



CNRLAB

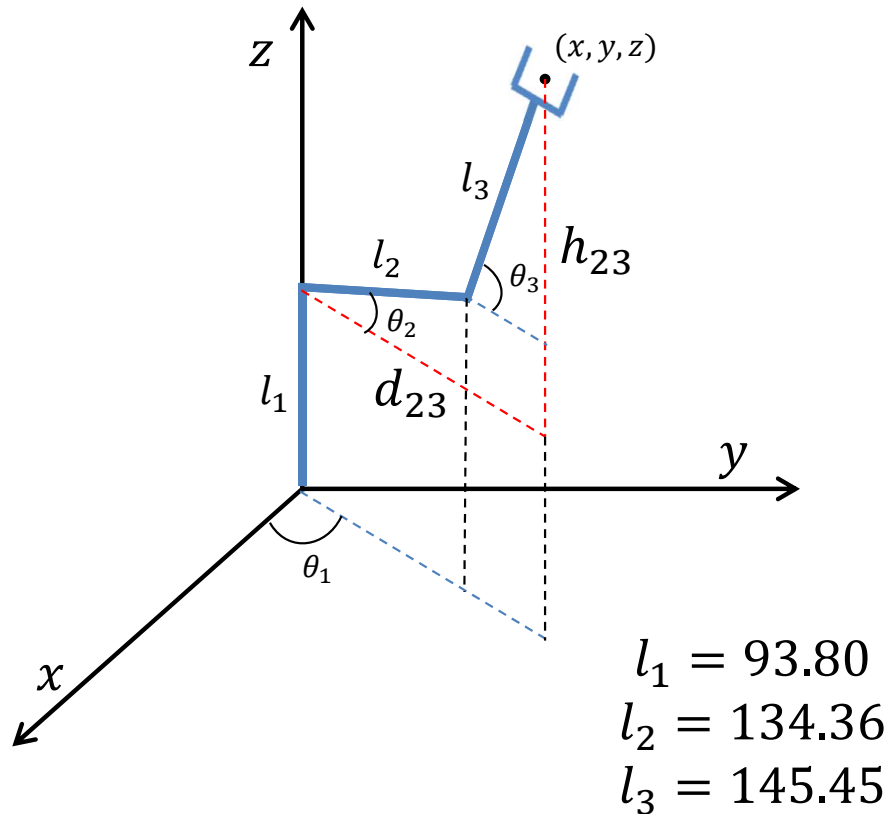
CIM&ROBOTICS LABORATORY

로봇공학입문설계

13주차 로봇 팔(2)

로봇공학과

□ 3 link arm



$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = ?$$

$$d_{23} = l_2 \cos \theta_2 + l_3 \cos \theta_3$$

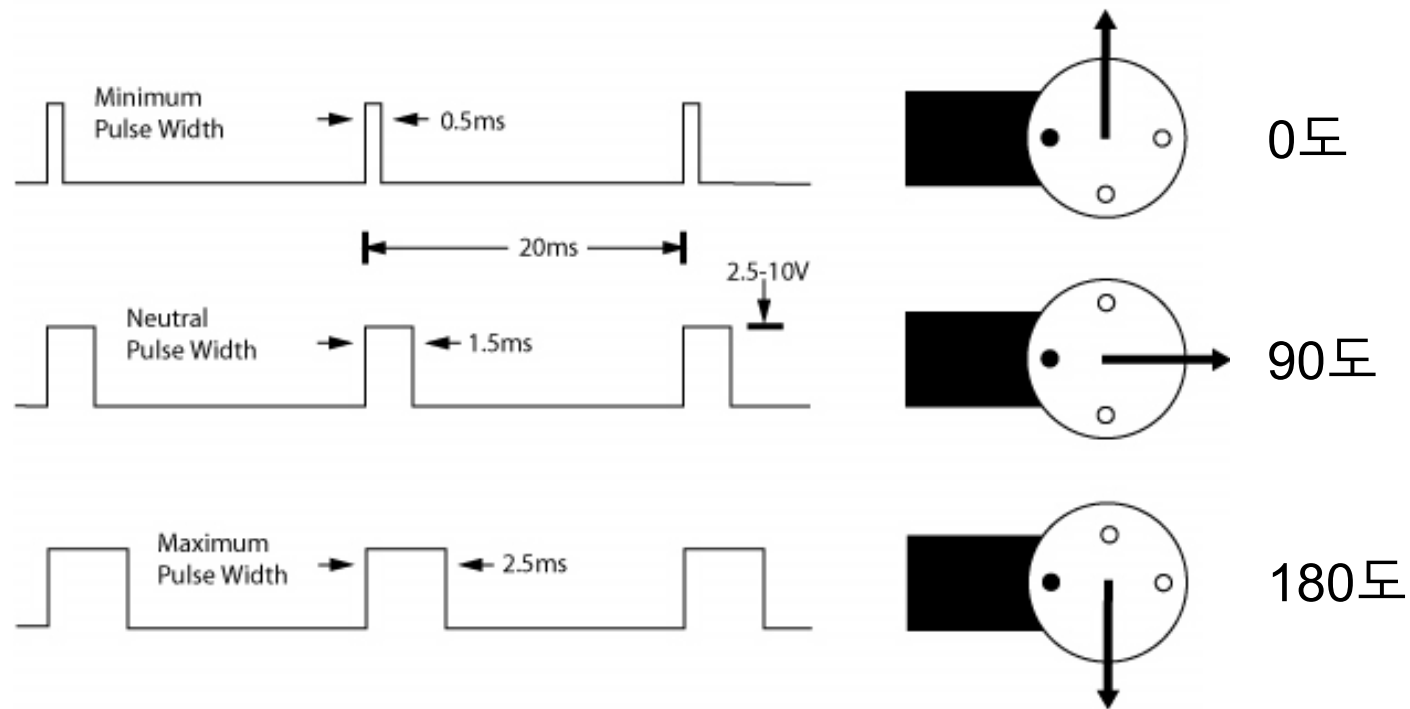
$$h_{23} = l_2 \sin \theta_2 + l_3 \sin \theta_3$$

$$\therefore \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} d_{23} \cos \theta_1 \\ d_{23} \sin \theta_1 \\ l_1 + h_{23} \end{bmatrix} = \begin{bmatrix} l_2 c_1 c_2 + l_3 c_1 c_3 \\ l_2 s_1 c_2 + l_3 s_1 c_3 \\ l_1 + l_2 s_2 + l_3 s_3 \end{bmatrix}$$

$$(c_2 = \cos \theta_2, s_2 = \sin \theta_2)$$


□ Servo Motor

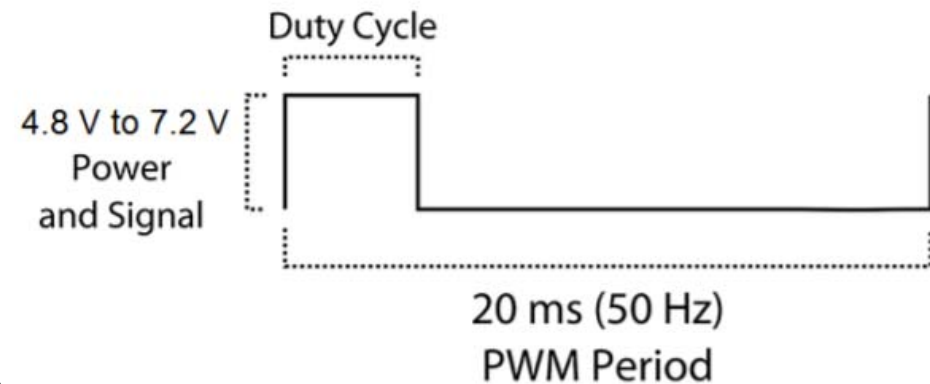
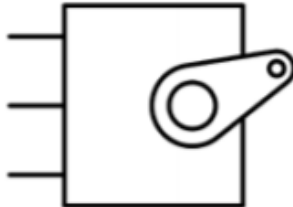
- 모터의 **회전각도**를 **PWM**에 의해 제어
- 일반적으로 0~180도 회전
- 아두이노 사용시, Servo 라이브러리 이용



□ MG996R Specification



PWM=Orange ()
Vcc = Red (+)
Ground=Brown (-)



- Stall torque: 9.4 kgf·cm(4.8V), 11 kgf·cm(6V)
- Operating speed: 0.17 s/60°(4.8V), 0.14 s/60°(6V)
- Operating voltage: 4.8 V a 7.2 V
- Running Current : **500 mA~**
- Stall Current : 2.5 A (6V)
- Dead band width: 5 μ s

□ 서보모터 연결

- Step-Down DC&DC Converter : 12V → 5V



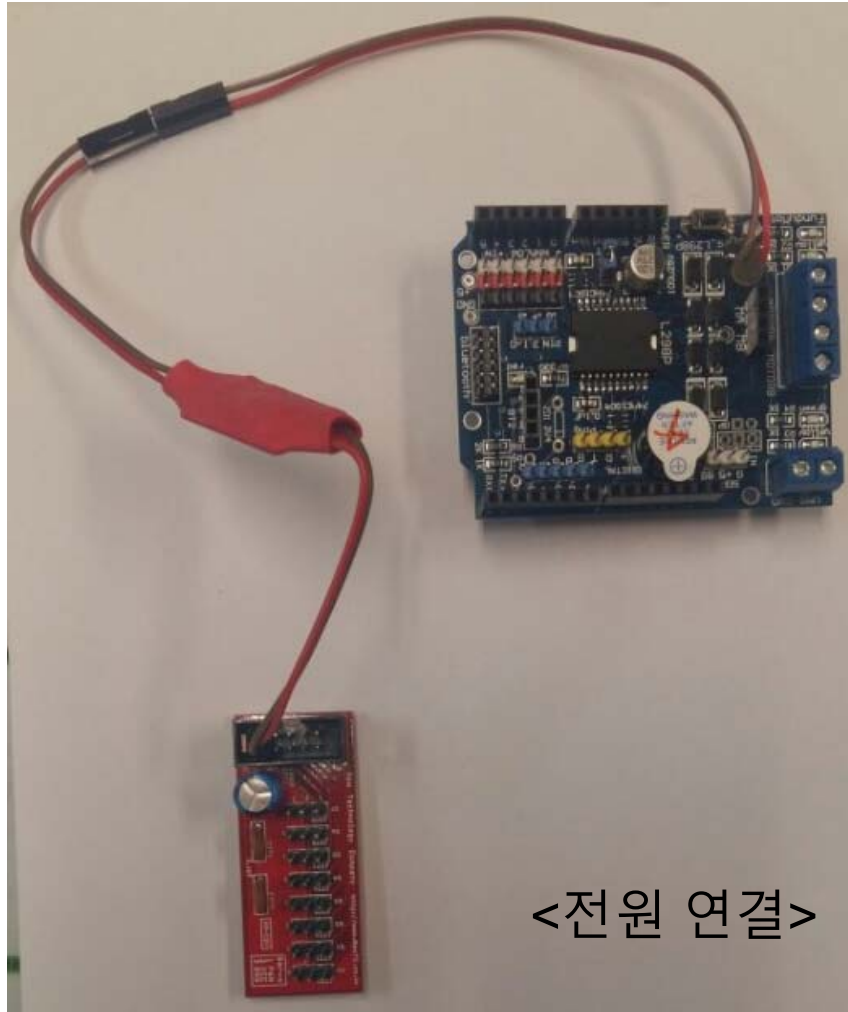
- 서보모터 제어 모듈(외부 전원 사용)



1	PWM
2	VCC
3	GND

1	3	5	7	9	VCC
					GND
2	4	6	8	10	

□ 서보모터 연결

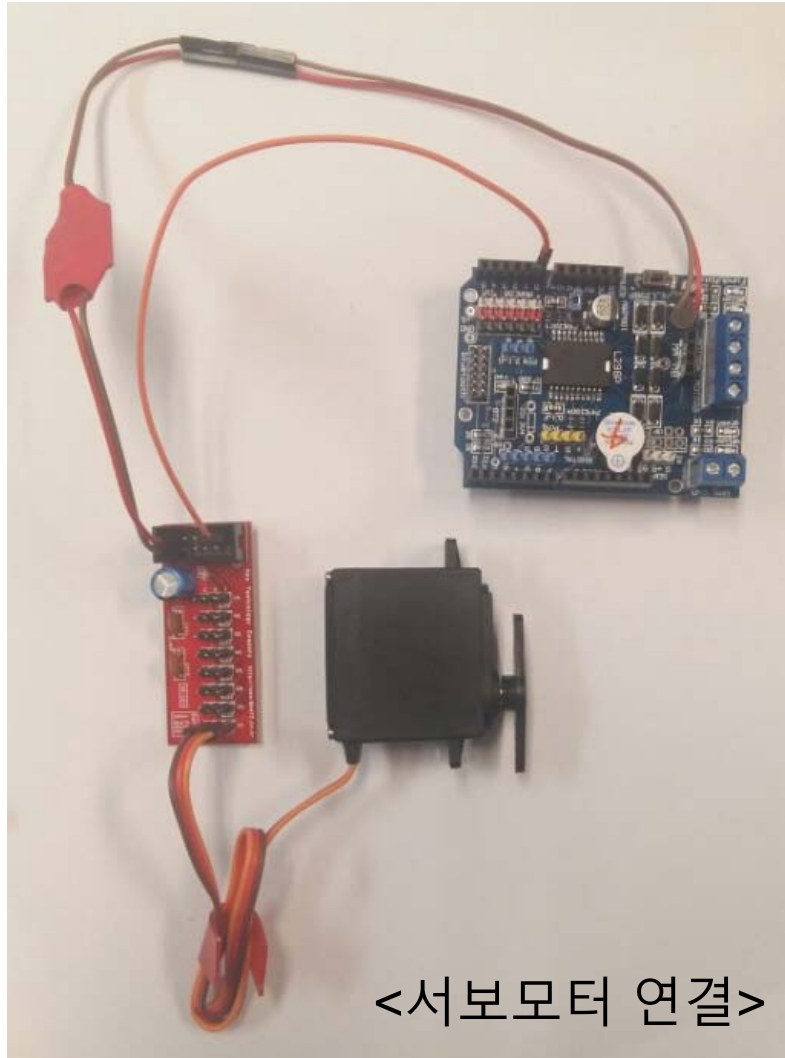


<전원 연결>

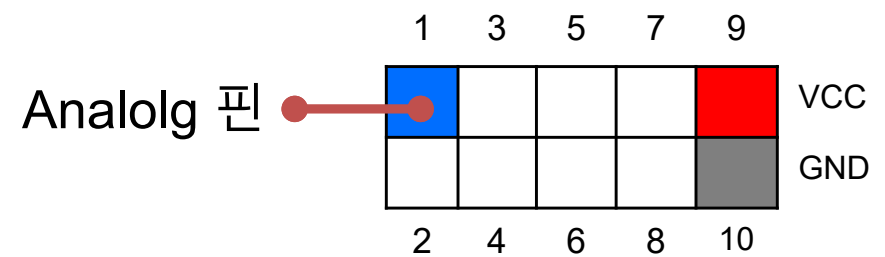
1	3	5	7	9	
					VCC
					GND
2	4	6	8	10	

서보제어모듈

□ 서보모터 연결



<서보모터 연결>



Arduino

서보제어모듈

[예제1] 가변저항을 이용하여 서보모터 구동

```
#include <Servo.h>
#define potPin1 A3

Servo servo1;

void setup() {
  servo1.attach(A0);
  Serial.begin(9600);
}

void loop(){
  int val1 = analogRead(potPin1);
  int ang1 = map(val1,0,1023,0,180);

  Serial.print(ang1);
  Serial.println();

  servo1.write(ang1); //서보의 각도를 설정
  delay(15);          // 서보가 지정한 각도까지 움직이는 동안 대기
}
```


□ Step1. 영점조절

```
#include <Servo.h>

Servo servo1;
Servo servo2;
Servo servo3;

void setup() {
  servo1.attach(A0);
  servo2.attach(A1);
  servo3.attach(A2);
  Serial.begin(9600);
}

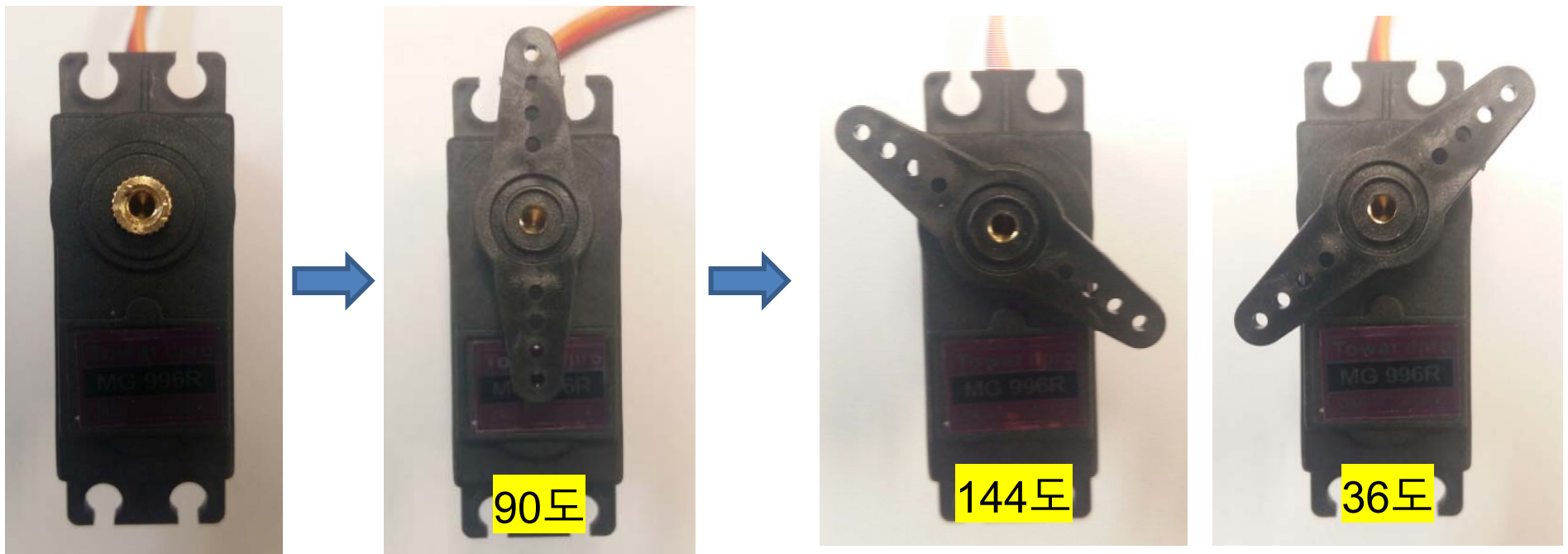
void loop(){
  servo1.write(36);
  servo2.write(90);
  servo3.write(144);
}
```

Tip

- 90도를 기준으로 맞춘다
→ 서보모터의 안정적 제어는 일반적으로 0.9~2.1ms(36~144도)범위에서 가능
- 가변저항을 이용하지 않는다
→ 가변저항을 이용할 경우 정확한 값을 주기 어려움

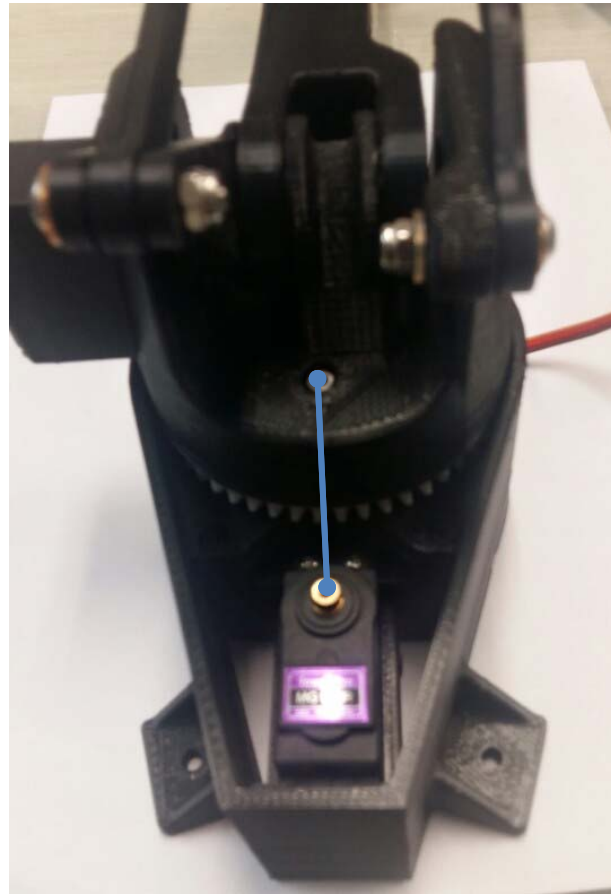
□ Step1. 영점조절

1. 서보모터를 90도로 맞춘 후 혼을 끼운다.
2. 서보모터에 36도와 144도 값을 각각 주었을 때 세개의 모터가 모두 같은 범위를 움직이는지 확인한다.



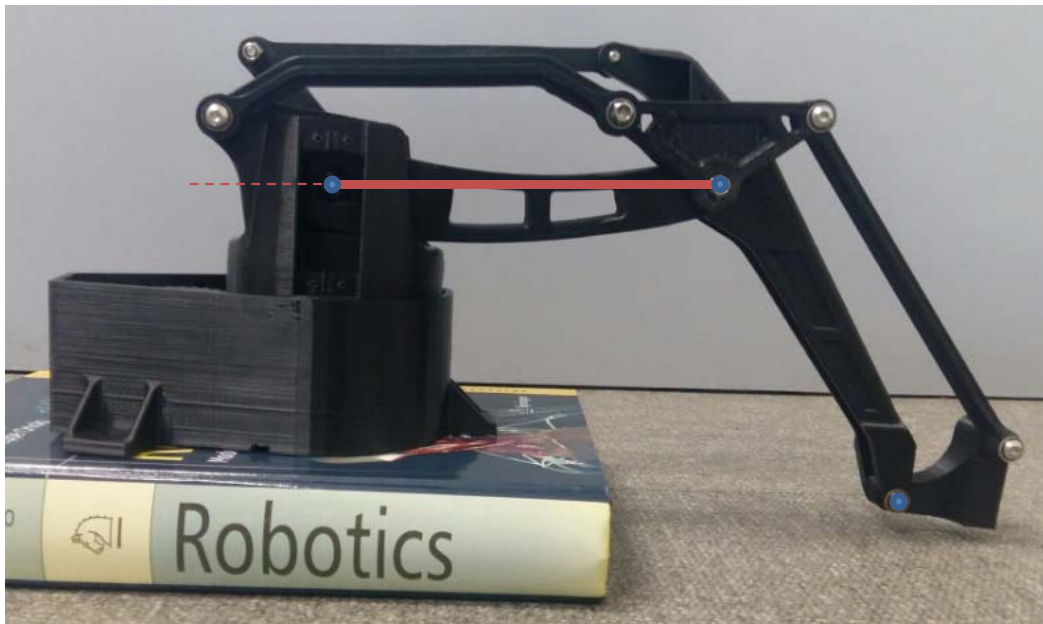
□ Step1. 영점조절

3.1 θ_1 회전범위의 중간지점에 기어를 끼운다.



□ Step1. 영점조절

3.2 θ_2 회전범위의 중간지점에서 혼을 끼운다.



0도



140도



중간지점 : 90도

□ Step1. 영점조절

3.2 θ_3 회전범위의 중간지점에서 혼을 끼운다.



-90도



13도



중간지점 : -45도

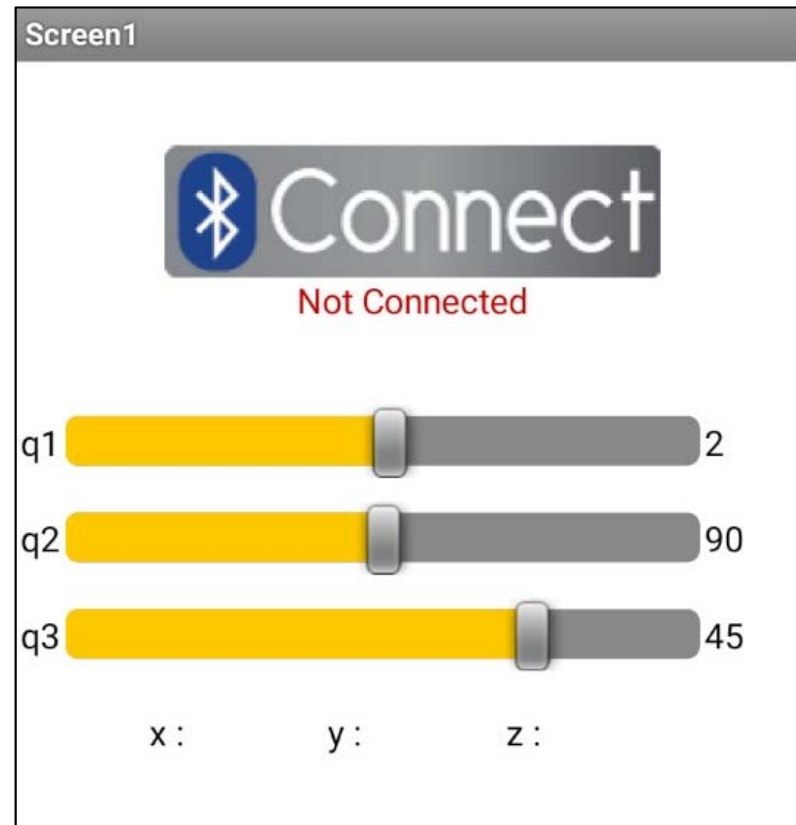
□ Step2. 각도 변환

```
#include <Servo.h>
```

```
Servo servo1;  
Servo servo2;  
Servo servo3;
```

```
void setup() {  
  servo1.attach(A0);  
  servo2.attach(A1);  
  servo3.attach(A2);  
  Serial3.begin(9600);  
}
```

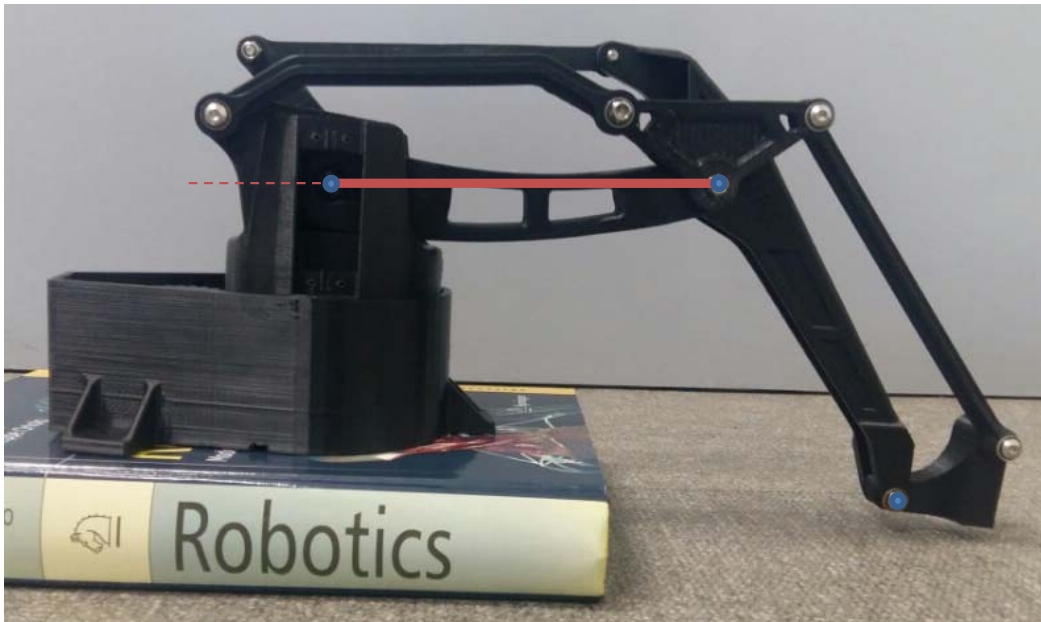
```
void loop() {  
  if(Serial3.available()){  
    int ang1 = Serial3.parseInt();  
    int ang2 = Serial3.parseInt();  
    int ang3 = Serial3.parseInt();  
    if(Serial3.read() == '\n'){  
      // 보정 및 변환  
      int s_val1 =   
      int s_val2 =   
      int s_val3 =   
      servo1.write(s_val1);  
      servo2.write(s_val2);  
      servo3.write(s_val3);  
      delay(15);  
    }  
  }  
}
```



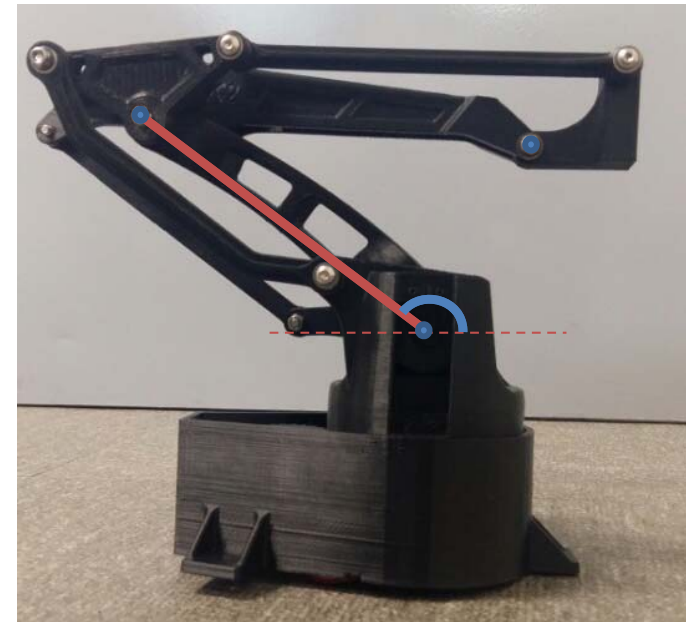
θ_1 : 기어비(50:22)

θ_3 : 45도 offset

□ Step3. 제한범위 설정(θ_2)



0도



140도

□ Step3. 제한범위 설정(θ_3)



-90도



13도

□ Step3. 제한범위 설정($\pi - (\theta_2 - \theta_3)$)



35도



144도

□ Step3. 제한범위 설정

```
#include <Servo.h>

Servo servo1;
Servo servo2;
Servo servo3;

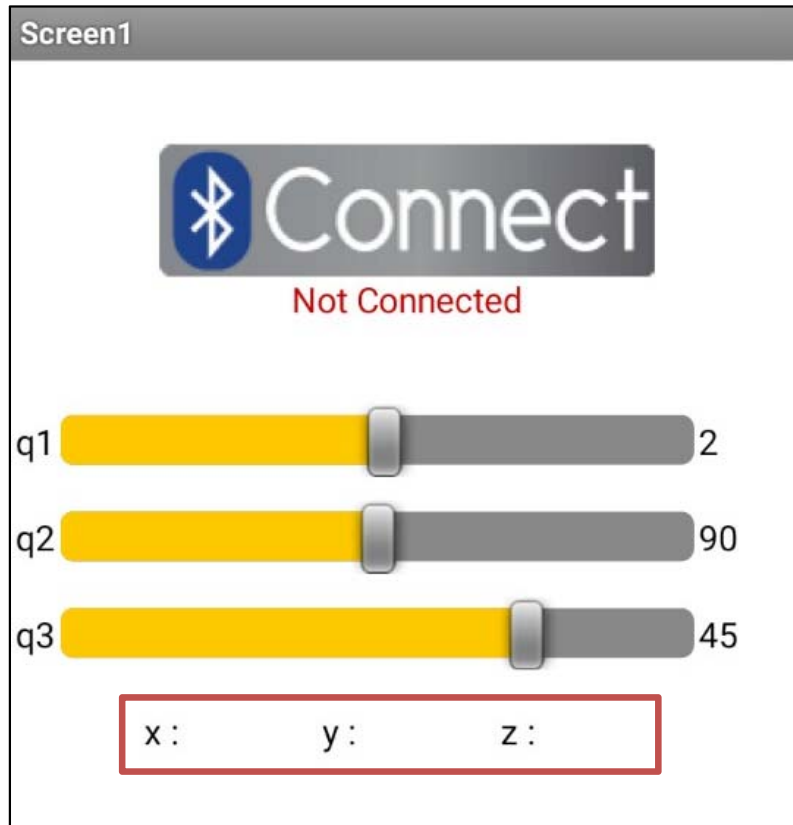
void setup() {
  servo1.attach(A0);
  servo2.attach(A1);
  servo3.attach(A2);
  Serial.begin(9600);
  Serial3.begin(9600);
}

bool isWorkspace(int ang1, int ang2, int ang3){
  /*YOUR CODE HERE*/
}
```

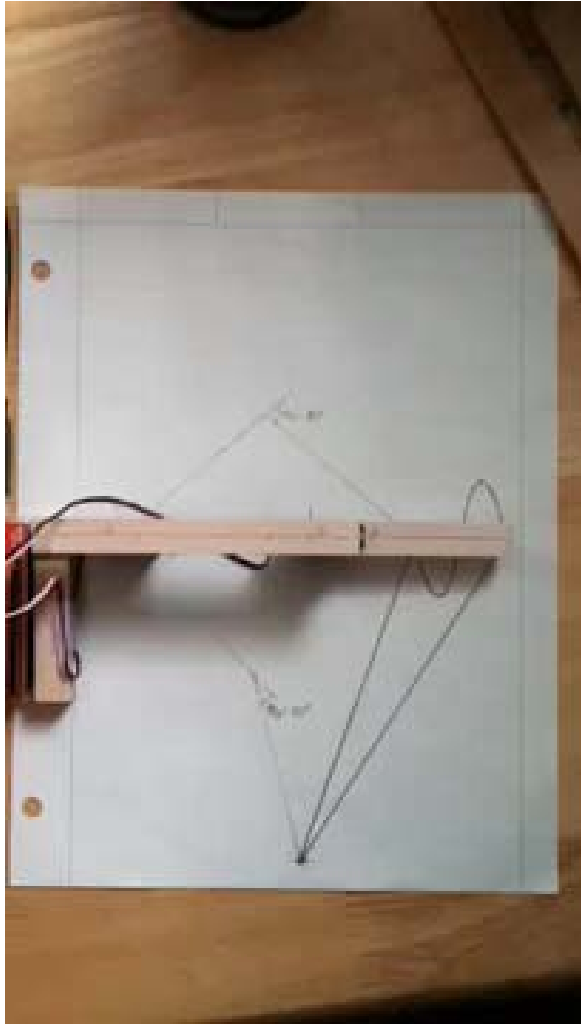
```
void loop() {
  if(Serial3.available()){
    int ang1 = Serial3.parseInt();
    int ang2 = Serial3.parseInt();
    int ang3 = Serial3.parseInt();
    if(Serial3.read() == '\n'){
      if(isWorkspace(ang1, ang2, ang3)==1){
        /*YOUR CODE HERE*/
        servo1.write(s_val1);
        servo2.write(s_val2);
        servo3.write(s_val3);
        delay(15);
      }
    }
  }
}
```

* slide의 min, max값은 수정하지 않을 것

□ Step4. Forward Kinematics

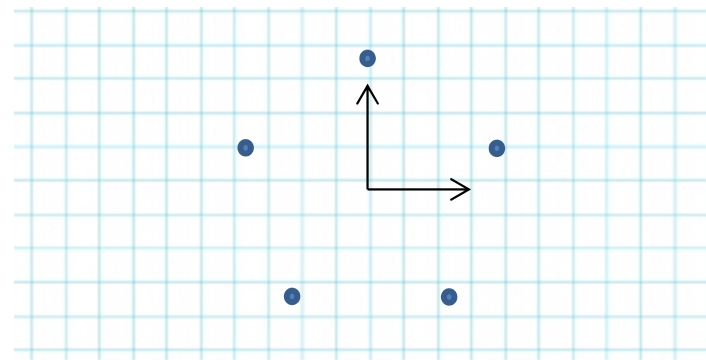


□ Make Star



- 로봇 팔의 end-effector로 별 모양 그리기
 - 구동 동영상 제출(시연 없음)
 - 아래와 같은 모눈종이에 별의 다섯 점의 좌표(측정값)과 로봇 팔의 각도로 부터 계산된 end-effector의 좌표 값 명시, 모눈종이 제출

※ end-effector에 펜을 달아 별 모양을 그릴 필요는 없음
※ 다섯 점을 이을 경우 경로가 직선이 아니어도 관계없으나 직선경로를 갖지 않는 이유를 적을 것.



□ 개별 레포트

1. 역할 분담

❓ 팀원 별 역할 수행 내용

❓ 기여도 : 점수 총합 10점 기준으로 자신 이외의 다른 팀원에 대한 점수 부여 (필수 사항 아님)

2. 실험 결과 및 분석

❓ 실험 목표

❓ 실험 과정

❓ 시행착오, 개선사항, 한계점

3. 소스코드

❓ Step4

❓ Make Star

※ 실험 결과 및 분석은 팀 내 공유 금지

□ 제출기한 : **6월 1일**

Term Project

□ 주제

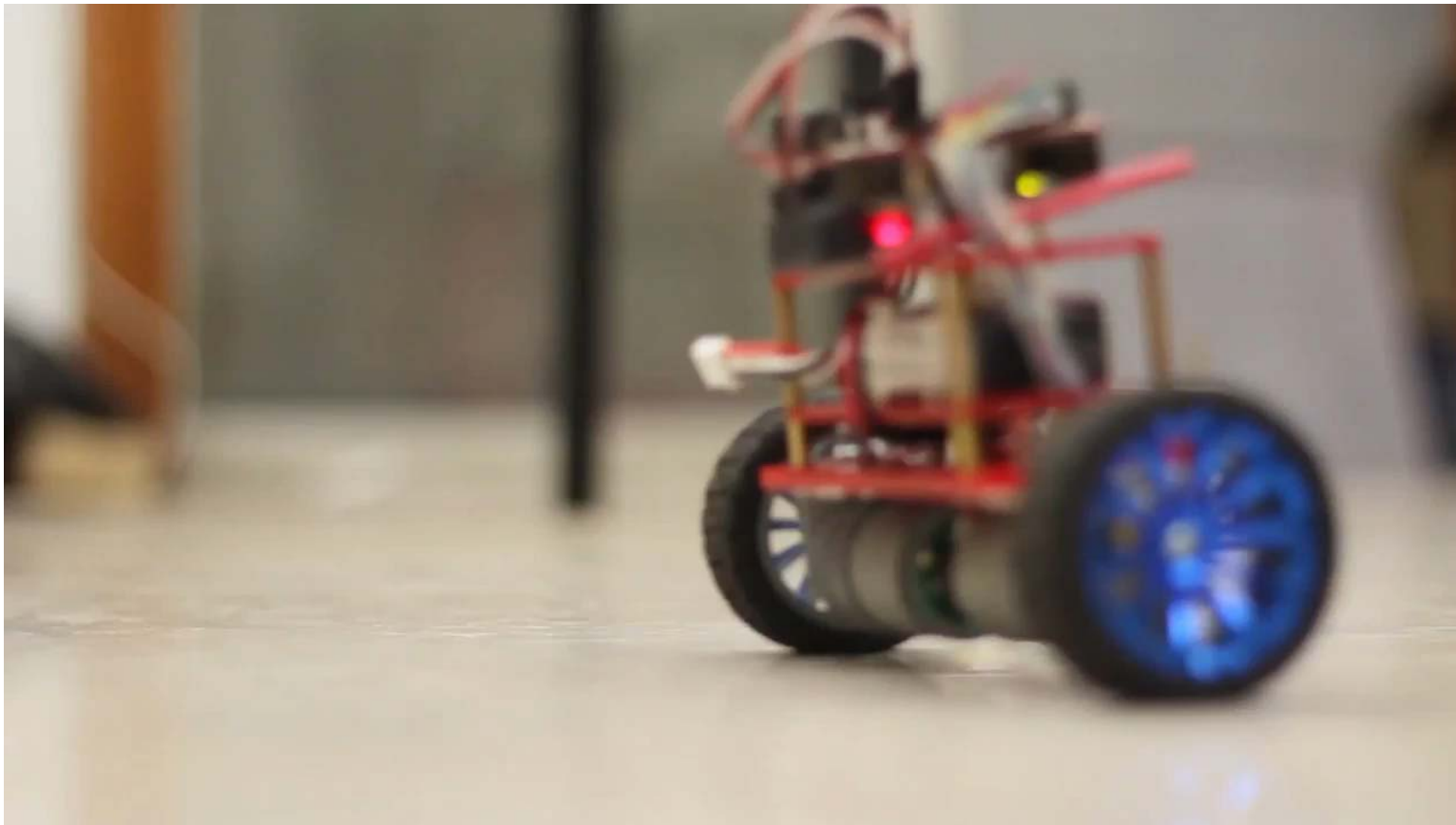
- 밸런싱 로봇
 - ❓ 자이로 센서, 하드웨어 제공
- 조종형 굴삭기 로봇
 - ❓ 조이스틱 2개 제공
- Slave-Master Robot Arm
- 기타 자유주제
 - 블루투스, CMU CAM, 자이로 센서 중 적어도 한 개 사용. 블루투스의 경우 어플을 이용할 필요 없음
 - 모바일 로봇 혹은 로봇 팔 이용
 - 모바일 로봇을 이용할 경우 주제 제한(기존의 것을 합치는 것과 다를 바 없는 주제는 허용하지 않음)
- 주의사항
 - ❓ 연구실에서 제공할 수 있는 부품의 경우 주제발표 후 제공, 나머지는 개별 구매
 - ❓ 3D 프린터를 이용하여 설계를 원하는 조는 **주제 발표 시 반드시 명시**

□ Term Project 주제 발표

- 내용
 - ❓ Term Project 주제
 - ❓ 관련 영상 및 자료
 - ❓ 필요한 부품
 - ❓ 수행 계획
- 날짜 : **6월 1일**

Term Project

□ 밸런싱 로봇



Term Project

❑ Slave-Master Robot Arm

