.93

	Wrist bend (pitch)	58.18
	Tool roll	116.36
POSITION FEEDBACK		
	Re0solution	100 pulse/rev
	Index	Marker pulse 1 per rev
	Output	Chnis A,B,Z sq wave TTL
JOINT RESOLUTION		
	Waist rotation	0.005
	Shoulder3	0.005
	Elbow	0.005
	Wrist bend (pitch)	0.023
	Tool roll	0.045
JOINT RESOLUTION		Inchs @ tool flang
	Waist rotation	0.0019
	Shoulder3	0.0009
	Elbow	0.0009
	Wrist bend (pitch)	
	Tool roll	0.0016



รูปที่ ง.1 แขนหุ่นยนต์แบบ articulated ของบริษัท CRS Robotics รุ่น 255 ด้านข้าง



รูปที่ ง.2 แขนหุ่นยนต์แบบ Articulated ของบริษัท CRS Robotics รุ่น 255 ด้านบน

ภาคผนวก จ.

โปรแกรมที่ใช้ในการหาค่าตัวแปรของแรงเสียดทาน

จ.1 นำเรื่อง

ในส่วนนี้ของวิทยานิพนธ์จะกล่าวถึงโปรแกรมที่เขียนขึ้นมาเพื่อหาค่าความเร็ว ความเร่ง จากข้อมูลตำแหน่งของหุ่นยนต์ CRS Robotics โดยวิธีดังที่ได้กล่าวมาแล้วในบทต้นๆและโปรแกรมในการหาค่าตัวแปรต่างๆ ของแรงเสียดทานโดยใช้หลักการของ Non-linear least square curvefit ในการทดสอบหาค่าตัวแปรที่เหมาะสมที่สุดสำหรับข้อมูลเท่าที่มีอยู่ ทั้งในการหาค่าตัวแปรของแบบจำลองความเสียดทานของ Coulomb Viscous Friction และแบบจำลองที่แสดงผลของสัญญาณความถี่สูง

โปรแกรมที่พัฒนาขึ้นมานี้ใช้การเขียนในรูปแบบของ m-file ซึ่งเป็นส่วนหนึ่งของโปรแกรม Matlab® รูปแบบการใช้โปรแกรมจะเข้าได้ง่ายเนื่องจากเป็นการเขียนด้วยภาษาชั้นสูง และยังมีฟังค์ชั่นภายในให้เลือกใช้มากมาย ส่วนหนึ่งของฟังค์ชั่นภายในที่มีประโยชน์มากและเป็นส่วนสำคัญส่วนหนึ่งที่ทำให้วิทยานิพนธ์นี้สำเร็จลุล่วงได้คือฟังค์ชั่น lsqrcurvefit.m ซึ่งได้อธิบายไว้ในภาคผนวก ค

จากโปรแกรมที่จะนำเสนอนี้ ผู้จัดทำคาดหวังว่าจะสามารถยังประโยชน์แก่ผู้ที่นำวิทยานิพนธ์นี้ไปศึกษาได้ต่อไป

จ.2 โปรแกรมที่ใช้การหาความเร็วจากข้อมูลตำแหน่งด้วยวิธีรักษาระห่างในแนวแกนตำแหน่ง

โปรแกรมนี้ใช้ในการประมวลผลบน Matlab® M-file editor ทำงานโดยการดึงข้อมูลตำแหน่งและสัญญาณคำสั่งจากไฟล์ D_CRS_01_250_Dith.mat ซึ่งเป็นไฟล์ข้อมูลที่รับมาจาก Target PC แล้วเก็บไว้บน Work space ของ Matlab® จากนั้นจึงบันทึกเป็นไฟล์ D_CRS_01_250_CR60_Dith.mat

ตัวแปรควบคุมสำหรับโปรแกรมนี้คือ ค่า Sampling time (h) ต้องเป็นค่าเดียวกับที่ใช้บน xPC Target ซึ่งในที่นี้ใช้ค่า 0.002 วินาที สำหรับตัวแปรควบคุมต่อมาคือ t ซึ่งใช้ในการแทนข้อมูลเวลา ze ใช้ในการแทนข้อมูลตำแหน่ง และ ComE ใช้ในการแทนข้อมูลสัญญาณคำสั่ง

โปรแกรมหลักที่ใช้ในการประมวลผล

```
clear;clc;load D_CRS_01_250_Dith.mat;clc
totdat=length(t);h=0.002;sec=60;
max_p=100;lim=10;
for i=1:max_p%Forward
   jp(i)=0;jm(i)=0;
   while (abs(ze(i)-ze(i+jp(i)))<lim & jp(i)~=max_p)%this loop for find the 1st point
      jp(i)=jp(i)+1;
   end
   kp(i)=round(jp(i)/2);%For section interval of period for velocity
   mp(i)=round(jp(i)/3);%For section interval of period for acceleration
   xv(i,:)=[ze(i+2*kp(i)) ze(i+kp(i)) ze(i) 0 0];
   xa(i,:)=[ze(i+3*mp(i)) ze(i+2*mp(i)) ze(i+mp(i)) ze(i) 0];
   v(i)=(-xv(i,1)+4*xv(i,2)-3*xv(i,3))/(2*h*kp(i));
   a(i)=(-xa(i,1)+4*xa(i,2)-5*xa(i,3)+2*xa(i,4))/((h*mp(i))^2);
end
for i=max_p+1:totdat-max_p%Central
   jp(i)=0;jm(i)=0;
   while (abs(ze(i)-ze(i+jp(i)))<lim & jp(i)~=max_p)%this loop for find the 2nd end point
      jp(i)=jp(i)+1;
   end
   while (abs(ze(i)-ze(i+jm(i)))<lim & jm(i)~=-max_p)%this loop for find 3rd end point
      jm(i)=jm(i)-1;
   end
   kp(i)=round(jp(i)/2);km(i)=round(-jm(i)/2);%For section interval of period for velocity
   mp(i)=round(jp(i)/2);mm(i)=round(-jm(i)/2);%For section interval of acceleration
   xv(i,:)=[ze(i+2*kp(i)) ze(i+kp(i)) ze(i) ze(i-km(i)) ze(i-2*km(i))];
   xa(i,:)=[ze(i+2*mp(i)) ze(i+mp(i)) ze(i) ze(i-mm(i)) ze(i-2*mm(i))];
   v(i)=(-xv(i,1)+8*xv(i,2)-8*xv(i,4)+xv(i,5))/(12*((t(i+2*kp(i))-t(i-2*km(i)))/4));
```

```
   a(i)=(-xa(i,1)+16*xa(i,2)-30*xa(i,3)+16*xa(i,4)-xa(i,5))/(12*((t(i+2*kp(i))-t(i-
2*km(i)))/4)^2);
end
for i=totdat-max_p+1:totdat%Backward
   jp(i)=0;jm(i)=0;
   while (abs(ze(i)-ze(i+jm(i)))<lim & jm(i)~=-max_p)%this loop for find 4th end point
      jm(i)=jm(i)-1;
   end
   km(i)=round(-jm(i)/2);%For section interval of period for velocity
   mm(i)=round(-jm(i)/3);%For section interval of period for acceleration
   xv(i,:)=[ze(i) ze(i-km(i)) ze(i-2*km(i)) 0 0];
   xa(i,:)=[ze(i) ze(i-mm(i)) ze(i-2*mm(i)) ze(i-3*mm(i)) 0];
   v(i)=(3*xv(i,1)-4*xv(i,2)+xv(i,3))/(2*h*km(i));
   a(i)=(2*xa(i,1)-5*xa(i,2)+4*xa(i,3)-xa(i,4))/((h*mm(i))^2);
end
n=round(2*pi/(sec*w*h))
for i=1:totdat-n%This part for correction velocity
   sxy(i)=0;sx2(i)=0;sx(i)=0;sy(i)=0;
   for j=1:n%-jp(i)%See "close all;plot([ze*.1 v'*.05 -jm'],'.')" for notice
      x(j)=t(i+j-1);y(j)=v(i+j-1);
      sxy(i)=x(j)*y(j)+sxy(i);sx2(i)=sx2(i)+x(j)^2;
      sx(i)=sx(i)+x(j);sy(i)=sy(i)+y(j);
   end
   a1(i)=(n*sxy(i)-sx(i)*sy(i))/(n*sx2(i)-sx(i)^2);%This is ar for Calculate Acceleration
   a0(i)=sy(i)/n-a1(i)*sx(i)/n;
   vr(i)=a0(i)+a1(i)*t(i);%ar is a1
end
clear scope2_time scope2_data scope1_time scope1_data tg unit
clear scope5_time scope5_data scope6_time scope6_data
figure;plot(ze);grid on;title('Actual position(pulse)');
xlabel('Time(s)');ylabel('Position (Count)');%legend('Actual',0);
```

```
figure;plot(v);grid on;xlabel('time(s)');ylabel('Velocity(pulse/s)');title('Velocity')
figure;plot(vr);grid on;xlabel('time(s)');ylabel('Velocity(pulse/s)')
figure;plot(a);grid on;xlabel('time(s)');ylabel('Acceleration');title('Accerelation')
figure;plot(a1);grid on;xlabel('time(s)');ylabel('Acceleration');title('Accerelation')
figure;plot(ComE);grid on;xlabel('time(s)');ylabel('Control Efford');title('Control Efford')
figure;plot(v,ComE);grid on;xlabel('Velocity(Counts/s)');ylabel('Command Input');
title('V-Td ; Fre = 20 rad/s')
figure;plot(vr,ComE(1:totdat-n));grid on;xlabel('Velocity');ylabel('Command Input');
figure;plot(a,ComE);grid on;xlabel('Acceleration(pulse/s^2)');ylabel('Command Input');
title('J & a');beep
```

จ.3 โปรแกรมที่ใช้การหาค่าตัวแปรจากแบบจำลองความเสียดทานแบบ Coulomb Viscous Friction

โปรแกรมนี้ใช้ในการประมวลผลบน Matlab® M-file editor ทำงานโดยการดึงข้อมูลตำแหน่งและสัญญาณคำสั่งจากไฟล์ D_CRS_01_250_CR60_Dith.mat ซึ่งเป็นไฟล์ข้อมูลที่รับมาจากการหาค่าความเร็วและความเร่งจากวิธีรักษาระห่างในแนวแกนตำแหน่งจากโปรแกรมก่อนหน้านี้

โปรแกรมหลักที่ใช้ในการประมวลผล

```
clear;load D_CRS_01_250_CR60_Dith.mat;clc
ini_j=.0012;dj=.0001;fin_j=.0014;
j=1;v1=vr;
%This Section for compensate Friction%
for int_j=ini_j:dj:fin_j
   p=[1 1 1 0 0];
   int_fric(j,:)=ComE(1:totdat-n)'-int_j*a1;
   %This part for fit the true data to  the friction function by Nonlinear Lease squre
   %The model to fit is Coulomb+Viscous curve
   %Variable need for fit curve are f(friction) fi(command torque) v(velocity) t(time)
```

```
    x=[100 1];% [Coulomb Static StribVel ViscousCoef]
    tor=50;
    for i=1:length(int_fric)
        if (v1(i)>tor&ComE(i)>0)
            v1_est(p(1))=v1(i);f1_est(j,p(1))=int_fric(j,i);p(1)=p(1)+1;
        elseif (v1(i)<-tor&ComE(i)<0)
            v2_est(p(2))=v1(i);f2_est(j,p(2))=int_fric(j,i);p(2)=p(2)+1;
        elseif ((v1(i)>-tor&ComE(i)<0&v1(i)<0)|(v1(i)<tor&ComE(i)>0&v1(i)>0))
            v3_est(p(3))=v1(i);f3_est(j,p(3))=int_fric(j,i);p(3)=p(3)+1;
        end
    end
[v1_par(j,:),nor(1),res1,flag1]=lsqcurvefit(@F_Coulomb,x,v1_est,f1_est(j,:));
    [v2_par(j,:),nor(2),res2,flag2]=lsqcurvefit(@F_Coulomb,-x,v2_est,f2_est(j,:));
    %This loop for cal torque from coefficient fron lease squre at v<>0
    t1_est(j,:)=v1_par(j,1)+v1_par(j,2)*v1_est;
    t2_est(j,:)=v2_par(j,1)+v2_par(j,2)*v2_est;
    p(4)=(max(f3_est(j,:))-min(f3_est(j,:)))/(2*tor);
    p(5)=min(f3_est(j,:))-p(4)*(-tor);
    v3_par(j,:)=[p(5) p(4)];
    t3_est(j,:)=p(5)+v3_est*p(4);%Friction Torque from Coulomb Model around 0
    Err(j)=nor(1)+nor(2)%+sum((f3_est-t3_est(j,:)).^2);
    j=j+1;
end
int_j=ini_j;
for j=1:length(Err)
    if Err(j)==min(Err)
        jr=int_j;FricR=int_fric(j,:);cp=v1_par(j,:);cm=v2_par(j,:);co=v3_par(j,:);
        Tp_Cou=t1_est(j,:);Tm_Cou=t2_est(j,:);To_Cou=t3_est(j,:)
    End
    int_j=int_j+dj;
end
```

```
jr
close all;plot(vr,ComE(1:(totdat-n)));title('Actual Velocity&Torque');grid on
figure;hold on;
plot(vr,FricR);title('Actual Velocity&Torque');grid on
plot(v1_est,Tp_Cou,v2_est,Tm_Cou,v3_est,To_Cou);title('Velocity&Torque');grid on
for j=1:length(Err)
   figure
   hold on;    plot(vr,int_fric(j,:));title('Actual Velocity&Torque');grid on
   plot(v1_est,t1_est(j,:),v2_est,t2_est(j,:),v3_est,t3_est(j,:));title('V&T');grid on
end
beep
clear tor p i j  int_j x v1 tor  v1_par v2_par v3_par  f1_est f2_est f3_est a a0 dj fin_j ini_j
clear  nor sec v ze t
```

โปรแกรมย่อย F_Coulomb.m ใช้ในการระบุรูปแบบสมการของแบบจำลองความเสียดทานของ Coulomb Viscous Friction

```
function torq=F_Coulomb(x,xdat)
for i=1:length(xdat)
torq(i)=x(1)+x(2)*xdat(i);%Coulomb model
end
```

จ.4 โปรแกรมที่ใช้การหาค่าตัวแปรจากแบบจำลองความเสียดทานแบบ Coulomb Viscous Friction with Dither Effect

โปรแกรมนี้ใช้ในการประมวลผลบน Matlab® M-file editor ทำงานโดยการดึงข้อมูลตำแหน่งและสัญญาณคำสั่งจากไฟล์ D_CRS_01_250_CR60_Dith.mat ซึ่งเป็นไฟล์ข้อมูลที่รับมาจากการหาค่าความเร็วและความเร่งจากวิธีรักษาระห่างในแนวแกนตำแหน่งจากโปรแกรมก่อนหน้านี้

โปรแกรมหลักที่ใช้ในการประมวลผล

```
clear;load D_CRS_01_250_CR60_Dith.mat;clc
ini_j=.0011;dj=.0001;fin_j=.0014;v1=vr;
%This Section for compensate Friction1
close all;k=1;
tor=2000;x_d=[100 tor];j=1;
   for int_j=ini_j:dj:fin_j
      p=[1 1 1 0 0];
      int_fric(j,:)=ComE(1:totdat-n)'-int_j*a1;
      %This part for fit the true data to  the friction function by Nonlinear Lease squre
      %The model to fit is Coulomb+Viscous+Dither curve
      %Variable need for fit curve are f(friction) fi(command torque) v(velocity) t(time)
      x=[100 1];% [Coulomb Static StribVel ViscousCoef]
      for i=1:length(int_fric)
         if (v1(i)>tor&ComE(i)>0)
            v1_est(p(1))=v1(i);f1_est(j,p(1))=int_fric(j,i);p(1)=p(1)+1;
         elseif (v1(i)<-tor&ComE(i)<0)
            v2_est(p(2))=v1(i);f2_est(j,p(2))=int_fric(j,i);p(2)=p(2)+1;
            %elseif ((v1(i)>-tor&ComE(i)<0)|(v1(i)<tor&ComE(i)>0))
         elseif ((v1(i)>-tor&ComE(i)<0&v1(i)<0)|(v1(i)<tor&ComE(i)>0&v1(i)>0))
            v3_est(p(3))=v1(i);f3_est(j,p(3))=int_fric(j,i);p(3)=p(3)+1;
         end
      end
      [v1_par(j,:),nor(1),res1,flag1]=lsqcurvefit(@F_Coulomb,x,v1_est,f1_est(j,:));
      [v2_par(j,:),nor(2),res2,flag2]=lsqcurvefit(@F_Coulomb,-x,v2_est,f2_est(j,:));
      [v3_par(j,:),nor(3),res3,flag3]=lsqcurvefit(@F_Dither,x_d,v3_est,f3_est(j,:));
      t1_est(j,:)=v1_par(j,1)+v1_par(j,2)*v1_est;%Fricion Torque + side
      t2_est(j,:)=v2_par(j,1)+v2_par(j,2)*v2_est;%Friction Torque at -side
      t3_est(j,:)=v3_par(j,1)*atan(pi*v3_est/v3_par(j,2));%Friction Torque around zero
      Err(j)=nor(1)+nor(2)+nor(3)%+sum((f3_est-t3_est(j,:)).^2);
```

```
    j=j+1;
  end
  int_j=ini_j;
  for j=1:length(Err)
    if Err(j)==min(Err)
      MinErr(k)=Err(j);
      jr(k)=int_j;FricR(k,:)=int_fric(j,:);cp(k,:)=v1_par(j,:)
      cm(k,:)=v2_par(j,:)
      co(k,:)=v3_par(j,:)%Coeffcient
      Tp_Cou(k,:)=t1_est(j,:);Tm_Cou(k,:)=t2_est(j,:);To_Cou(k,:)=t3_est(j,:);
    End
    int_j=int_j+dj;
  end
  jr
  figure;hold on;plot(vr,FricR(k,:));title('Actual Velocity&Torque');grid on
  %figure;plot(v1_est,Tp_Cou(k,:),v2_est,Tm_Cou(k,:),v3_est,To_Cou(k,:));grid on
beep
```

โปรแกรมย่อยที่ใช้ในโปรแกรมหลักส่วนนี้ประกอบด้วย F_Coulomb.m ซึ่งได้กล่าวไว้แล้วในข้างต้นและ F_Dither.m ใช้ในการระบุรูปแบบสมการของแบบจำลองความเสียดทานของ Coulomb Viscous Friction with Dither

```
function torq=F_Coulomb(x,xdat)
for i=1:length(xdat)
torq(i)=x(1)*atan(pi*xdat(i)/x(2));%Coulomb model
end
```

ประวัติผู้เขียนวิทยานิพนธ์

นายสิรณัฐ ขันธจีรวัฒน์ เกิดเมื่อวันที่ 20 มิถุนายน พ.ศ.2516 ที่กรุงเทพมหานคร หลังจากจบมัธยมศึกษา ปีที่ 6 จาก โรงเรียน ราชวินิตบางแก้ว ได้เข้าศึกษาต่อในคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่ และสำเร็จการศึกษาในระดับปริญญาตรีวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเชียงใหม่ ในปีการศึกษา 2538 และได้เข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีพ.ศ. 2543