The intersections options for second run when the robot encounter them are determined, so based on the Maze diagram at cell-2 is R-turn, cell-8 is L-turn, cell-14 is L-turn, cell-21 is L-turn, cell-20 is F-turn.

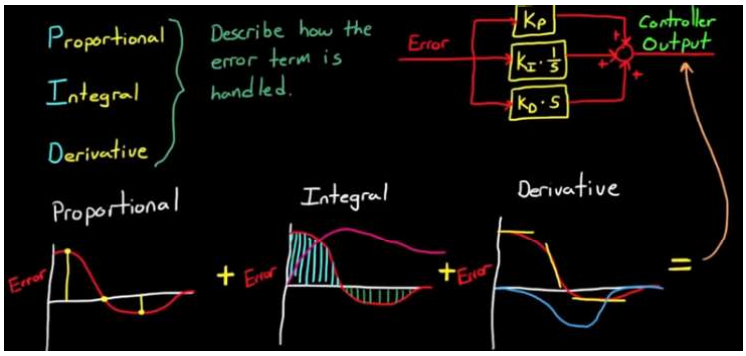
Software Code development:

The software start by reading distance value from the three sensors. When the robot reach an intersections it uses the DFS algorithm to decide which way to go. The robot stops when it reaches the goal and start the second run. If not it keeps reading the sensor and following the walls.

PID control is used to control the movement of the robot to stay in the middle of the lane. The setpoint is zero. The error is the different of right sensor and left sensor. Proportional control as Kp compensate the error either to the right or the left, derivative control al Kd predicts the trend of the error by subtracting current error from previous error, and integrating control Ki erase the steady error.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)\,dt + K_d \frac{de(t)}{dt}$$

Output = Proportional + Integral + Derivative



Code snippet for PID control:

```
void myPIDcontrol (double Kpp, double Kii, double Kdd) {
  unsigned long speedA=10000, speedB=10000;
  unsigned char setPoint = 0;
  signed long Ep=0, Ed=0, Ei=0, lastError=0;
  Ep = setPoint - (distance1-distance2);
  Ep = Ep;
  Ed = Ep - lastError;
  if ((Ep>-30)&&(Ep<30)) Ei = Ei + Ep; //25
  if (Ep==0) Ei = 0;
//  if(Ep>0) GPIO_PORTF_DATA_R = 0x08;
//  else if (Ep<0) GPIO_PORTF_DATA_R = 0x0C;
  lastError = Ep;
  speedA = speedA - (Ep)*(Kpp) - (Ed)*(Kdd);
  speedB = speedB + (Ep)*(Kpp) + (Ed)*(Kdd);
  if (speedA>39000) speedA=39000; //40,000 == 100%
  if (speedB<100)  speedB=100;    //40,000 == 100%
  if (speedB>39000) speedB=39000; //40,000 == 100%
  if (speedA<100)  speedA=100;    //40,000 == 100%
  motorA(speedA);
  motorB(speedB);
}
```

Complete Software Code:

```
//standard C library with <>
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <limits.h>

#include "UART.h"
#include "PLL.h"
#include "PWM.h"
#include "SysTick.h"
#include "Timer0.h"
#include "Timer1.h"
#include "Timer2.h"
#include "tm4c123gh6pm.h"

unsigned long distance0=0, distance1=0, distance2=0;
unsigned long count0=0, count1=0, count2=0;

long StartCritical (void);    // previous I bit, disable interrupts
void EndCritical(long sr);    // restore I bit to previous value

void WaitForInterrupt(void);
void DisableInterrupts(void);
void EnableInterrupts(void);
void PortF_Init(void);
void PortE_Init(void);
void PortA_Init(void);
void PortB_Init(void);
void PortC_Init(void);
void PortD_Init(void);

void timer0FrontDistance0(unsigned long *distance0);
void timer1RightDistance1_1(unsigned long *distance1);
void timer2LeftDistance2_2(unsigned long *distance2);
void stirrRight(void);
```

```
void stirrRight(void);
void stirrLeft(void);
void stop(void);

void UserTask(void){    //Timer0
  if (GPIO_PORTE_DATA_R&0x10)
    count0 = count0 + 1;
}

void UserTask1(void){   //Timer1
  if (GPIO_PORTD_DATA_R&0x04)
    count1 = count1 + 1;
}

void UserTask2(void){   //Timer2
  if (GPIO_PORTE_DATA_R&0x04)
    count2 = count2 + 1;
}
void OutCRLF(void){
  UART_OutChar(CR);
  UART_OutChar(LF);
}
```

```
int main (void) {
  unsigned long index = 0, i = 0, ii = 0;
  unsigned char tt = 0, t_index = 0,stp = 0;
  unsigned char r=0, l=0, f=0;
  unsigned char DFS_1stRun;
  unsigned char turn = 0, deadEnd=0, goal=0;
  unsigned char arrayTurns[4] = {0, 0, 0, 0};
  unsigned char travalArray[250];
  unsigned char secndRunArray[250];

  unsigned char tempArray[33] = {'F',0x03,'R','R','U',0x03,'F',0x06,
                                 0x06,'F','U',0x06,'R','R','U',0x06,
                                 0x06,'R','R','R',0x05,'R','R','U',
                                 0x05,'F','R',0x03,'R','U',0x03,0x05,
                                 'R'};
  PLL_Init();                   // bus clock at 80 MHz
  PortF_Init();
  PortE_Init();
  PortA_Init();
  PortB_Init();
  PortC_Init();
  PortD_Init();
  SysTick_Init();
  UART_Init();
  Timer0_Init(&UserTask,80);   // 1us interrupt
  Timer1_Init(&UserTask1,80);  // 1us interrupt
  Timer2_Init(&UserTask2,80);  // 1us interrupt
  PWM0A_Init(40000,15000);     // initialize PWM0
  PWM0B_Init(40000,15000);     // initialize PWM0

  GPIO_PORTF_DATA_R = 0x04;    // Blue led
  GPIO_PORTB_DATA_R |= 0x01;   // MotorA foward
  GPIO_PORTB_DATA_R |= 0x02;   // MotorB foward
  PWM0A_Duty(1000);
  PWM0B_Duty(1000);

  stop();
  GPIO_PORTF_DATA_R = 0x04;    // Blue led

  for (unsigned char t=0; t<250; t++)
    travalArray[t] = 0;
  for (unsigned char s=0; s<250; s++)
    secndRunArray[s] = 0;

  UART_OutString("InString---> Distance is: ");
  OutCRLF();

  while (goal == 0) {
    while (turn != 'U') {
      while (turn == 0) {

        timer0FrontDistance0(&distance0);
        timer1RightDistance1_1(&distance1);
        timer2LeftDistance2_2(&distance2);

////Test Distance Sensor
        while(1) {
          GPIO_PORTF_DATA_R = 0x02;     //RED led, Timer0
          GPIO_PORTB_DATA_R |= 0x01;    //MotorA, foward
          GPIO_PORTB_DATA_R |= 0x02;    //MotorB, forward
          PWM0A_Duty(35000);            //10%   MotorA, foward
          PWM0B_Duty(35000);            //10%   MotorB, foward
          //Timer0 Port PE4-5, distance0
          timer0FrontDistance0(&distance0);
          //Timer1 Port PD2-3, distance1, 1
          timer1RightDistance1_1(&distance1);
          //Timer2 Port PE2-3, distance2, 2
          timer2LeftDistance2_2(&distance2);
```

```
129             UART_OutString("Front sensor:");
130             UART_OutUDec(distance0);
131             UART_OutString(",  Right sensor:");
132             UART_OutUDec(distance1);
133             UART_OutString(",  Left sensor:");
134             UART_OutUDec(distance2);
135             UART_OutString("  mm");
136             OutCRLF();
137             SysTick_Wait10ms(10);
138           }
139
140           GPIO_PORTF_DATA_R = 0x00;
141           if (t_index == 33)
142             turn = 'G';
143           else {
144             turn = tempArray[t_index++];
145             if ((turn > 0x0F)&&(turn != 'U')) {
146               turn = 0;
147             }
148           }
149           if (turn == 'G') {
150             goal = 1;
151             turn = 'U';
152           }
153           else if (turn == 'U') {
154             deadEnd = 1;
155             turn= 0;
156           }
157           UART_OutString("****turn: ");
158           UART_OutUDec(turn);
159           OutCRLF();
160         } ////END of while(turn == 0)
161
162         if ((deadEnd == 1)&&(goal == 0)){
163           turn = 'U';
164           goal = 0;
165         }

166       else if ((deadEnd == 0)&&(goal == 0)){
167         if (turn == 0x03){////***turn RF == 0x03
168           travalArray[index++] = 0x03;
169           travalArray[index++] = 'R';
170           travalArray[index++] = 'F';
171           turn = 'F';
172           turn = 0;
173           f = 1;
174         }
175         else if (turn == 0x05) {/////***turn LF ==0x05
176           travalArray[index++] = 0x05;
177           travalArray[index++] = 'L';
178           travalArray[index++] = 'F';
179           turn = 'F';
180           turn = 0;
181           f = 1;
182         }
183         else if (turn == 0x06) {/////***turn LR == 0x06
184           travalArray[index++] = 0x06;
185           travalArray[index++] = 'L';
186           travalArray[index++] = 'R';
187           turn = 'R';
188           r = 1;
189           turn = 0;
190         }
191         else if (turn == 0x07) {////***LRF == 0x07
192           travalArray[index++] = 0x07;
193           travalArray[index++] = 'L';
194           travalArray[index++] = 'R';
195           travalArray[index++] = 'F';
196           turn = 'F';
197           f = 1;
198           turn = 0;
199         }
200       }
201     }
202     /////END OF:"while(turn!='U')"; now turn == 'U', goal==0;
```

```
204       ////deadEnd==1; goal==0; turn=='U'; go to TOP STACK, get the 2ND TURN
205       if (goal == 0) {
206         while (turn == 'U'){
207           deadEnd=0;
208           DFS_1stRun = 'U';
209           for (i=0; i<4; i++)
210             arrayTurns[i] = 0;
211           i = 0;
212
213           ////array that hold the turns:0x03,0x05,0x06,0x07
214           if ((travalArray[index]) == 0)
215             index--;
216
217           /*////pop off the TOP stack and save values to array[4];
218           ////0x07==LRF--->[0]=='F', [1]=='R', [2]=='L',[3]==0x07
219           ////0x06==LR--->[0]=='R', [1]=='L', [2]== 0, [3]==0x06
220           ////0x05==LF--->[0]=='F', [1]=='L', [2]== 0, [3]==0x05
221           ////0x03=RF---->[0]=='F', [1]=='R', [2]== 0, [3]==0x03*/
222           while (((DFS_1stRun>0x0F)&&(index>0))||
223                 ((DFS_1stRun>0x0F)&&(index==0))) {
224             DFS_1stRun = travalArray[index];
225             travalArray[index--] = 0;
226             if (DFS_1stRun > 0x0F){
227               arrayTurns[i] = DFS_1stRun;
228               i++;
229             }
230           }
231           arrayTurns[3] = DFS_1stRun;
232
233           ////RF,LF=0x03,0x05.....................................................
234           if ((arrayTurns[3] == 0x03)||(arrayTurns[3]==0x05)) {
235             if ((arrayTurns[0]=='F')&&(arrayTurns[1]=='R')) {
236               deadEnd = 0;
237               turn = 'L';
238               index++;
239               travalArray[index++] = arrayTurns[3]; //push 0x03
240               travalArray[index++] = arrayTurns[1]; //push 'R'
241             }
242             else if ((arrayTurns[0]=='F')&&(arrayTurns[1]=='L')){
243               deadEnd = 0;
244               turn = 'R';
245               index++;
246               travalArray[index++] = arrayTurns[3]; //push 0x05
247               travalArray[index++] = arrayTurns[1]; //push 'L'
248             }

249           /*////2ND TURN, deadEnd = 1;
250           ////0x05==LF--->[0]=='L',[1]==0,[2]== 0,[3]==0x05
251           ////0x03=RF---->[0]=='R',[1]==0,[2]== 0,[3]==0x03 */
252           else if ((arrayTurns[0]=='R')||(arrayTurns[0]=='L')){
253             deadEnd = 1;
254             if (arrayTurns[0]=='R')
255               turn = 'L';
256             else if (arrayTurns[0]=='L')
257               turn = 'R';
258           }
259         }
260         ////END OF: 0x03==RF, 0x05==LF.....................................
261
262         ////RL=0x06.....................................................
263         else if (arrayTurns[3]==0x06) {
264           ////printing travalArray**************************************
265           if (arrayTurns[0]=='R') {
266             deadEnd = 0;
267             turn = 'F';
268             index++;
269             travalArray[index++] = arrayTurns[3]; //push 0x06
270             travalArray[index++] = arrayTurns[1]; //push 'L'
271           }
272           else if (arrayTurns[0]=='L') {
273             deadEnd = 1;
274             turn = 'R';
275           }
276         }
277         ////END OF: 0x06==LR.....................................
```

```
278         if (turn == 'R'){
279           r = 1;
280           turn = 0;
281         }
282         else if (turn == 'L'){
283           l = 1;
284           turn = 0;
285         }
286         else if (turn == 'F'){
287           f = 1;
288           turn = 0;
289         }
290       }////END OF:"while(turn=='U')"; now turn == 0, turn != 'U'
291     }////END OF if(goal==0)
292   }////END OF  while(goal == 0), Reach the GOAL,  goal == 1
293
294   UART_OutString("goal: ");
295   UART_OutUDec(goal);
296   OutCRLF();
297
298   i = 0; ii = 0;
299   while (travalArray[i] != 0){
300     if ((i>0)&&(travalArray[i-1]>0x0F)&&
301         (travalArray[i]>0x0F)) {
302       secndRunArray[ii-1] = travalArray[i];
303     }
304     else {
305       secndRunArray[ii] = travalArray[i];
306       ii++;
307     }
308     i++;
309   }
```

```
310   ii = 0;
311   while(secndRunArray[ii] != 0) {
312     UART_OutString("secndAry:");
313     if (secndRunArray[ii] == 70)
314       UART_OutString(" F");
315     else if (secndRunArray[ii] == 82)
316       UART_OutString(" R");
317     else if (secndRunArray[ii] == 76)
318       UART_OutString(" L");
319     else
320       UART_OutUDec(secndRunArray[ii]);
321     OutCRLF();
322     ii++;
323   }
324   ii = (ii+1)/2;
325   OutCRLF();
326   UART_OutString("number of Turns secndArray: ");
327   UART_OutUDec(i);
328   UART_OutString("---");
329   UART_OutUDec(ii);
330   OutCRLF();
331   if (goal == 1){
332     stop();
333   }
334 }
335 //////END OF Main**********
```

```
336 /*
337  goal: 1
338  secndAry:3
339  secndAry: R
340  secndAry:6
341  secndAry: L
342  secndAry:5
343  secndAry: L
344  secndAry:3
345  secndAry: R
346  secndAry:5
347  secndAry: F
348 */
349 void stop(void) {
358 void stirrLeft(void) {
447 void stirrRight(void) {
532  //Timer0 Port PE4-5, distance0
533 void timer0FrontDistance0(unsigned long *distance0) {
549  //Timer1 Port PD2-3, distance1, distance11, 1, 11
550 void timer1RightDistance1_1(unsigned long *distance1) {
566  //Timer2 Port PE2-3, distance2, distance22, 2, 22
567 void timer2LeftDistance2_2(unsigned long *distance2) {
584 void PortF_Init(void){
605  //PB6, PB7 is PWM, PB0, PB1 is Direction (PB6-->PB0, PB7-->PB1)
606 void PortB_Init(void) {
623 void PortC_Init(void){
635 void PortD_Init(void){
647 void PortE_Init(void){
666 void PortA_Init(void){
```

Reference:

Timer Example: http://users.ece.utexas.edu/~valvano/arm/Timer0A.c

http://users.ece.utexas.edu/~valvano/index.html

http://users.ece.utexas.edu/~valvano/arm/