C 프로그래밍

로봇 과제 1

-핵 연료봉 옮기기 -

담당교수	:	차의영 교수님	
분반	:	001	
학과	:	기계공학부	기계공학부
학번	:	200721338	200721395
이름	:	정상효	한경수
제출일자	:	2011.03.26	

1. Mission
   - 핵시설 안에 있는 핵 연료봉을 찾아서 로봇이 핵 연료봉을 바닷물 속에 넣는다.
   - 사람이 들어가면 방사능에 노출되어 사망하므로 로봇을 원격조종한다.

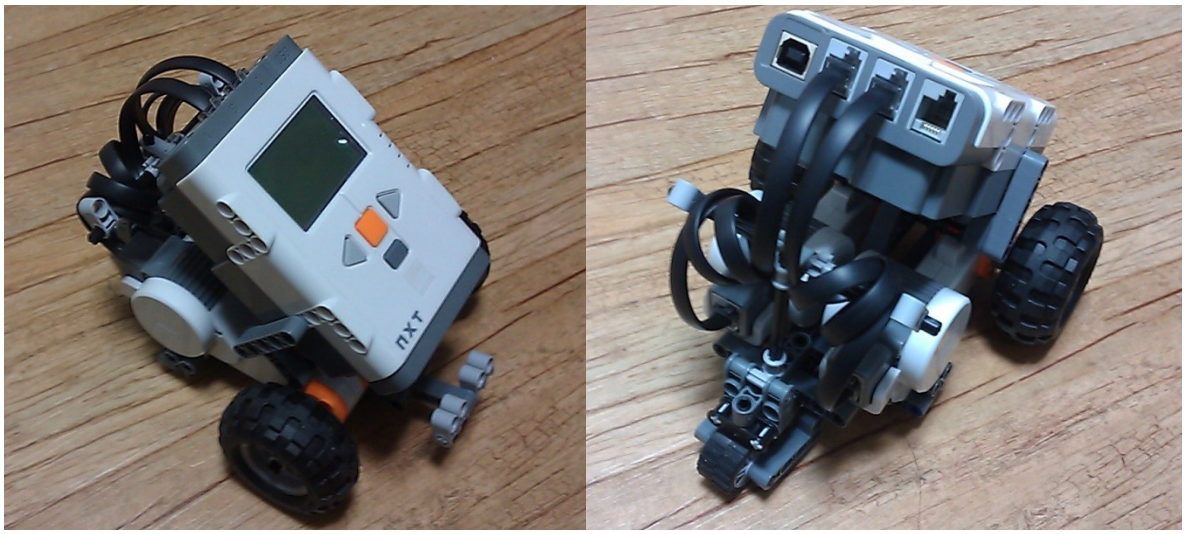
2. 해결 방안
   - LEGO NXT Mindstorm 과 스마트폰을 이용하여 로봇을 조종하며, 주어진 RobotC 를 이용, 프로그래밍하여 로봇을 조종한다.
   - 빨간 공을 주워서 원하는 위치에 내려 놓을 수 있도록 프로그래밍해야 한다.

3. 문제 해결 과정
   A. 우선 전반적인 로봇의 움직임을 구상해야 했다.
   B. 두 바퀴는 모터 B 와 모터 C 를 사용했다. 처음에 빨간 공을 잡기 위한 수단으로, 집게를 이용하여 양 측면에서 잡는 방법을 구상했고, 모터 A 를 통해서 구동시키려고 했으나, 레고 블록의 특성상 정형화 되어있으므로 원하는 방향으로 기어, 블록을 구성할 수 없었다.
   C. 결국은 포크레인과 같은 방법을 모색하게 되었으며, 빨간 공을 효율적으로 쉽게 잡기 위해 고민하였고, 지금에 이르렀다.
   D. 하드웨어(로봇)적인 문제는 큰 걱정이 없었으나, 블루투스 통신에 대해 경험이 없는 우리조로서는 매우 막막했다. 안드로이드 운영체제를 사용했는데, NXT 와 통신이 가능한 공개된 소프트웨어는 단지 로봇을 전후좌우로 움직이게 할 뿐이었다. 또한, 기존에 있었던 프로그램은 특별히 내부적인 프로토콜에 의해 신호를 보내면 그에 따라 모터가 움직이는 방식이었다. 우리에게 중요한 것은 빨간 공을 잡기 위한 모터 A 를 구동시키는 것이므로 우리에게 맞는 프로그램이 사실상 전무하다고 보았다. 그에 따라 여러가지 정보를 찾아 본 결과, 휴대용 단말기에서 로봇으로 보내는 블루투스 신호를 잡아내면 그것을 이용, catch 하여 내가 원하는 명령으로 바꾸는 방법을 생각했다.
   E. 블루투스를 이용하여 NXT 에 특정 데이터를 보내는 프로그램을 찾아내었으나, 그 특정 데이터는 모터 구동에 관한 데이터가 아닌 단순한 데이터(i.e : 0x20, 0x21) 였으므로 모터 구동을 시킬 수가 없었다. 하지만 결과적으로 블루투스를 이용하여 데이터를 전송할 수 있다는 것을 깨달았으므로 그 데이터를 읽어들여 모터를 구동시키기 위해 도움말 기능을 참고하였고, 내부적인 샘플 코드를 찾기에 이르렀다. 그 중에 우리가 원하던 BtBasicMessage 프로젝트를 찾게 되었다.
   F. 휴대용 단말기에서 패킷을 보내고 NXT 에서 그 패킷을 인식하여 그 패킷에 따라 행동을 시키게 하면 된다는 틀이 잡혔고, 이내 성공하게 되었다.

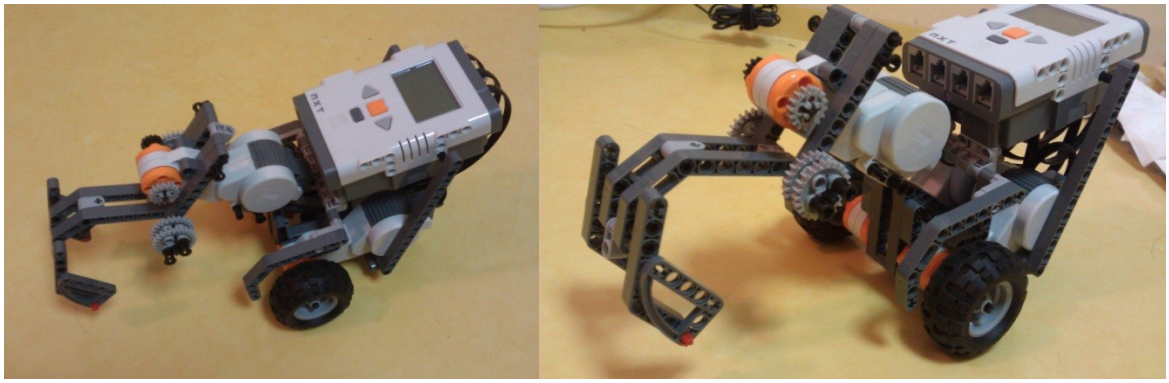
4. 로봇 설계 과정
   A. 최초에는 매뉴얼을 보고 따라서 만들었으나, 매뉴얼에 있던 방식은 지게차 원리로 동작하는 방식이었고 효율적이지 못했다.
   B. 그래서 생각해낸 방식이 집게 방식이었다. 쉽게 되리라 생각했고, 조립이 덜 된 상태에서 테스트를 진행했을때는 매우 만족스러웠다. 그러나 모터 A 와 집게를 연결하는 과정에서 모터의 힘 방향 전환이 무척이나 까다로웠고, 잘 들어올려지지 않았다. 그래서 다른 방법을 모색하게 되었다.
   C. 뜯고 조립하는 과정을 수 회 반복하면서 생각해낸 결과, 포크레인 방식이 가장 이상적이라는 생각을 하고 새로이 만들었다. 이 방식을 이용하면 모터의 구동 방향과 같으므로 따로 모터 힘의 방향 전환을 하지 않아도 되었고, 공을 들어올리거나 떨어뜨리는 동작이 매우 안정적인 것을 확인 할 수 있었다.

5. 로봇 관련 사진

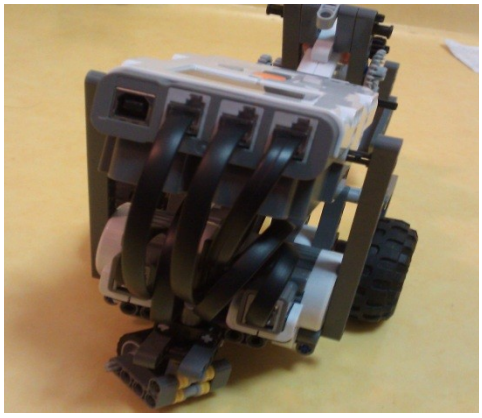
A. 정보컴퓨터공학부 사무실에서 대여받은 LEGO Mindstorm 이다.



B. 기본적인 전후좌우 이동 기능만 수행할 수 있도록
최소한으로 꾸민 프로토 타입의 로봇이다.



C. 완성된 현재의 로봇 모습이다. 처음의 모습과는 사뭇 다르다

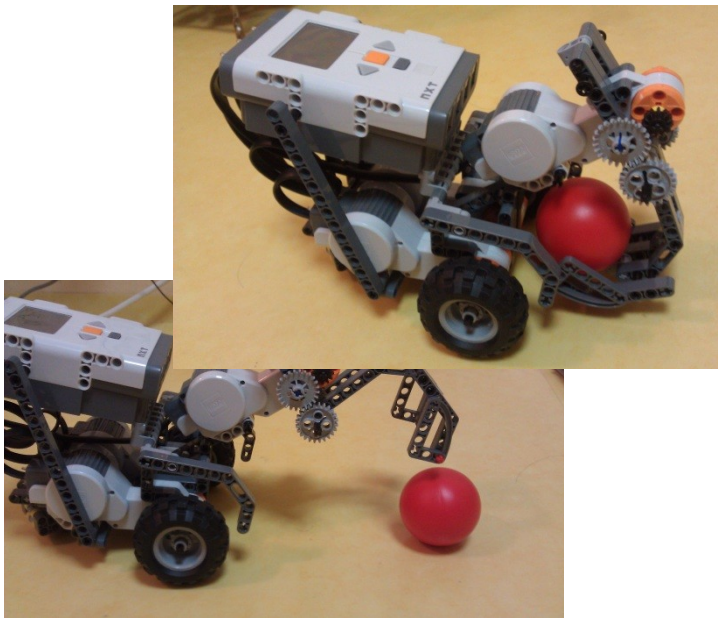


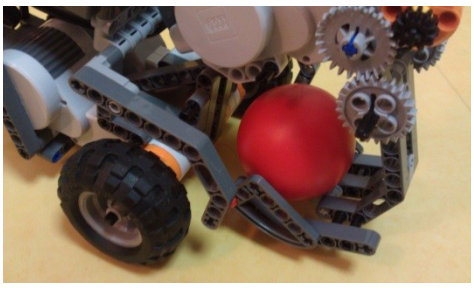
D. 후면의 모습이다.
좌측부터 C, B, A 모터가 연결된 것이며,
케이블이 긴 관계로 두어번 살짝 감아주어
모터와 NXT 사이를 연결했다.

E. 우측의 사진은 완성 후, 빨간 공을
로봇이 담는 과정을 간단하게
사진으로 나타내었다.

포크레인 부가 빨간공보다 높이
설치되어 있으므로 이 공을
건드리지 않고 안전하게 실을 수 있으며,
양 측면의 가이드를 이용하여
운반 중에도 떨어뜨리지
않게 하였다.

내려놓을 때는 모터의 구동만
반대방향으로 하면 자연스레
내려놓을 수 있다.





6. 프로그램 소스 코드는 다음과 같고, 직접 작성하였다.

```
pragma platform(NXT)
void action(ubyteinp, int length) // 들어오는입력값에대해동작
{
ubyte act = inp;
int static speed = 10; // 기본속도를 static
if(act == 1) // forward
 {
motor[motorB] = speed;
motor[motorC] = speed;
motor[motorA] = 0;
  speed += 10; // 누를때마다속도증가
 }
else if(act == 5) // rotation
 {
motor[motorB] = -10;
motor[motorC] = 10;
motor[motorA] = 0;
 }
else if(act == 3) // rotation
 {
motor[motorB] = 10;
motor[motorC] = -10;
motor[motorA] = 0;
 }
else if(act == 7) // backward
 {
motor[motorB] = -speed;
motor[motorC] = -speed;
motor[motorA] = 0;
  speed += 10; // 누를때마다속도증가
 }
else if(act == 4) // all stop
 {
motor[motorB] = 0;
motor[motorC] = 0;
motor[motorA] = 0;
  speed = 10; //전부스톱시스피드 10
//    PlaySound(soundBeepBeep);
 }
else if(act == 8) // pick up
 {
nMotorEncoder[motorA] = 0;
  while(nMotorEncoder[motorA] <= 90) // 90 도만큼회전
   motor[motorA] = 20; // 20% 파워로
  motor[motorA] = 0; // 다회전시켰으면 0%
 }
```

```
else if(act == 2) // pick up
 {
nMotorEncoder[motorA] = 0;
  while(nMotorEncoder[motorA] >= -90) // -90 도만큼회전
   motor[motorA] = -20; // 20% 파워로뒤로돌림
  motor[motorA] = 0; // 다돌렸으면 0% 로.
 }
}
voidreadDataMsg()
{
constTMailboxIDskQueueID = mailbox1; // mailbox1 사용
constintkMaxSizeOfMessage = 5;
TFileIOResultnBTCmdRdErrorStatus;
intnSizeOfMessage;
   ubytenRcvBuffer[kMaxSizeOfMessage * 5]; // Max 배열크기는있지만
                                                    그냥적당히 *5
memset(nRcvBuffer, 0, sizeof(nRcvBuffer)); // nRvcBuffer 모두 0 으로초기화
while (1)
 {
nSizeOfMessage = cCmdMessageGetSize(kQueueID); // 받아와야할
                                                        메세지의크기를구함
  if (nSizeOfMessage == 0) // 없다면함수에서나가게됨
  {
wait1Msec(1);
break;
  }
  if (nSizeOfMessage>kMaxSizeOfMessage) // 크기가너무클경우엔
                                                    지정해준크기만큼만받음.
nSizeOfMessage = kMaxSizeOfMessage;
nBTCmdRdErrorStatus = cCmdMessageRead(nRcvBuffer, nSizeOfMessage,
kQueueID);
  if (nBTCmdRdErrorStatus == ioRsltSuccess) // 에러가나지않았다면
  {
eraseDisplay(); // 화면을지우고
   //아래주석부분은디버깅용코드
   /*ubyte exist = false;
   int i;
   for(i=0; i<kMaxSizeOfMessage * 5; ++i)
   {
    if(nRcvBuffer[i] != 0)
    {
     exist = 1;
     break;
    }
   }
   if(exist)
    nxtDisplayCenteredTextLine(3, "nice!!");*/
```

```
    action(nRcvBuffer[0], kMaxSizeOfMessage * 5); // 받은패킷을
                                                                    action 함수로넘김.
    // 모터의속도크기를 LCD 에전시
nxtDisplayCenteredTextLine(0, "A's speed : %d", motor[motorA]);
nxtDisplayCenteredTextLine(1, "B's speed : %d", motor[motorB]);
nxtDisplayCenteredTextLine(2, "C's speed : %d", motor[motorC]);
  }
else
nxtDisplayTextLine(2, "Failed");
 }
}
task main()
{
nMotorPIDSpeedCtrl[motorB] = mtrSpeedReg; // PID 적용
nMotorPIDSpeedCtrl[motorC] = mtrSpeedReg; // PID 적용
while(1)
 {
  if (nBTCurrentStreamIndex>= 0) // 접속한 bluetooth client 가있다면
  {
readDataMsg(); // 메세지를받아옴.
  }
  else // 접속한 bluetooth client 가없다면메세지출력.
  {
nxtDisplayCenteredTextLine(3, "BlueTooth");
nxtDisplayCenteredTextLine(4, "not Connected");
  }
 }
}
```