รายการอ้างอิง

Balchen, J., G., *Process Control*, Van Nostrand Reinhold, New York, (1988).

Cheney, W. and David Kincaid, *Numerical Mathematics and Computing*, Brooks/Cole Publishing Company, California, (1994).

Douglas, J.,M., *Process Dynamics and Control*, Prentice Hall, New Jersey, (1972).

Heckenthaler, T. and Sebastian Engell., Approximately Time-Option Fuzzy Control of a Two-Tank System, *IEEE Control System*, June (1994).

Luyben, W., L., *Process Modeling Simulation and Control for Chemical Engineers*, McGraw-Hill, New York, (1990).

McCord, J., W., *Borland C++ Programmer's Guide to Graphics*., Sams (1991).

Perry, R., H., *Perry's Chemical Engineers' Handbook*, McGraw-Hill, New York, (1984).

ภาคผนวก ก
โฟลว์ชาร์ทและอัลกอริธึม

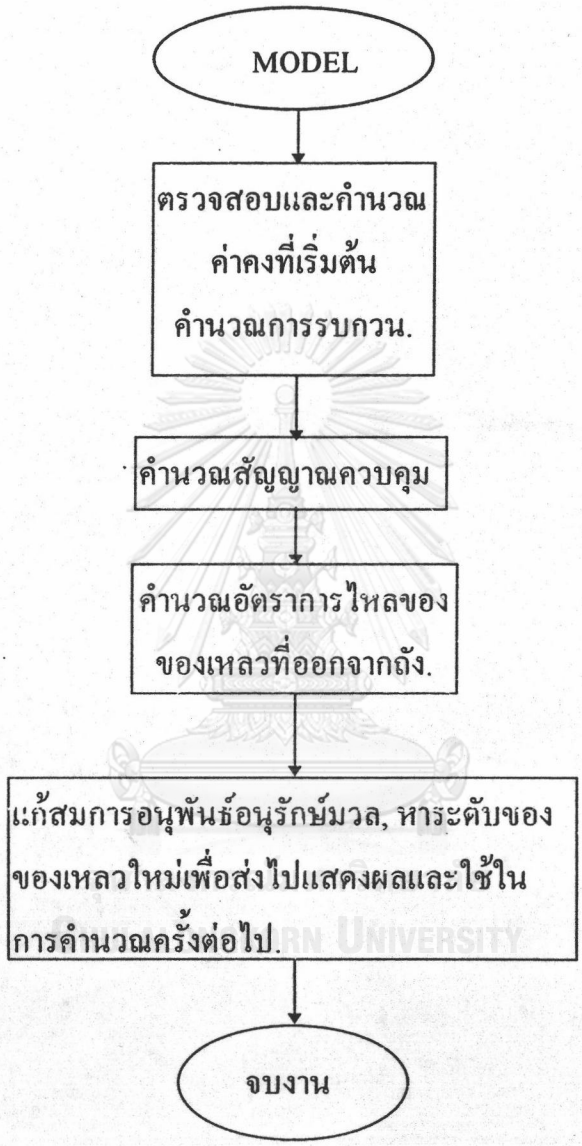
ส่วนผนวกนี้จะขอยกฟังก์ชันบางฟังก์ชันมาเสนอไว้ เพื่อเป็นประโยชน์สำหรับผู้ที่ต้องการนำแนวคิดของการควบคุมระดับ ไปประยุกต์เพื่อเขียนโปรแกรมของตัวเอง และเพื่อเป็นประโยชน์สำหรับผู้ที่ต้องการศึกษาการเขียนโปรแกรมเลียนแบบระบบในแบบ พลวัต โดยตัวสมการการจำลองระบบอยู่ในรูปของสมการอนุพันธ์ ไม่ใช่ทรานส์เฟอร์ฟังก์ชัน.

ก.1 ฟังก์ชัน MODEL

ฟังก์ชันนี้จะทำหน้าที่จำลองแบบระบบทั้งหมด คือทั้งสองถังและถังเดียว ซึ่งในตัวโปรแกรม จะรู้จักระบบถังเดียวในชื่อของถังที่ 3.

ก.1.1 โฟลว์ชาร์ทของฟังก์ชัน

จะแสดงโฟลว์ชาร์ทเพียงถังเดียวเท่านั้น ส่วนในถังอื่นๆ ก็มีลักษณะเดียวกัน.



รูปที่ ก.1 โฟลว์ชาร์ทของฟังก์ชัน MODEL.

ก.1.2 อัลกอริธึมของฟังก์ชัน MODEL

```
void Model(void)
{
    float a1 = PI * (g_tank[0].dia*g_tank[0].dia)/4 ;
    float a2 = PI * (g_tank[1].dia*g_tank[1].dia)/4 ;
    float a3 = PI * (g_tank[2].dia*g_tank[2].dia)/4 ;
    float dh1,  dh2,  dh3;
    float Kc,   R,   Td;
    float Disturb;
    float p1,  p2, p3;
    float temp,  flow_direction;
    int   i;

    /* time unit in second */
    if (g_recalc){
        InitVar();
        g_recalc = OFF;
        g_opr_t  = 0;
        g_SimNo  = 0;
        for(i=0; i<3; i++){
            g_tank[i].cur_h        = 0;
            g_model[i].error       = 0;
            g_model[i].error1      = 0;
            g_model[i].error2      = 0;
            g_model[i].integral    = 0;
            g_model[i].pv          = 0;
            g_model[i].pv1         = 0;
```

```
                g_model[i].pv2          = 0;
        }
}

/***************/
/*** TANK 1 ***/
/***************/
g_f1    =  g_fp0 * g_v1_k/(g_v1_k+g_v4_k)/1000/60;
Disturb  = Disturbance();

Get_K_R_Td(0,&Kc,&R,&Td);
co1.t    =  GetPID(0,Kc,R,Td);
p1       =  1-co1.t;
g_mvt[0]  = p1;
if (((g_tank[0].cur_h+g_h0)-g_tank[1].cur_h)<=0)
        flow_direction = -1;
else
        flow_direction =  1;
temp = (float)(fabs((g_tank[0].cur_h+g_h0)-g_tank[1].cur_h));
if ( g_tank[1].cur_h > g_hf )
        g_f2   =  flow_direction * KVC(0,p1)*sqrt(temp);
else
g_f2    =  KVC(0,p1)*sqrt(g_tank[0].cur_h);

/* ODE. */
dh1     =  (g_f1 + Disturb - g_f2)*g_dt/a1;
g_tank[0].cur_h    = g_tank[0].cur_h + dh1;
if (g_tank[0].cur_h < 0)                    g_tank[0].cur_h  = 0;
```

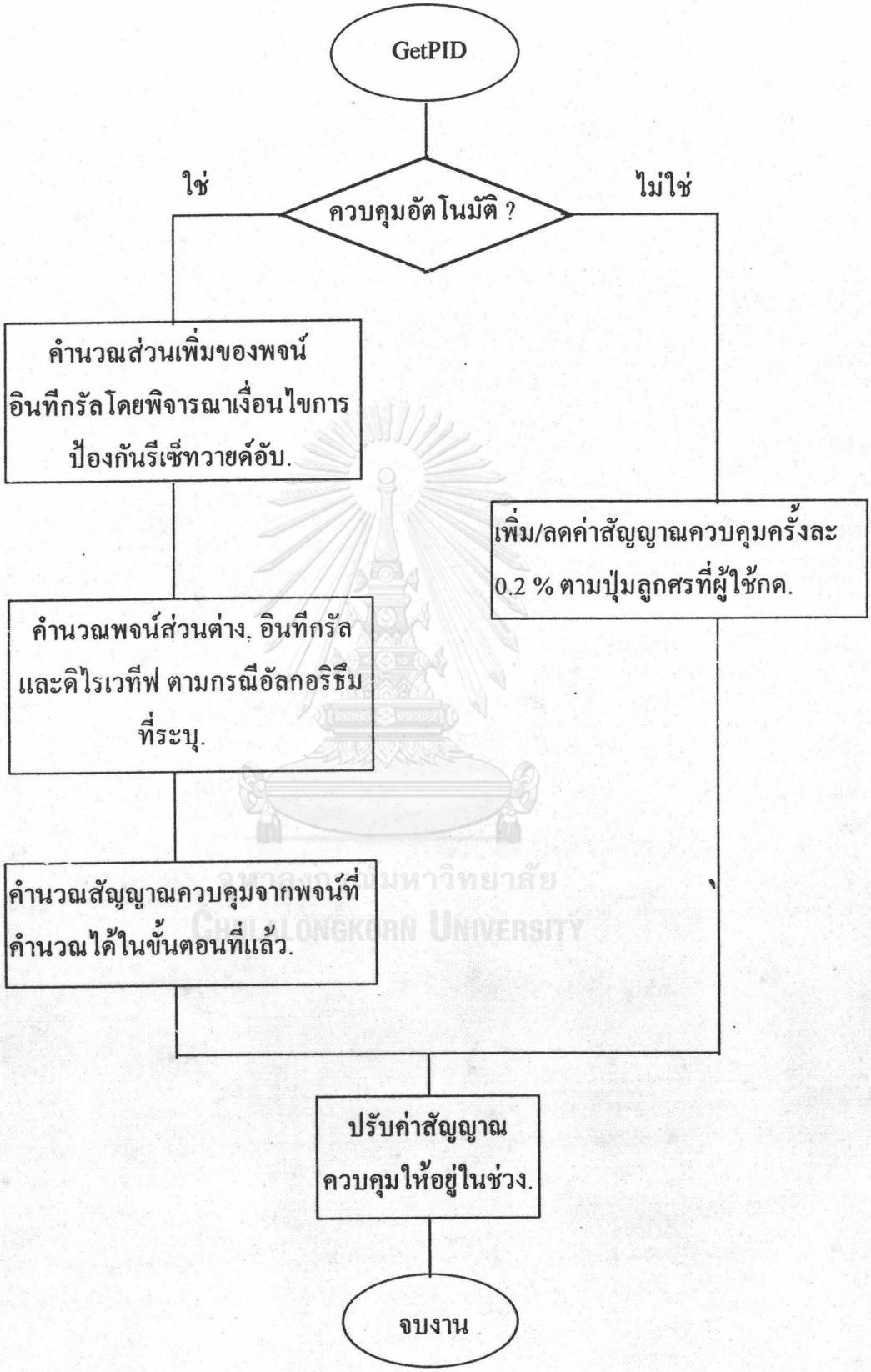
```
if (g_tank[0].cur_h > g_tank[0].h)        g_tank[0].cur_h  = g_tank[0].h;

/***************/
/*** TANK 2 ***/
/***************/

Get_K_R_Td(1,&Kc,&R,&Td);
co2.t    =  GetPID(1,Kc,R,Td);
p2       =  1-co2.t;
g_mvt[1]  = p2;
g_f3     =  KVC(1,p2)*sqrt(g_tank[1].cur_h);

/* ODE. */
dh2      =  (g_f2 - g_f3)*g_dt/a2;
g_tank[1].cur_h    = g_tank[1].cur_h + dh2;
if (g_tank[1].cur_h < 0)                  g_tank[1].cur_h  = 0;
if (g_tank[1].cur_h > g_tank[1].h)        g_tank[1].cur_h  = g_tank[1].h;

/*****************************/
/*** TANK 3 (one tank model) ***/
/*****************************/

Get_K_R_Td(2,&Kc,&R,&Td);
co3.t    =  GetPID(2,Kc,R,Td);
p3       =  1-co3.t;
g_mvt[2]  =  p3;
g_f4     =  KVC(2,p3)*sqrt(g_tank[2].cur_h);
```

```
        /* ODE. */
        dh3     =  (g_f1 + Disturb - g_f4)*g_dt/a3;
        g_tank[2].cur_h    = g_tank[2].cur_h + dh3;
        if (g_tank[2].cur_h < 0)                    g_tank[2].cur_h  = 0;
        if (g_tank[2].cur_h > g_tank[2].h)          g_tank[2].cur_h  = g_tank[2].h;
        g_opr_t  +=  g_dt;
}
```

ก.2 ฟังก์ชัน GetPID

ฟังก์ชันนี้จะผ่านค่าพารามิเตอร์ของตัวควบคุมผ่านทางแอดเดรส ตามหมายเลขของตัวควบคุมที่ผ่านค่าเข้ามา.

ก.2.1 โฟลว์ชาร์ทของฟังก์ชัน GetPID



รูปที่ ก.2 โฟลว์ชาร์ทของฟังก์ชัน GetPID.

ก.2.2 อัลกอริธึมของฟังก์ชัน GetPID

```
float GetPID(int lic_no, float Kc, float R, float Td)
{
        float   error,
                integral,
                diff,
                delta_int;
        float   coMax = 1,
                co,
                delta_co = -0.002/coMax;

        g_model[lic_no].error2  = g_model[lic_no].error1;
        g_model[lic_no].error1  = g_model[lic_no].error;
        g_model[lic_no].error   =(g_set[lic_no] - g_tank[lic_no].cur_h)/g_tank[lic_no].h;

        g_model[lic_no].pv2     = g_model[lic_no].pv1;
        g_model[lic_no].pv1     = g_model[lic_no].pv;
        g_model[lic_no].pv      =(g_tank[lic_no].cur_h)/g_tank[lic_no].h;

        if (g_lic[lic_no].mode==0){
                delta_int       = g_model[lic_no].error*g_dt;
                if (g_lic[lic_no].reset==1){
                        if (g_lic[lic_no].co>=coMax || g_lic[lic_no].co<=0){
                                if ( fabs(g_model[lic_no].integral+delta_int)
                                                >g_mode[lic_no].integral)
                                        delta_int = 0;
                        }
                }
```

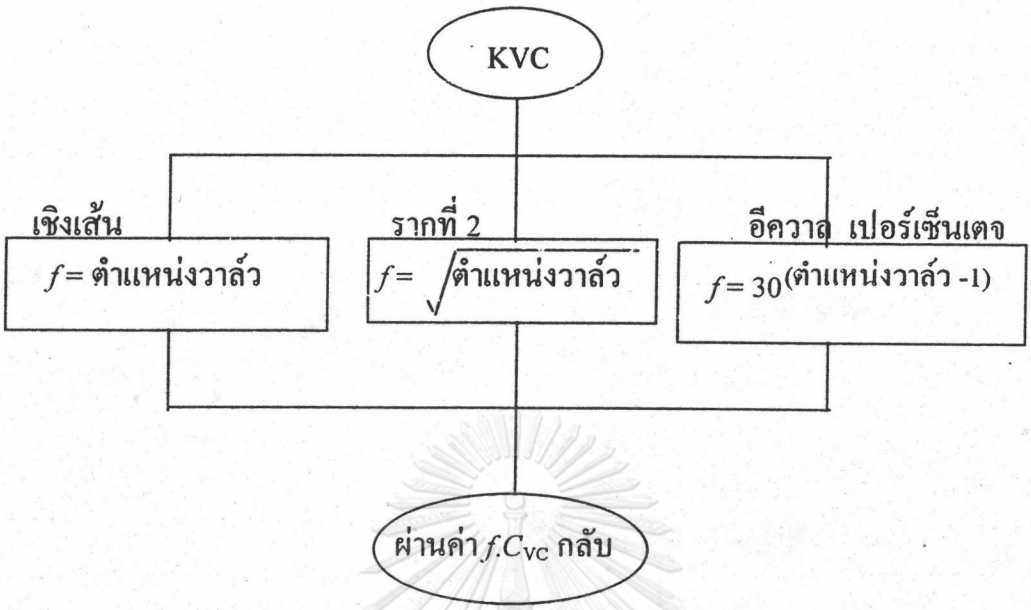
```
switch(g_lic[lic_no].algor){
        case 0:;
                error    = g_model[lic_no].error;
                integral = g_model[lic_no].integral + delta_int;
                diff=(g_model[lic_no].error-g_model[lic_no].error1)/g_dt;
                break;

        case 1:;
                error    = g_model[lic_no].error;
                integral = g_model[lic_no].integral + delta_int;
                diff     =(g_model[lic_no].pv-g_model[lic_no].pv1)/g_dt;
                break;

        case 2:;
                error    = g_model[lic_no].error-g_model[lic_no].error1;
                integral = g_model[lic_no].error*g_dt;
                diff     =(g_model[lic_no].error-2*g_model[lic_no].error1
                                +g_model[lic_no].error2)/g_dt;
                break;

        case 3:;
                error    =  g_model[lic_no].pv1-g_model[lic_no].pv;
                integral =  g_model[lic_no].error*g_dt;
                diff     = (2*g_model[lic_no].error1-g_model[lic_no].error
                        -g_model[lic_no].error2)/g_dt;
                break;

}
```

```
            co = Kc*(error +R*integral +Td*diff)   ;
            if (g_lic[lic_no].algor < 2) co = g_lic[lic_no].co0 +co/coMax;
            else co = g_lic[lic_no].co + co/coMax;

        }
        if (g_lic[lic_no].mode==1){
            co = g_lic[lic_no].co;
            if (g_sim_kbd==72)  co =  co +delta_co;
            if (g_sim_kbd==80)  co =  co -delta_co;
        }
        if (co > coMax)       co = coMax;
        if (co < 0)           co = 0;
        g_lic[lic_no].co        = co;
        g_model[lic_no].integral  = integral;
        return (co);
}
```

ก.3 ฟังก์ชัน KVC

ฟังก์ชันนี้รับหมายเลขของวาล์วควบคุมและตำแหน่งการเปิดปิดวาล์ว และผ่านค่าคงที่ของวาล์วกลับทางหัวฟังก์ชัน.

ก.3.1 โฟลว์ชาร์ทของฟังก์ชัน KVC



รูปที่ ก.3 โฟลว์ชาร์ทของฟังก์ชัน KVC.

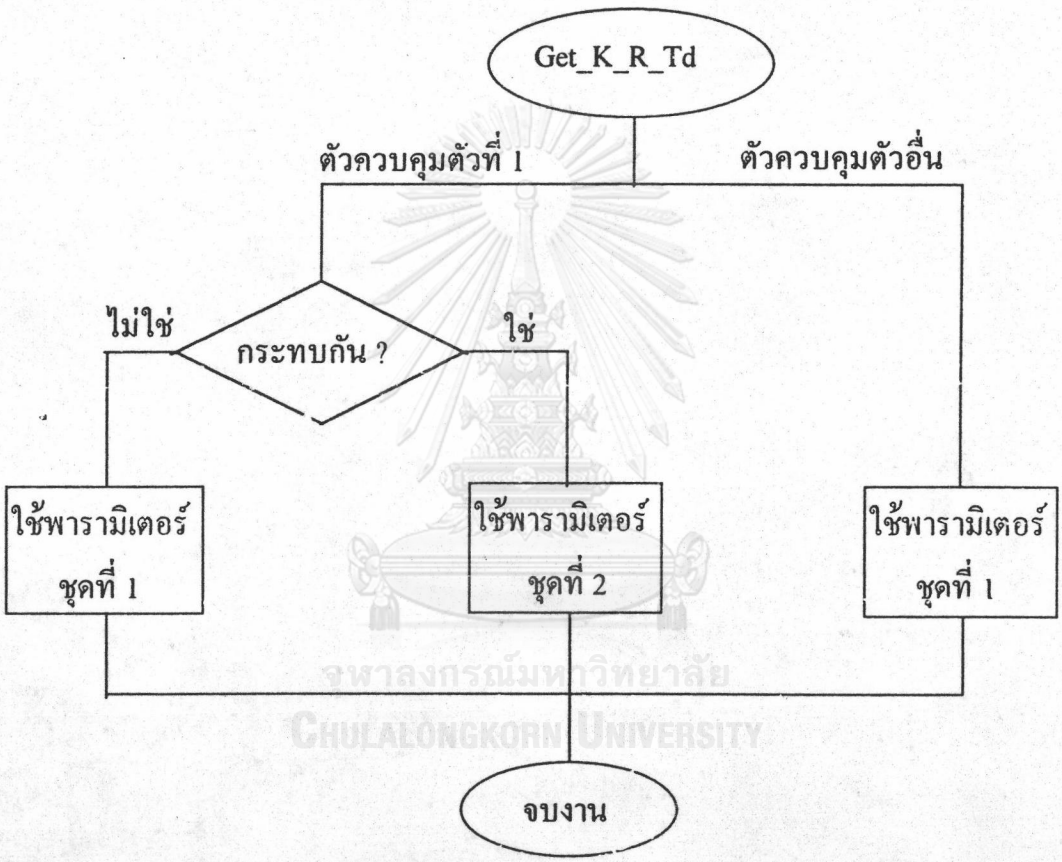
ก.3.2 อัลกอริธึมของฟังก์ชัน KVC

```
float KVC(int valve_no, float valve_position)
    {
        float f;

        if (g_valve[valve_no].f==0) f=valve_position;
        if (g_valve[valve_no].f==1) f=sqrt(valve_position);
        if (g_valve[valve_no].f==2) f=pow(30.0,(valve_position-1.0));
        return(f*g_valve[valve_no].c);
    }
```

ก.4 ฟังก์ชัน Get_K_R_Td

ฟังก์ชันนี้รับการผ่านหมายเลขของตัวควบคุมเข้ามาแล้วตรวจสอบความถูกต้องของพารามิเตอร์ที่ผู้ใช้ตั้งไว้เพื่อผ่านค่ากลับทางแอดเดรสของตัวแปร.

ก.4.1 โฟลว์ชาร์ทของฟังก์ชัน Get_K_R_Td



รูปที่ ก.4 โฟลว์ชาร์ทของฟังก์ชัน Get_K_R_Td.

ก.4.2 อัลกอริธึมของฟังก์ชัน Get_K_R_Td

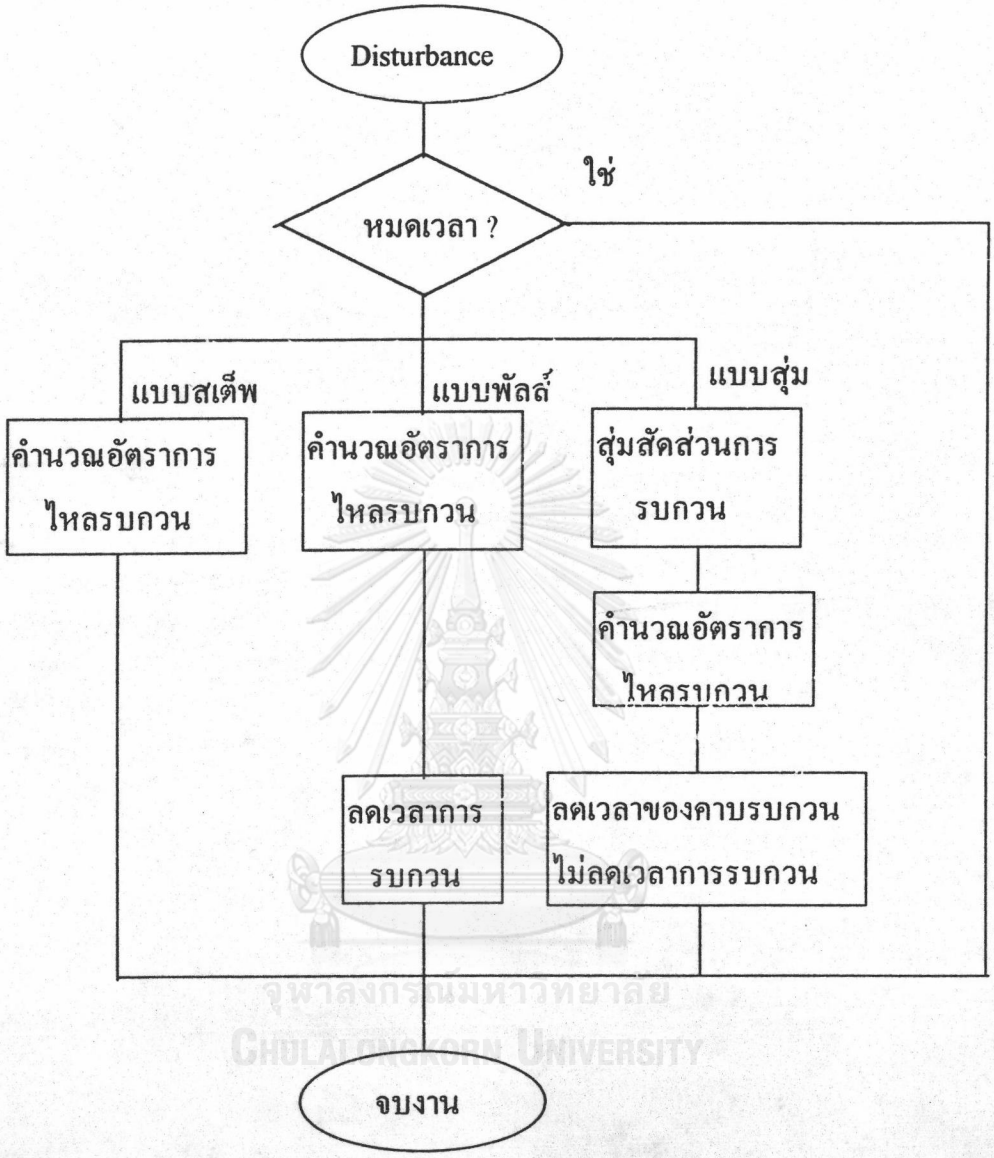
```
void Get_K_R_Td(int lic_no, float *Kc, float *R, float *Td)
{
        if (lic_no==0){
```

```
            if (g_tank[1].cur_h>g_hf){
                if (g_lic[0].pb[1]==0) *Kc = 1000;
                else *Kc = 1/g_lic[0].pb[1]*100;
                *R      = g_lic[0].r[1];
                *Td     = g_lic[0].td[1];
            }
            else{
                if (g_lic[0].pb[0]==0) *Kc = 1000;
                else *Kc = 1/g_lic[0].pb[0]*100;
                *R      = g_lic[0].r[0];
                *Td     = g_lic[0].td[0];
            }
        }
        else{
            if (g_lic[lic_no].pb[0]==0) *Kc = 1000;
            else *Kc = 1/g_lic[lic_no].pb[0]*100;
            *R      = g_lic[lic_no].r[0];
            *Td     = g_lic[lic_no].td[0];
        }
}
```


ก.5 ฟังก์ชัน Disturbance

ฟังก์ชันนี้ประกอบด้วยฟังก์ชันผลิตสัญญาณรบกวน 3 ฟังก์ชัน คือ ฟังก์ชันแบบสเต็พ, ฟังก์ชันแบบพัลส์ และฟังก์ชันแบบสุ่ม ความโตของสัญญาณคิดเทียบกับกระแสที่ป้อนเข้าสู่ถังที่ 1 โดยปั๊มพ์.

ก.5.1 โฟลว์ชาร์ทของฟังก์ชัน Disturbance



รูปที่ ก.5 โฟลว์ชาร์ทของฟังก์ชัน Disturbance.

ก.5.2 อัลกอริธึมของฟังก์ชัน Disturbance

```
float Disturbance(void)
{
        static float disturbance_time  = 999;
        static float disturbance_white = 0;
        float  disturbance_value = 0;
        if (flow_disturb.duration>0 || flow_disturb.type==0){
                switch(flow_disturb.type){
                        case 0 :;
                                disturbance_value = flow_disturb.size/100*g_fl;
                                break;
                        case 1 :;
                                disturbance_value = flow_disturb.size/100*g_fl;
                                flow_disturb.duration -= g_dt;
                                break;
                        case 2 :;
                                if (disturbance_time > flow_disturb.duration){
                                        disturbance_white =
                                                (float)(2*random(flow_disturb.size)
                                                -flow_disturb.size)/100*g_fl;
                                        disturbance_time  = 0;
                                }
                                disturbance_time += g_dt;
                                disturbance_value =  disturbance_white;
                                break;
                }
        }
```

```
    if (flow_disturb.duration < 0) flow_disturb.duration = 0;
    return(disturbance_value);
}
```

ภาคผนวก ข
การจำลองแบบเครื่องปฏิกรณ์ถังกวนแบบต่อเนื่อง

ข.1 คำนำ

จุดประสงค์ข้อหนึ่งของงานวิจัยนี้คือต้องการนำโปรแกรมที่เขียนขึ้นไปใช้ช่วยสอนในวิชาพลวัตและการควบคุมของกระบวนการ ซึ่งผลของงานวิจัยได้สรุปไว้แล้วในบทที่ 6 และเพื่อให้การช่วยสอนทำได้กว้างขวางขึ้น ผู้เขียนจึงจัดทำโปรแกรมเลียนแบบระบบการควบคุมกระบวนการในถังกวนแบบต่อเนื่องโดยมีการควบคุมอุณหภูมิด้วยแจกเก็ต (Jacketed CSTR) เพิ่มขึ้นอีก 1 โปรแกรม. ในวิทยานิพนธ์นี้จะกล่าวถึงกระบวนการของระบบ, สมการพลวัต, และลักษณะหน้าจอของโปรแกรมอย่างย่อๆ เท่านั้น โดยจะขอข้ามส่วนของผลการเลียนแบบระบบไป.

ข.2 กระบวนการของระบบ

ระบบนี้ประกอบด้วยถังปฏิกรณ์เคมีซึ่งเป็นถังปิดป้อนสารเข้าทางด้านบนตลอดเวลา. สารเคมีจะทำปฏิกิริยาแบบอันดับ 1 ดังสมการที่ ข.1 และจะถ่ายสารผลิตภัณฑ์ออกทางด้านล่าง.

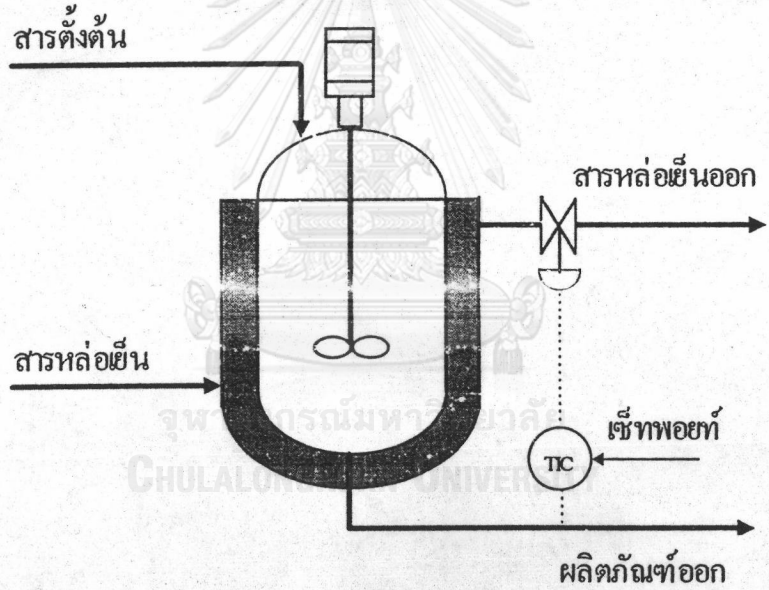
$$A \rightleftharpoons B \qquad \text{(ข.1)}$$

กฏิกิริยาเคมีเป็นแบบคายความร้อนหรือดูดความร้อนก็ได้แล้วแต่ผู้ใช้จะกำหนดค่าความร้อนของปฏิกิริยาในขณะใช้งาน.

ความร้อนในถังปฏิกรณ์ถ่ายเทกับภายนอกด้วยสารหล่อเย็นหรือสารให้ความร้อนแล้วแต่กรณี. สารถ่ายเทความร้อนจะผ่านเข้ามาทางแจกเก็ตซึ่งหุ้มอยู่รอบถัง. อัตราการไหลของสารตั้งต้นจะคงที่ในขณะที่อัตราการไหลของสารหล่อเย็นถูกควบคุมด้วยวาล์วควบคุมซึ่งได้รับสัญญาณควบคุมจากตัวควบคุม.

ตัวควบคุมนำค่าของตัวแปรกระบวนการ ซึ่งในที่นี้คืออุณหภูมิของผลิตภัณฑ์ มาคำนวณสัญญาณควบคุมด้วยอัลกอริธึมที่กล่าวในบทที่ 2 และบทที่ 4 แล้วส่งสัญญาณควบคุมไปควบคุมวาล์วควบคุมเพื่อควบคุมอัตราการไหลของสารหล่อเย็นที่ไหลผ่านแจกเก็ต.

ภาพไดอะแกรมของกระบวนการแสดงในรูปที่ ข.1.



รูปที่ ข.1 ภาพไดอะแกรมของระบบ.

ข.3 สมการจำลองกระบวนการ

ในกระบวนการนี้จะสามารถจำลองได้ด้วยสมการอนุรักษ์มวลของสารตั้งต้น, สมการอนุรักษ์พลังงานในถังปฏิกรณ์, สมการอนุรักษ์พลังงานในแจกเก็ต และสมการค่าคงที่ในการเกิดปฏิกิริยาเคมี. ดังจะกล่าวต่อไปนี้.

สมการอนุรักษ์มวลของสารตั้งต้น

$$\frac{dC_A}{dt} = \frac{F}{V}\left(C_{A0} - C_A\right) - kC_A \qquad (ข.2)$$

เมื่อ C_A คือ ความเข้มข้นของสารตั้งต้นในถังปฏิกรณ์ที่เวลาใดๆ

C_{A0} ความเข้มข้นของสารตั้งต้นในถังปฏิกรณ์ที่เวลาใดๆ

F อัตราการไหลเชิงปริมาตรของสารตั้งต้นที่ป้อน

V ปริมาตรของถังปฏิกรณ์

k ค่าคงที่ของการเกิดปฏิกิริยาเคมี

สมการอนุรักษ์พลังงานในถังปฏิกรณ์

$$\frac{dT}{dt} = \frac{F}{V}\left(T_0 - T\right) - \frac{\Delta H_R}{\rho C_p}kC_A - \frac{UA}{V\rho C_p}\left(T - T_J\right) \qquad (ข.3)$$

เมื่อ T คือ อุณหภูมิของสารภายในถังปฏิกรณ์

T_0 อุณภูมิของสารตั้งต้นที่ป้อน (กระแสป้อน)

ΔH_R ความร้อนต่อโมลของการเกิดปฏิกิริยา

C_P ความจุความร้อนต่อโมลของกระแสป้อน

ρ ความหนาแน่นของกระแสป้อน

U ค่าสัมประสิทธิ์การถ่ายเทความร้อนรวมของแจกเก็ตกับถังปฏิกรณ์

A พื้นที่ของการถ่ายเทความร้อน

T_J อุณหภูมิของสารหล่อเย็นในแจกเก็ต

สมการอนุรักษ์พลังงานในแจกเก็ต

$$\frac{dT_J}{dt} = \frac{UA}{V_J \rho_J C_{pJ}}\left(T - T_J\right) + \frac{F_J}{V_J}\left(T_{J0} - T_J\right) \qquad \text{(ข.4)}$$

เมื่อ V_J คือ ปริมาตรของแจกเก็ต

F_J อัตราการไหลเชิงปริมาตรของสารหล่อเย็นผ่านแจกเก็ต

ρ_J ความหนาแน่นของสารหล่อเย็น

C_{pJ} ความจุความร้อนของสารหล่อเย็น

T_{J0} อุณหภูมิของสารหล่อเย็นที่ป้อน

สมการค่าคงที่ของการเกิดปฏิกิริยาเคมี

$$k = k_0 e^{-E/RT} \qquad \text{(ข.5)}$$

เมื่อ k_0 คือ ค่าคงที่มาตรฐานของการเกิดปฏิกิริยาที่ศึกษา

e ค่าเอ็กซ์โปเนนเชี่ยล

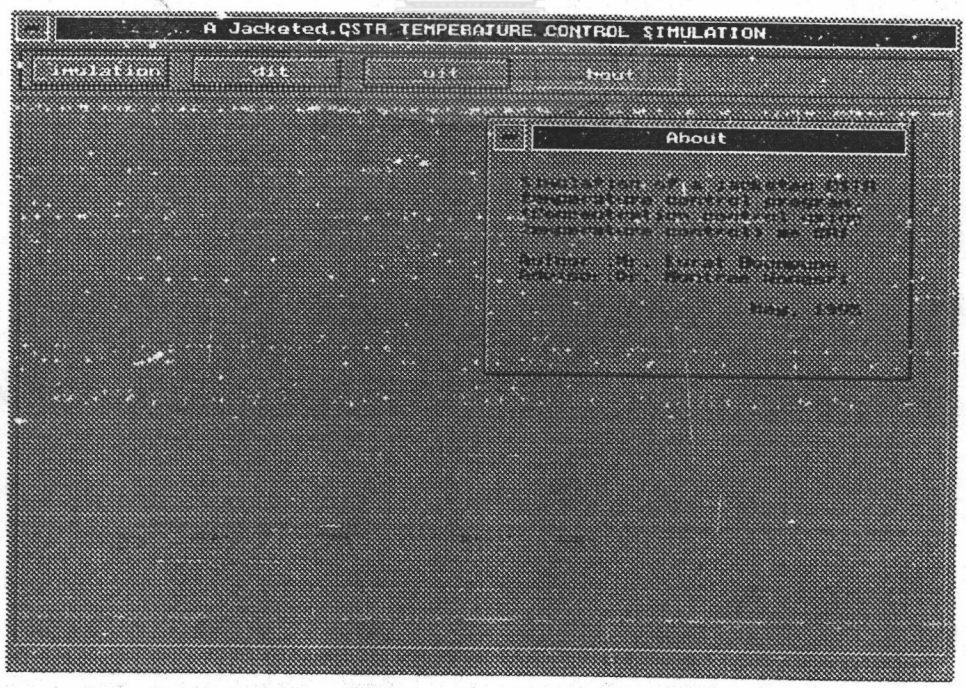
E ค่าพลังงานกระตุ้น

R ค่าคงที่ของแก๊ส

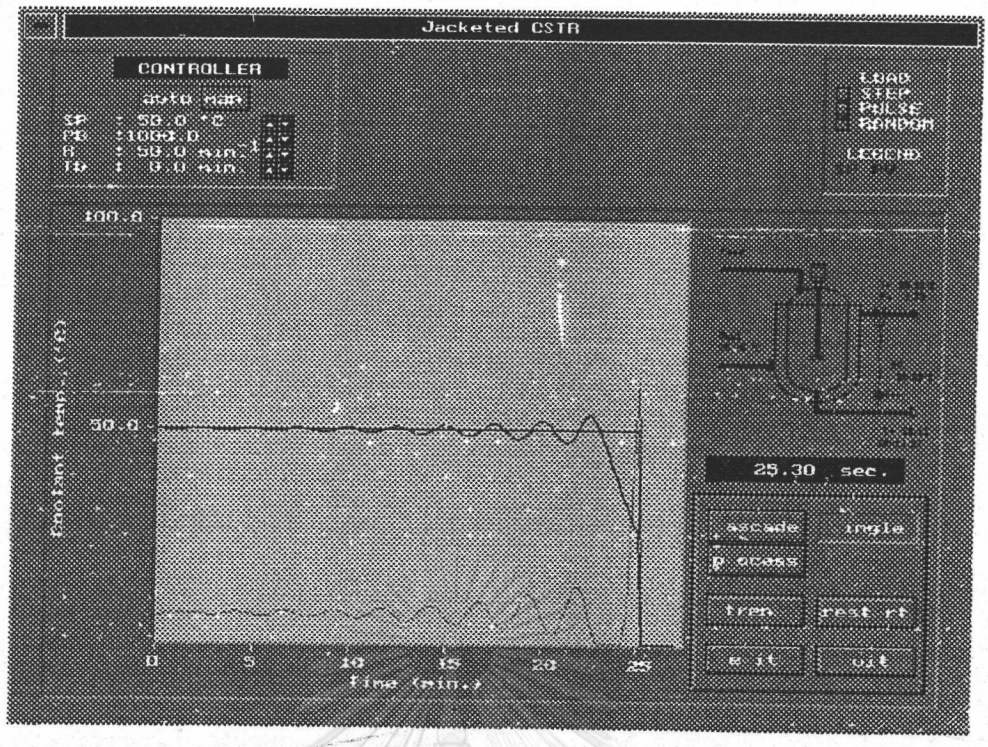
สมการที่ ข.2 ถึง ข.4 เป็นสมการที่ได้รับการจัดแต่งมาแล้วซึ่งอาจทำให้ผู้อ่านเข้าใจยากอยู่บ้าง. อย่างไรก็ดี สมการเหล่านี้ได้จากสมการอนุรักษ์มวลดังกล่าวแล้วในบทที่ 2 และสมการอนุรักษ์พลังงานซึ่งมีวิธีเขียนในทำนองเดียวกันกับสมการอนุรักษ์มวล.

ข.4 หน้าจอของโปรแกรม

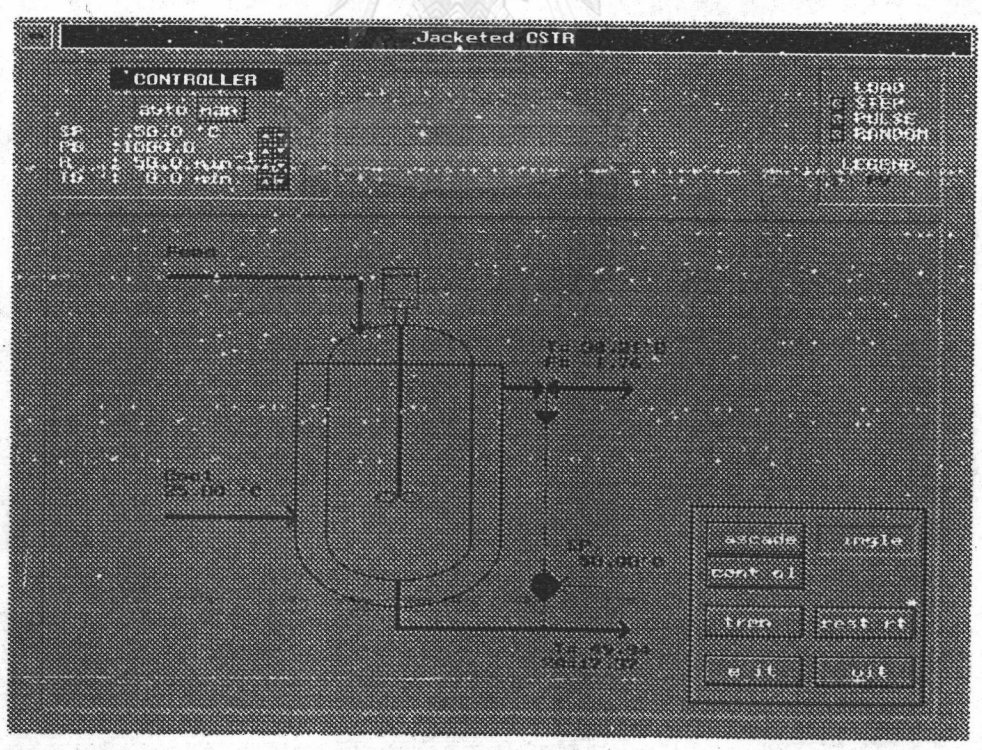
หน้าจอของโปรแกรมและส่วนติดต่อกับผู้ใช้ซึ่งรวมถึงการแก้ไขค่าพารามิเตอร์ต่างๆ ของโปรแกรมนี้จะเหมือนกับ TANKSIM ฉะนั้นผู้เขียนจึงข้ามการกล่าวถึงเสีย. และหากผู้อ่านมีโอกาสใช้โปรแกรมนี้ก็จะสามารถศึกษาผลตอบของการเลียนแบบระบบได้. ในส่วนนี้ผู้เขียนขอแสดงหน้าจอบางหน้าเพื่อเป็นการแนะนำโปรแกรมเท่านั้น.



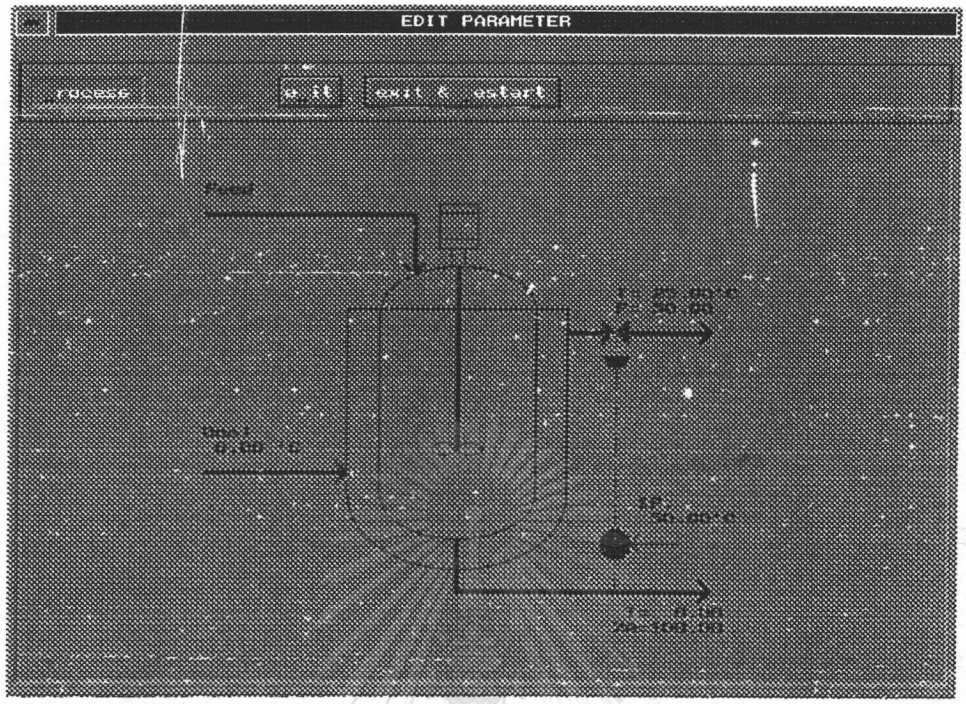
รูปที่ ข.2 หน้าจอหลักของโปรแกรมเมื่อกดปุ่ม 'About'.



รูปที่ ข.3 หน้าจอการเลียนแบบระบบและแก้ไขพารามิเตอร์ของตัวควบคุมแสดงกราฟผลตอบ.



รูปที่ ข.4 รูปไดอะแกรมของกระบวนการ แสดงผลตอบเป็นตัวเลข.



รูปที่ ข.5 หน้าจอแก้ไขพารามิเตอร์ทั้งหมดของระบบ.

ประวัติผู้เขียน

นายสุรัตน์ บุญพึ่ง เกิดเมื่อวันที่ 18 พฤศจิกายน พ.ศ. 2502 สำเร็จการศึกษาในระดับชั้นมัธยมศึกษาปีที่ 5 จากโรงเรียนอยุธยาวิทยาลัย. สำเร็จการศึกษา ปริญญาวิศวกรรมศาสตรบัณฑิต สาขา วิศวกรรมเคมี จากมหาวิทยาลัยสงขลานครินทร์ เมื่อ พ.ศ. 2527.เข้ารับราชการในแผนกวิชาเคมีอุตสาหการ คณะวิชาเทคนิคเคมี สถาบันเทคโนโลยีราชมงคล วิทยาเขตเทคนิคกรุงเทพฯ. เมื่อ พ.ศ. 2529 ปัจจุบันดำรงตำแหน่งอาจารย์ 1 ระดับ 4.