C 프로그래밍

과제 3

- 글자 인식 로봇 -

담당 교수	:	차의영 교수님	
학과	:	기계공학부	기계공학부
학번	:	200721338	200721395
이름	:	정상효	한경수
과제 제출일	:	2011/04/23	

1 Mission

가) Light Sensor 를 이용하여 종이에 쓰인 숫자를 인식하는 과제이다.

나) Light Sensor 는 밝기에 따라 1 부터 100 까지의 값을 가진다. 글자의 어두운 부분을 센서가 읽으면 낮은 값이 출력되고, 인쇄되지 않은 하얀 부분을 지나면 높은 수가 출력된다.

2 해결방안 모색

가) Light Sensor 를 이용한 글자 인식에 대해 생각해 본 결과, 센서를 통해 들어온 값 중에 최소값 = min, 최대값 = max 으로 잡고 avg = (min+max)/2 를 기준으로 avg 보다 작으면 인쇄된 문자이고, 크면 배경으로 하는것으로 생각하여 프로그래밍 하였다.
NXT 의 화면 해상도는 일일히 점을 찍어가며 테스트 해본 결과 다음과 같았다.

1. 0 <= rows <= 64

2. 0 <= width <= 99

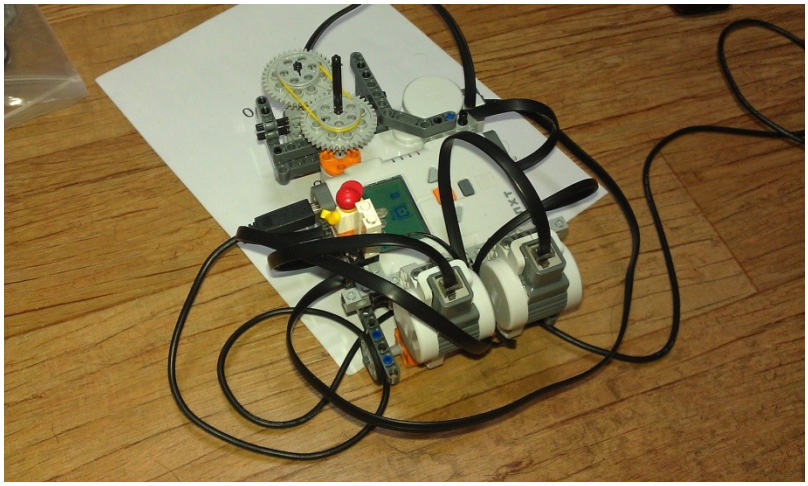
나) 따라서 char map[65][100]; 전역 변수를 잡고, 센서로 스캐닝 하면서 글자가 있는 부분은 값을 1 로 주고, 글자가 없는 부분은 0 이고, 스캔을 하지 않은 부분은 -1 로 세팅을 한다. 스캐닝을 시작하는 버튼은 안드로이드 폰에서 전송되는 bluetooth 9 번 신호이다.

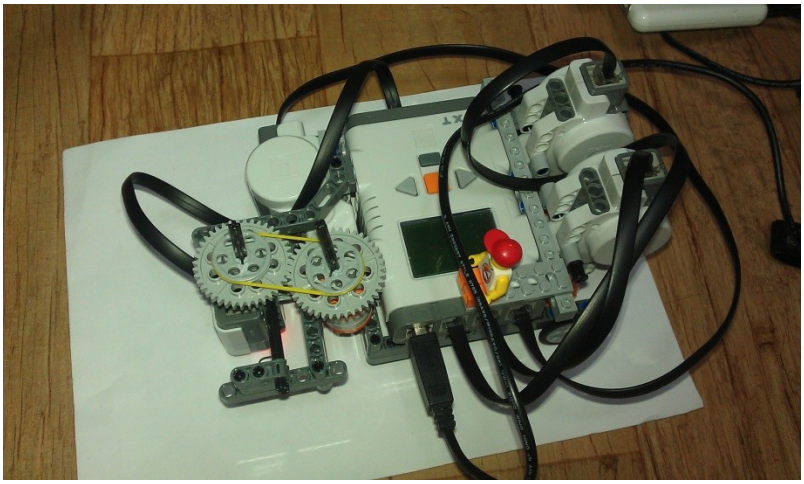
다) 스캐닝을 하기 위한 센서의 이동은 왼쪽에서 오른쪽으로 일직선으로 가게 하였다. 오른쪽으로 센서를 이동시키며 읽어들인 뒤, 센서가 오른쪽 끝까지 이동이 되었으면 다시 왼쪽으로 당겨준다. 처음에는 ㄹ자처럼 오른쪽으로 읽어들이고 전진하고, 왼쪽으로 읽어들이고 전진하는 방법으로 왔다 갔다 하면서 스캔을 하는 방식으로 했었는데, 제대로 수행이 되지 않아 왼쪽에서 오른쪽으로 가는 부분만 스캔 하기로 했다. 스캔이 끝난 후에 displaymap() 함수가 호출되는데, 이 때 전역 변수 map 에는 sensor 값들이 들어있는데, 최대값, 최소값을 고려하여 동적으로 avg 값을 구한다. 그리고 avg 값이 만족스럽지 못한 경우를 대비해서 블루투스 신호 8 번과 10 번을 누르면 임계치를 직접적으로 조절할 수 있게 하였다.

라) 전체적인 그림 인식을 먼저 하고, thinning(세선화) 작업을 하기 위해, 하드웨어가 조립되고 테스트 되는 동안 win32 환경에서 미리 프로그래밍을 해놓았다. 아래는 지금까지 우리 조가 연구했던 thinning 작업의 예시이다.

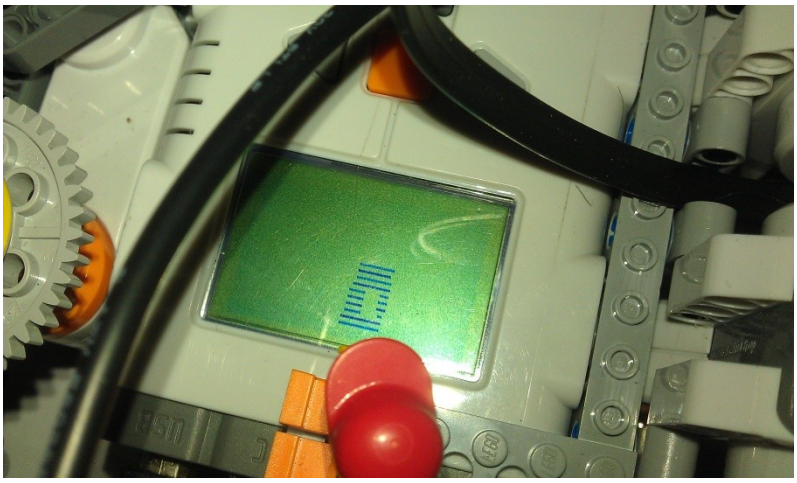
마) 위에서도 언급했다시피 win32 환경하에서 구현한 알고리즘이다, 알고리즘을 미리 구현해놓으면 NXT 에서도 무리없이 적용할 수 있을거라 생각하고 미리 해놨었는데, 지면의 상태가 완전히 평평하지 않으면 센서에서 인식되는 값이 너무 차이가 났고, 정상적으로 인식이 되지 않아서 thinning 알고리즘을 적용하기 무척이나 힘이 들었다.

3 아래는 로봇 사진이다.





위 두 사진은 만든 후 촬영한 사진이다.



0 을 스캔한 후, 나온 결과이다.
센서 리딩을 빠르게 하기 위해 진행을 빠르게 하다보니
가로줄로 0 이 형성된 것을 볼 수 있다.

4 프로그램 소스

가) 다음은 robotc 로 제작한 프로그래밍 소스이다. 처음부터 끝까지 순수 제작했음을 적어두는 바이다.

```
1  #pragma config(Sensor, S3,     LightSensor,          sensorLightActive)
2  //*!!Code automatically generated by 'ROBOTC' configuration wizard
   !!*//

3  short adj = 0;
4  int bDraw = 0;
5  char map[65][100];
6  #pragma platform(NXT)
7  void action(ubyte inp, int length) // 들어오는 입력값에 대해 동작
8  {

9  ubyte act = inp;
10 static int speed = 0; // 기본 속도를 static
11 if(act == 1) // forward
12 {
13 motor[motorC] = -speed;
14 motor[motorB] = -speed;
15 speed += 5; // 누를 때 마다 속도증가
16 }
17 else if(act == 7) // backward
18 {
19 motor[motorB] = speed;
20 motor[motorC] = speed;
21 speed += 5; // 누를 때 마다 속도증가
22 }
23 else if(act == 3)
24 {
25 motor[motorB] = speed;
26 motor[motorC] = -speed;
27 speed += 5; // 누를 때 마다 속도증가
28 }
29 else if(act == 5)
30 {
31 motor[motorB] = -speed;
32 motor[motorC] = +speed;
33 speed += 5; // 누를 때 마다 속도증가
34 }
35 else if(act == 4) // all stop
36 {
37 motor[motorB] = 0;
38 motor[motorC] = 0;
39 motor[motorA] = 0;
40 speed = 1; //전부 스톱시 스피드 1
41 }
42 else if(act == 8) //
43 {
44 adj++;
45 }
46 else if(act == 10)
47 {
48 adj--;
49 }

50 else if(act == 0) //
51 {
52 nMotorEncoder[motorA] = 0;
```

```
53  while(nMotorEncoder[motorA] <= 80)
54  {
55  //eraseDisplay();
56  nxtDisplayCenteredTextLine(0, "Tracking...");
57  motor[motorA] = 5;
58  }
59  motor[motorA] = 0;
60  }
61  else if(act == 2) //
62  {
63  nMotorEncoder[motorA] = 0;
64  while(nMotorEncoder[motorA] >= -70)
65  {
66  //eraseDisplay();
67  nxtDisplayCenteredTextLine(0, "Tracking...");
68  motor[motorA] = -5;
69  }
70  motor[motorA] = 0;
71  }
72  else if(act == 9) //
73  {
74  adj = 0;
75  for(int y=0; y<65; y++)
76  for(int x=0; x<100; x++)
77  map[y][x] = -1;

78  nMotorEncoder[motorA] = 0;
79  nMotorEncoder[motorB] = 0;
80  nMotorEncoder[motorC] = 0;
81  int f = 0;
82  do
83  {
84  static int linenumber = 0;
85  motor[motorA] = -1;
86  while( nMotorEncoder[motorA] > -90)
87  {
88  if( nMotorEncoder[motorA] > 0) continue;
89  nxtDisplayCenteredTextLine((linenumber++)%7, "%d",
    SensorValue(LightSensor));
90  map[-f][-nMotorEncoder[motorA]] = SensorValue(LightSensor);
91  }

92  motor[motorB] = -50;
93  motor[motorC] = -50;
94  while( nMotorEncoder[motorB] > f) ;
95  motor[motorB] = 0;
96  motor[motorC] = 0;
97  f -= 3;
98  nxtDisplayCenteredTextLine(7, "%d", f);
99  if(f == -60)
100 {
101 motor[motorA] = 0;
102 break;
103 }
104 motor[motorA] = 10;
105 while(nMotorEncoder[motorA] <= 0);

106 if(f == -60)
```

```
107 {
108 motor[motorA] = 0;
109 break;
110 }
111 }while(1);
112 bDraw = 1;
113 }
114 }

115 void displaymap()
116 {
117 if(bDraw == 0) return;
118 eraseDisplay(); // 화면을 지우고
119 int y, x;
120 int mymin = 999;
121 int mymax = -1;
122 for(y=0; y<65; y++)
123 {
124 for(x=0; x<100; x++)
125 {
126 if(map[y][x] == -1) continue;
127 if(mymin > map[y][x])
128 mymin = map[y][x];
129 if(mymax < map[y][x])
130 mymax = map[y][x];
131 }
132 }
133 int myavg = (mymin + mymax) / 2;

134 for(y=0; y<65; y++)
135 {
136 for(x =0; x<100; x++)
137 {
138 if(map[y][x] == -1) continue;
139 if(myavg+adj> map[y][x])
140 nxtSetPixel(x, y);
141 }
142 }
143 }
144 void readDataMsg()
145 {
146 const TMailboxIDs kQueueID = mailbox1; // mailbox1 사용
147 const int kMaxSizeOfMessage = 5;
148 TFileIOResult nBTCmdRdErrorStatus;
149 int nSizeOfMessage;
150 ubyte nRcvBuffer[kMaxSizeOfMessage * 5]; // Max 배열크기는 있지만 그냥 적당히
      *5
151 memset(nRcvBuffer, 0, sizeof(nRcvBuffer)); // nRvcBuffer 모두 0 으로 초기화
152 while (1)
153 {
154 nSizeOfMessage = cCmdMessageGetSize(kQueueID); // 받아와야하는 메세지 크기
    를 구함
155 if (nSizeOfMessage == 0) // 없다면 함수에서 나가게 됨
156 {
157 wait1Msec(1);
158 break;
159 }
160 if (nSizeOfMessage > kMaxSizeOfMessage) // 크기가 너무 클 경우엔 지정해준 크기만
```

```
    큼만 받음.
161 nSizeOfMessage = kMaxSizeOfMessage;
162 nBTCmdRdErrorStatus = cCmdMessageRead(nRcvBuffer, nSizeOfMessage,
    kQueueID);
163 if (nBTCmdRdErrorStatus == ioRsltSuccess) // 에러가 나지 않았다면
164 {
165 action(nRcvBuffer[0], kMaxSizeOfMessage * 5); // 받은 패킷을 action 함수로 넘김.
166 displaymap();
167 }
168 else
169 nxtDisplayTextLine(2, "Failed");
170 }
171 }
172 task main()
173 {
174 //nMotorPIDSpeedCtrl[motorB] = mtrSpeedReg; // PID 적용
175 //nMotorPIDSpeedCtrl[motorC] = mtrSpeedReg; // PID 적용
176 while(1)
177 {
178 if (nBTCurrentStreamIndex >= 0) // 접속한 bluetooth client 가 있다면
179 {
180 readDataMsg(); // 메세지를 받아옴.
181 }
182 else // 접속한 bluetooth client 가 없다면 메세지 출력.
183 {
184 nxtDisplayCenteredTextLine(3, "BlueTooth");
185 nxtDisplayCenteredTextLine(4, "not Connected");

186 }
187 }
188 }
```

5 아래는 직접 제작한 thinning 알고리즘 소스이다. VS2010 에서 복사해서 가져왔다.

```
1  inline BYTE mygetpixel(int y, int x)
2  {
3    if(x < 0 || y < 0)
4            return 0;
5    if(x >= 640)
6            return 0;
7    if(y >= 480)
8            return 0;
9    return !!table[y][x];
10 }
11
12 int conn(int y, int x)
13 {
14   int n=0;
15   char A[8];
16   A[0] = x+1>=640 ? 0 : table[y][x+1];
17   A[1] = y-1<0 || x+1>=640 ? 0 : table[y-1][x+1];
18   A[2] = y-1<0 ? 0 : table[y-1][x];
19   A[3] = y-1<0 || x-1<0 ? 0 : table[y-1][x-1];
20   A[4] = x-1<0 ? 0 : table[y][x-1];
21   A[5] = y+1>=480 || x-1 < 0 ? 0 : table[y+1][x-1];
22   A[6] = y+1>=480 ? 0 : table[y+1][x];
23   A[7] = y+1>=480 || x+1>640 ? 0 : table[y+1][x+1];
24
25   if( A[0] == 1 && A[1] == 0 ) n++;
26   if( A[1] == 1 && A[2] == 0 ) n++;
27   if( A[2] == 1 && A[3] == 0 ) n++;
28   if( A[3] == 1 && A[4] == 0 ) n++;
29   if( A[4] == 1 && A[5] == 0 ) n++;
30   if( A[5] == 1 && A[6] == 0 ) n++;
31   if( A[6] == 1 && A[7] == 0 ) n++;
32   if( A[7] == 1 && A[0] == 0 ) n++;
33   return n;
34 }
35
36 int around(int y, int x)
37 {
38   int k,l,N=0;
39
40    for(k=y-1;k<=y+1;k++)
41    for(l=x-1;l<=x+1;l++)
42       if(k!=y || l!=x)
43               {
44                       if(k < 0 || l < 0 || k>=480 || l>=640)
45                               continue;
46          if(table[k][l] >=1) N++;
47               }
48   return N;
49 }
50
51 void thinning()
52 {
53   BYTE copytable[480][640];
54   memset(copytable, 0, sizeof(BYTE)*640*480);
55   int WIDTH = 640, HEIGHT = 480, temp = 0;
56   bool again = true;
57   while(again)
58   {
59           again = false;
60           for(int y=0; y<HEIGHT; y++)
```

```
61          {
62                  for(int x=0; x<WIDTH; x++)
63                  {
64                          if(mygetpixel(y,x) != 1)
65                                  continue;
66                          int N = around(y, x);
67                          if((N>=2 && N<=6) && conn(y, x) == 1)
68                          {
69                                  if( ( (mygetpixel(y, x+1) == 0 ||
70                                          mygetpixel(y-1, x) == 0 ||
71                                          mygetpixel(y, x-1) == 0 ) &&
72                                          (mygetpixel(y-1, x) == 0 ||
73                                          mygetpixel(y+1, x) == 0 ||
74                                          mygetpixel(y, x-1) == 0 )
75                                          ) )
76                                  {
77                                          copytable[y][x] = 1;
78                                          again = true;
79                                  }
80                          }
81                  }
82          }
83          for(int y=0; y<HEIGHT; y++)
84          {
85                  for(int x=0; x<WIDTH; x++)
86                  {
87                          if(copytable[y][x] == 1)
88                          {
89                                  table[y][x] = 0;
90                          }
91                  }
92          }
93          if(again == false) break;
94          ////////////////////----------------------------------------------------------
95          memset(copytable, 0, sizeof(BYTE)*640*480);
96          for(int y=0; y<HEIGHT; y++)
97          {
98                  for(int x=0; x<WIDTH; x++)
99                  {
100                         if(mygetpixel(y,x) != 1)
101                                 continue;
102                         int N = around(y, x);
103                         if((N>=2 && N<=6) && conn(y, x) == 1)
104                         {
105                                 if( (mygetpixel(y-1, x) == 0 ||
106                                         mygetpixel(y, x+1) == 0 ||
107                                         mygetpixel(y+1, x) == 0 ) &&
108                                         (mygetpixel(y, x+1) == 0 ||
109                                         mygetpixel(y+1, x) == 0 ||
110                                         mygetpixel(y, x-1) == 0 ) )
111                                 {
112                                         copytable[y][x] = 1;
113                                         again = true;
114                                 }
115                         }
116                 }
117         }
118         for(int y=0; y<HEIGHT; y++)
119         {
120                 for(int x=0; x<WIDTH; x++)
121                 {
```

```
1 2 2                                   if(copytable[y][x] == 100)
1 2 3                                   {
1 2 4                                           table[y][x] = 0;
1 2 5                                   }
1 2 6                           }
1 2 7                   }
1 2 8                   memset(copytable, 0, sizeof(BYTE)*640*480);
1 2 9           }
1 3 0 }
```