기말고사 레포트

201636417 심우석

1. Sound

레고 Mind storm 에는 EV3 Brick 이라는 몸통역할의 부품이 있다. 그것을 이용하여 Sound Function 들을 사용할 수 있다. Sound Functions 는 크게 두가지로 나눌 수 있다. 내부에 내장된 내부 사운드를 이용하거나, 외부 사운드 혹은 내가 직접 설정한 값들의 소리를 나게 할 수 있다.

첫번째로 내부 사운드를 이용하는 방법이다. PlaySound(sound)를 이용한다. 여기서 Input parameter 는 미리 설정된 이름들 중 하나를 사용할 수 있다. 미리 설정된 값들은 soundBeepBeep, soundLowBuzz, soundBlip, soundLowBuzzShort, soundDownwardTones, soundShortBlip, soundException, soundUpwardTones, soundFastUpwordTones 등이 있고, RobotC firmware 는 총 10 개의 sound items 를 queue 에 저장할 수 있다.

두번째로 외부 사운드 혹은 내가 직접 설정한 값들의 소리를 나게 하는 방법 중 첫째로 PlayTone(frequency, durationIn10Msec), PlayImmediateTone(frequency, durationIn10Msec)를 이용하는 방법이 있다. Frequency 는 옥타브와 음계 표를 참고해서 설정해 놓으면 그 음에 맞춰 소리가 난다. durationIn10Msec 는 설정해 놓은 값에 따라서 음이 지속되는 시간이다. 일반적으로 PlayTone 을 사용하고 바로 소리가 나게 하고 싶으면 PlayImmediateTone 을 사용한다.

둘째로 외부 사운드의 소리를 내고 싶으면 .rsf file 을 EV3 Brick 에 저장 해놓고 playSoundFile("파일이름")을 입력하면 사운드 파일에 저장된 소리를 출력하게 된다.

소리를 내는 것 이외에도 다른 기능들이 있는데, ClearSounds()는 존재하고 버퍼 된 소리들을 지워준다. MuteSound(), UnmuteSound()는 소리를 안 나게 하거나 나게 할 수 있다. 소리를 내다가 멈추고 싶으면 sleep(Msec)를 입력한다면 Msec 간 소리가 나다가 소리를 멈춘다.

bSoundActive 는 소리가 연주 중인지 아닌지 확인할 수 있고, nVolume(0 to 4)를 이용해 소리의 크기를 조절할 수 있다.

2. Display Function

EV3 Brick 의 LCD 화면을 이용해 Display Function 들을 사용할 수 있다. Display 는 보통 다른 기능들을 이용할 때 EV3 Brick 에 표시를 해주는 식의 보조적인 역할을 수행한다. Touch Sensor, Color Sensor, Ultrasonic Sensor 등과 같이 센서들의 측정값을 실시간으로 표현해주며 사용자를 보조해줄 수 있다.

EV3 Brick 에 내장된 LCD 의 Size 는 178 pixels by 128 pixels 로 (0 ~ 177, 0 ~ 127) 값을 입력해서 지정된 위치에서 Text 를 display 해줄 수 있다. 일반 글꼴로 각각의 글씨는 8 pixels 를 이용하며 위치를 지정해주려면 세로줄을 (0 ~ 15)의 값을 지정해주면 된다.

displayTextLine(nLineNumber, sFormatString, ...);
displayBigTextLine(nLineNumber, sFormatString, ...);
위 코드가 일반형이고, 만약 8 pixel 사이즈가 아닌 16 pixel 사이즈로 출력하기를 원한다면 displayBigTextLine 을 이용하면 된다.

또한 eraseDisplay() 코드를 사용하면 현재 LCD 화면에 나오고 있는 글들을 없앨 수 있다. displayTextLine 을 이용하면 string type 뿐만 아니라 character, integer 값도 사용할 수 있다.
displayTextLine(0, "%s %s", "Hello", "RobotC!"); string type
displayTextLine(0, "%d", value); integer type

또한 while 문, for 문을 이용해 연속적으로 진행되는 loop 속에서 eraseDisplay()를 사용하지 않는다고 하더라도 값들은 지속적으로 변경된다.
Ultrasonic sensor 로 값을 계속 받아오고 while 문을 사용한다고 했을 때

```
Int value;
While(1) {
        Value = SensorValue( Ultra );
        displayTextLine( 0, "%d cm", value); }
```

이와 같은 코드를 사용한다면 LCD 에 표시되는 값은 지속적으로 업데이트된다.

3. Touch Sensor

Touch Sensor 는 레고의 모듈 앞에 물리적인 버튼의 상태가 눌린 것 혹은 놓인 것의 변화를 감지합니다. Touch Sensor 가 눌린 상태라면 1 로 유지되고 놓인 상태라면 0, 눌렸다가 놓이면 1 -> 0 으로, 놓였다가 눌리면 0 -> 1 로 변화합니다.

Touch Sensor 를 사용하기 위해서는 첫째로 EV3 Brick 의 LCD 모니터를 통해서 Touch Sensor 의 값이 눌린 것과 놓인 것의 변화를 확인해야합니다.
그 다음으로 EV3 Brick 을 컴퓨터와 연결해서 Sensor configuration 을 해줘야 합니다.
그 후 센서 값을 실시간으로 측정합니다.

Touch Sensor 의 대표적인 사용의 예로는 로봇이 앞으로 가는 도중에 벽 혹은 장애물이 있다면 Touch Sensor 의 값이 0 에서 1 로 변화될 것이고 그에 따른 후속 행동들을 하게 도와줄 수 있습니다.

Touch Sensor 의 실제 사용 가능한 코드입니다.

```
// 직접 코딩해도 되지만 sensor configuration 과정을 통해서 자동 입력할 수 있습니다.
#pragma config(Sensor, S1, touchSensor, sensorEV3_touch)
//*!!Code automatically generated by 'ROBOTC' configuration wizard!! *//
Task main()
{       //앞으로 직진
        Motor[motorB] = 50;
        Motor[motorC] = 50;

        //진행하다가 Sensor 가 눌린다면 로봇을 0.5 초간 뒷걸음치게 한다.
        While(getTouchValue(touchSensor) == 0) {}

        Motor[motorB] = -50; Motor[motorC] = -50;
        Wait1Msec(500);
}
```

이렇듯 Touch Sensor 는 다른 행동들의 보조적인 역할로 사용된다.

4. Color Sensor

Color Sensor 는 Red LED for indirect light 와 photo transistor 를 이용해서 0.5 ~ 1.0 cm 의 거리의 색, 명암 등을 고려해서 값을 읽는다.

Color Sensor 는 Touch Sensor 와 마찬가지로 연결을 확인하고 LCD 모니터를 통해서 값의 변화를 확인한다. 그 다음으로 EV3 Brick 을 컴퓨터와 연결해서 Sensor configuration 을 해줘야 합니다. 그 후 센서 값을 실시간으로 측정합니다.

Color Sensor 는 크게 3 가지로 구분됩니다. 첫번째로 Active mode(reflected light), 두번째로, Inactive mode(ambient light), 세번째로 Color mode(seven different colors) 입니다. Reflected light 는 흰색과 검은색을 0 ~ 100 값으로 반환해서 측정한다. Ambient light 는 밝은 것과 어두운 것을 0 ~ 100 값으로 반환해서 측정한다. Seven different colors 의 경우 빨, 주, 노, 초, 파, 남, 보의 색으로 구분해서 각각의 색을 0 ~ 7 값으로 반환해서 측정한다.
Reflected light 과 Ambient light 은 검은색, 어두울 때의 값이 0 이고 색이 점점 밝아 짐에 따라서 100 으로 가까워진다. 일반적으로 Threshold 를 설정 해놓고 사용한다. Value = (Black + white) / 2 값으로 놓는다.

Color Sensor 의 대표적인 사용의 예로는 로봇이 진행하는 도중에 Ambient light, Reflected light 가 인식한 값들이 어둡다고 인식하면 로봇의 진행을 멈출 것입니다.
Seven different light 의 경우 만약 신호등이 있다고 했을 때 로봇이 진행하는 도중에 인식한 것이 빨간색이라면 로봇을 멈추게 하고 초록색이라면 로봇을 계속 진행하게 설계할 수 있습니다.

```
// 만약 다른 모드로 바꾸고 싶다면 표시된 값들을 변경하면 됩니다.
#pragma config(Sensor, S3 reflectedValue, sensorEV3_Color, modeEV3Color_Reflected)
// 로봇이 앞으로 진행합니다.
Motor[motorB] = 50, motor[motorC] = 50;
// Threshold 가 50 일 때 그 값보다 낮아(어두워)지면 로봇의 진행을 멈춥니다.
While(SensorValue(reflectedValue) > THRESHOLD) {}
Motor[motorB] = 0, motor[motorC] = 0;
```

5. Ultrasonic Sensor

Ultrasonic Sensor 는 잠수함에 사용되는 메커니즘과 같습니다. 초음파 시간을 보내고 반사된 것의 신호를 측정하고, 만약 변화가 감지된다면 알려주는 형태입니다. MindStorm 은 cm 단위로 측정을 합니다. 거리 측정은 약 3 ~ 100 cm 가량 측정이 가능합니다.

Ultrasonic Sensor 를 사용하기 위해서는 첫번째로 EV3 brick 과의 연결을 확인하고 손과 같은 물질을 이용해서 변화를 주었을 때 LCD 에 표시되는 값의 변화가 있는지 확인합니다. 그 다음으로 EV3 brick 을 컴퓨터와 연결해서 Sensor configuration 을 해줘야 합니다. 그 후 센서 값을 실시간으로 측정하며 사용합니다.

Ultrasonic Sensor 의 대표적인 사용의 예로 앞차와의 거리를 계속 측정해서 자율주행하는 크루즈 모드 기능이 있고, 더 나아가서 장애물과의 거리를 지속적으로 측정하며 부딪히지 않고 계속 진행할 수 있게 완전한 자율주행 설정을 할 수 있다.

Ultrasonic Sensor 의 실제 사용 코드입니다.
Robot 과 물체 사이의 거리 30cm 를 지속적으로 유지합니다.

```
// 직접 입력할 수도 있지만 설정을 통해 자동입력 가능합니다. S4 포트를 이용합니다
#pragma config(Sensor, S4, ultra, sensorEV3_Ultrasonic)
Task main() {
        Int value;
        While (1) {       // SensorValue(ultra) 값을 지속적으로 측정하여 로봇 앞에
                          // 물체가 있는지 확인하고 물체의 거리를 표시해줍니다.
                Value = SensorValue(ultra);
                displayTextLine(0, "%d cm", value);
                // 물체와 로봇의 거리 30cm 를 앞뒤로 움직이며 유지합니다.
                if (value > 30) { motor[motorB] = 30; motor[motorC] = 30; }
                else if (value < 30) { motor[motorB] = -30; motor[motorC] = -30; }
                else { motor[motorB] = 0; motor[motorC] = 0; }
}       }
```

6. Servo motor

Mindstorm 에서 Servo motor 는 크게 두가지를 조절할 수 있다. 하나는 모터의 파워를 조절할 수 있는데 -100 부터 100 까지의 값들로 설정할 수 있습니다. 또 회전 값을 정확하게 측정이 가능한데, 이 값을 통해서 정확한 회전제어가 가능합니다.

Servo motor 를 사용하기 위해 우선 EV3 brick 과의 연결을 확인하고 LCD 에 포트가 잘 연결이 되어있는지를 확인합니다. 그 다음 첫번째로 Servo motor 의 상태를 읽고 값을 초기화 해주어야 합니다. 두번째로 모터의 각도와 파워를 설정해 입력해줍니다.

예시를 보면

nMotorEncoder[motorB] = 0; 혹은 resetMotorEncoder(motorB);를 입력, 초기화합니다. setMotorTarget(motorB, 268, 30); 을 입력해 사용할 모터, 모터의 각도, 파워를 설정합니다.

또한 EV3 의 Servo motor 는 회전된 각도가 목표 값에 도달하면 모터 회전을 약간 느리게 해주기 때문에 Servo motor 를 배우기 전에 이용했던 방법보다 훨씬 부드럽다는 장점을 가지고 있습니다.

정확한 Turn 을 위해서는 Swing turn 과 Point turn 두가지 방법을 사용합니다. Swing turn 은 한 개의 바퀴만을 움직여서 특정 각도만큼 회전시킵니다. 바퀴 사이의 거리 a 와 바퀴의 반지름 r, 바퀴가 로봇을 t 도 회전시키는 회전 각도 α 를 이용합니다. 왼쪽으로 90 도 꺾는다고 가정했을 때의 코드는

첫번째로 각도와 반지름 거리를 설정해주고

```
Int degree;        float angle = 90.0, radius = 2.8, axis = 12;

Degree = (int)(axis / radius * angle);
```

모터를 리셋하고 각도와 힘을 설정 후 그 Turn 동작이 끝나면 기계를 멈춥니다.

```
resetMotorEncoder(motorB);      setMotorTarget(motor, degree, 30);

while (getMotorRunning(motorB)){}
```

Point turn 은 두개의 바퀴를 같이 움직여서 특정 각도만큼 회전시키는데 만약 오른쪽 회전이라면 오른쪽 바퀴는 앞으로, 왼쪽 바퀴는 뒤로 움직입니다. 또한 위의 코드에서 거리를 / 2 해주면 위의 코드와 같이 사용할 수 있습니다.

로봇공학에 적용되는 AI 기술 사례 조사

As soon as I saw a case study of Artificial Intelligence – AI technology applied to robotics, Boston Dynamics came to my mind. Boston Dynamics is an engineering and robotics design company based in the US that was derived from MIT in 1992. Boston Dynamics is known to use deep learning such as Big Dog, Spot and Atlas to develop robots used in industrial, military and business applications. As a side note, it was acquired by Hyundai motor Company on December 10, 2020 for a scale of $921m.

At Boston Dynamics, the dog-shaped robot called Spot is the most popular. Spot is an autonomous robot that walks on four legs, and processes data using onboard AI software. Use deep learning to process data, act on it, and protect if from hazardous environments. "Spot can collect a variety of data, such as visual images, 3D laser scans or thermal images, and operate in specialized environments." Said Michael Perry, vice president of business development at Boston Dynamics.

Spot has a total of 5 cameras, 2 at the front, 1 at the back, and 1 at each side. A light projector attached to each of the cameras illuminates toward the ground, helping the robot to stand securely while providing obstacle avoidance. If the spot is over 30cm in height, it is recognized as an obstacle and avoided sideways.
Also, the Spot uses cameras for location realization and mapping (SLAM) for autonomous navigation. Spot provides a route within the map by navigating the space to secure the mapping and creating a 3D point cloud of space. This determines where to go and how to behave.

Even the development of a small dog-shaped robot will require a lot of skills. It is really fascinating to use cameras to analyze information in real time and to synthesize data to design how the robot will move. Advanced AI technology is required for robots with only shells to set their own path and avoid objects.
Deep learning can supplement this and will be the answer.