

인체의 움직임을 디지털 형태로 기록하는 모션 캡처 시스템 개발 중간 보고서

분과: C

Team: 박박

201824484 박해미

201824471 박민수

부산대학교 전기컴퓨터공학부 정보컴퓨터공학전공
School of Electrical and Computer Engineering, Computer Engineering Major
Pusan National University

2022년 8월 5일

지도교수: 정 상 화 (인)

목 차

제 1장 수정사항

1. 요구조건 및 제약 사항 분석에 대한 수정사항

제 2장 설계 및 구현의 상세화

1. 블루투스 메시
2. 시리얼 통신
3. 유니티 3D

제 3장 과제 수행 내용

1. STM32 보드
2. 유니티 3D

제 4장 개발 일정 및 역할분담

1. 갱신된 과제 개발 일정
2. 구성원별 진척도

1. 수정 사항

1.1 요구조건 및 제약 사항 분석에 대한 수정사항

1.1.1 하드웨어적 제약사항

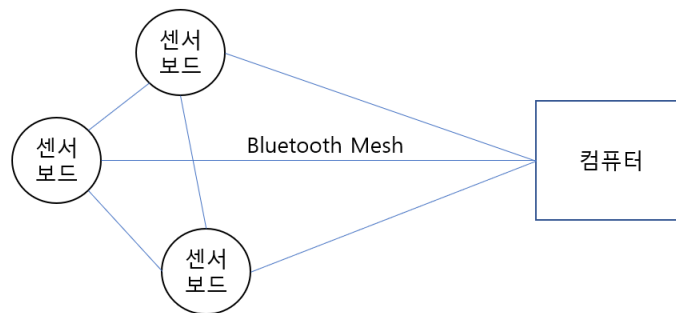
현재 보드의 생산량이 줄었기에, 해외 공식 사이트에서도 물량이 충분하지 않은 상황이다. 그로 인해서 인당 구매수가 제한이 되었기에 센서를 부착하여 움직임을 받아올 인체를 최대한 제한하였고, 그로 인해서 전신이 아닌 상체 위주의 그 중에서도 팔을 집중적으로 제한하여 구현되도록 개발할 예정이다.

Part Number	제조사	설명	재고	최소구매	가격	납기	
 NUCLEO-L476RG	STMicroelectronics	NUCLEO-64 STM32L476RG DEV EVAL 	1 개	1 개	23,700 원	4~6월	상품보기
 NUCLEO-L476RG	STMICROELECTRONICS	STMICROELECTRONICS - NUCLEO-L476RG - Development Board, STM32L476RG MCU, On-Board STLINK/V2-1, Arduino & ST Morpho Connectivity 	1 개	1 개	23,990 원	5~7월	상품보기

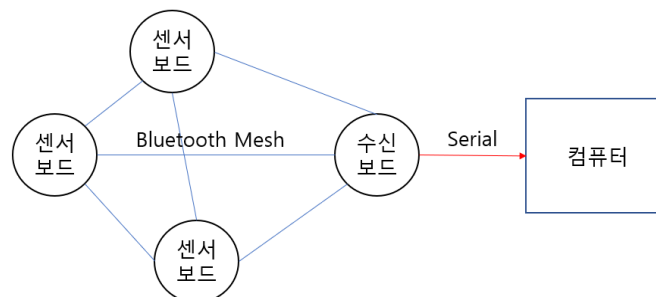
[그림 1. 구매 수량 제한(디바이스 마켓)]

1.1.2 설계 수정사항

기존 착수보고서 및 계획에서는 데이터를 받는 주체를 블루투스 기능이 있는 컴퓨터 및 노트북으로 정했다. 때문에 데이터를 받아 유니티 모션에 적용할 컴퓨터는 블루투스 메시망에 포함되어 있었다. 하지만 통신의 편이 및 개발 환경 상 한계로 특정 보드 하나를 데이터를 받는 주체로 정하고 이 보드에서 컴퓨터로 시리얼 통신하여 센서값을 보내는 방법으로 수정하였다. 기존 설계 방식에서는 수신 보드의 역할을 블루투스 디바이스가 처리하므로 구성상 기존 방식과 크게 달라지지 않는다.

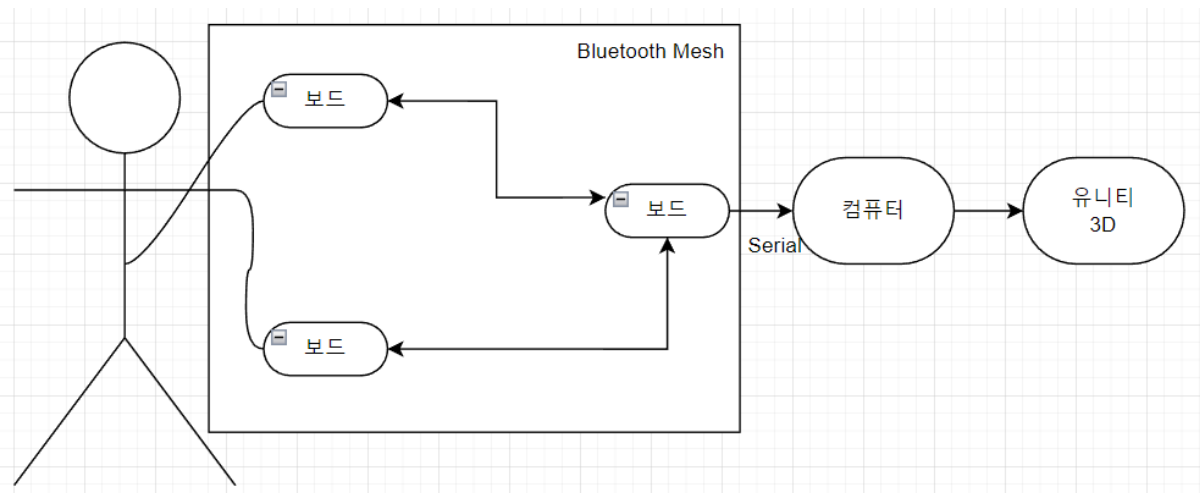


[그림 2. 기존 보드와 컴퓨터 연결 구조]



[그림 3. 변경된 보드와 컴퓨터 연결 구조]

2. 설계 및 구현의 상세화



[그림 4. 설계도]

2.1 블루투스 메시

인체에 직접적으로 장착될 센서 장치, 그 센서를 터미널로 옮겨주며 시리얼 통신을 할 장치, 이렇게 최소 세 개의 보드들로 **Bluetooth Mesh** 망을 만들어 연결한다.

이를 직관적으로 알기 위해서 **STM** 회사에서 제공하는 **ST BLE Mesh** 어플리케이션을 이용할 수 있다. 회사에서 제공하는 **BLE Mesh Lighting**, **FP SNS BLE Mesh**를 이용하여, **Bluetooth Mesh**를 구성한다.

이 때에, **publisher**인 노드는 인체에 장착되는 보드 두 개, **subscriber**인 노드는 컴퓨터에서 시리얼 통신을 해줄 데이터 수신기이다. 이 과정에서 어플리케이션이 둘의 **publish**와 **subscribe**를 도와줄 것이다.

2.2 시리얼 통신

```
static void Accelero_Sensor_Handler(uint32_t Instance)
{
    float odr;
    int32_t fullScale;
    IKS01A3_MOTION_SENSOR_Axes_t acceleration;
    displayFloatToInt_t out_value;
    uint8_t whoami;

    if (IKS01A3_MOTION_SENSOR_GetAxes(Instance, MOTION_ACCELERO, &acceleration))
    {
        snprintf(dataOut, MAX_BUF_SIZE, "\r\nACC[%d]: Error\r\n", (int)Instance);
    }
    else
    {
        snprintf(dataOut, MAX_BUF_SIZE, "\r\nACC_X[%d]: %d, ACC_Y[%d]: %d, ACC_Z[%d]: %d\r\n", (int)Instance,
            (int)acceleration.x, (int)Instance, (int)acceleration.y, (int)Instance, (int)acceleration.z);
    }

    printf("%s", dataOut);
}
```

수정 전

[그림 5. Accelero_Sensor_Handler() 의 수정 전 코드]

센서로부터 **Bluetooth Mesh**를 통해서 받아들이고, 이를 컴퓨터와 연결된 보드 내부에서 필터링을 하고, 데이터를 퓨전시킨다. 이렇게 퓨전시킨 데이터를 위와 같이 터미널에서 직관적으로 확인하는 작업을 할 수 있도록, 시리얼 통신을 통해서 터미널에 나타낸다.

터미널로 보여주는 필터링이 완료된 데이터들을 유니티 **3D**로 보낸다. 이 때에, 우리가 **publisher**로 사용하는 센서는 두 개이며, 두 센서들이 실시간으로 데이터를 전송하게 될 경우, 둘 중 어떠한 센서에서 보내는 데이터임을 알 수 없게된다. 이를 방지하여 파싱이 가능하도록 하기 위하여, 시리얼 통신 내부의 출력 구문을 아래의 그림처럼 수정하여 각 보드에 다르게 발드한다.

```
else
{
    snprintf(dataOut, MAX_BUF_SIZE, "\r\nBoard#1 ACC_X[%d]: %d, ACC_Y[%d]: %d, ACC_Z[%d]: %d\r\n", (int)Instance,
        (int)acceleration.x, (int)Instance, (int)acceleration.y, (int)Instance, (int)acceleration.z);
}

printf("%s", dataOut);
```

[그림 6. 수정 후의 출력 구문]

2.3 유니티 3D

인체의 움직임을 3D로 보여주기 위해서, 이를 표현할 수 있는 인체의 형태를 띄고 있는 무료 에셋, 로봇 카일(Robot Kyle)을 이용한다.

```
public class TestRigBones : MonoBehaviour
{
    public GameObject humanoid;
    private float x, y, z;

    public GameObject rightLowerArm;
    public GameObject rightUpperArm;

    // Start is called before the first frame update
    void Start()
    {
        x = 0;
        y = 0;
        z = 0;
    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.W))
            y += 10;
        if (Input.GetKeyDown(KeyCode.A))
            x += 10;
        if (Input.GetKeyDown(KeyCode.S))
            y -= 10;
        if (Input.GetKeyDown(KeyCode.D))
            x -= 10;

        Quaternion q = Quaternion.Euler(x, y, z);
        rightLowerArm.transform.localRotation = q;
    }
}
```

[그림 7. TestRigBones.cs]

센서 보드의 부착 위치를 오른팔의 어깨와 팔꿈치로 상정하고 프로젝트를 진행하였다. 시리얼 통신을 통해서 데이터를 받아들였을 경우에, 오른팔의 움직임을 주는 코드를 작성한다.

추후에 더 많은 보드를 이용하여 모션 감지를 진행하게될 경우, 이와 같은 코드를 추가하여, **GameObject**를 바꾸어 카일의 움직임을 더욱 늘릴 수 있다.

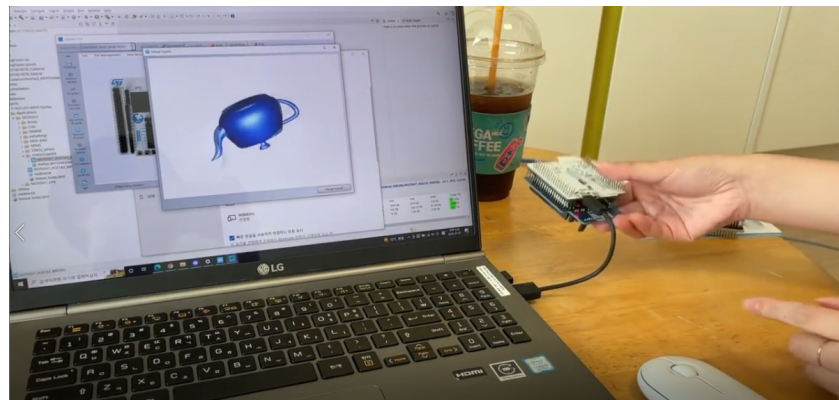
3. 과제 수행 내용

3.1 STM32 보드

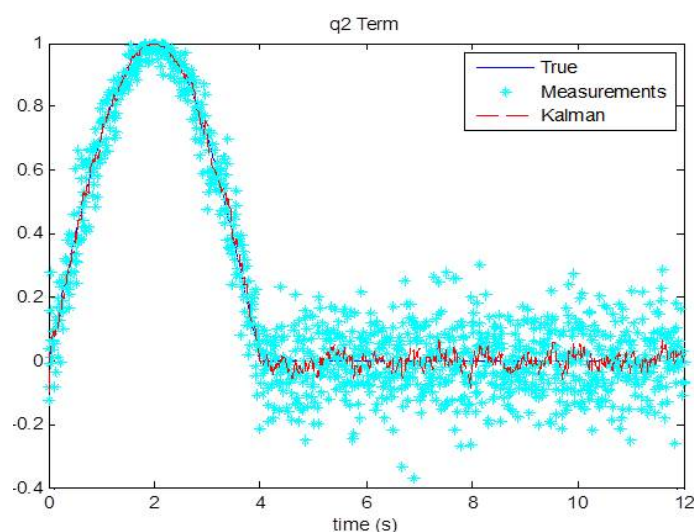
본 과제의 목표는 센서가 연결된 보드에서 센서값을 Bluetooth Mesh로 전달하여 유니티에서 모델을 조작하여 모션을 캡처하는 것이다. 때문에 기본 보드의 동작에 대해 이해할 필요가 있어 Stm32에서 제공하는 예제를 사용해보았다.

3.1.1 DataLogFusion 예제

DataLogFusion 예제는 9축 IMU 센서로부터 얻어지는 가속도 센서 값, 자이로 센서 값, 지자기 센서 값을 Sensor Fusion을 진행한다. 이때 그림에서 보듯이 Kalman filter를 이용해 IMU 센서 특유의 오차를 잡아내 정확도를 높인다. 이 결과는 Stm32에서 공식적으로 제공하는 SW인 Unicleo GUI에서 확인할 수 있다.



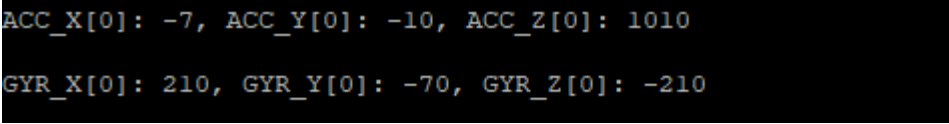
[그림 8. Unicleo GUI를 통한 Sensor Fusion 확인]



[그림 9. 기존 측정 값과 Kalman Filter 처리 후 데이터의 차이]

3.1.2 DataLogTerminal 예제

DataLogTerminal 예제는 DataLogFusion과 비슷하게 9축 IMU 센서로부터 얻어지는 값을 이용하지만 Sensor Fusion을 진행하지 않고 Virtual COM Port를 이용해 터미널로 그 값을 출력하는 예제다. 본 졸업과제는 Bluetooth Mesh로 컴퓨터와 연결된 보드에 값을 전송한 뒤, 보드에서 시리얼 통신으로 컴퓨터에 값을 전송하는 것이 목표이므로 본 예제를 통해 시리얼 통신으로 센서를 보내는 방법을 확인할 수 있다.



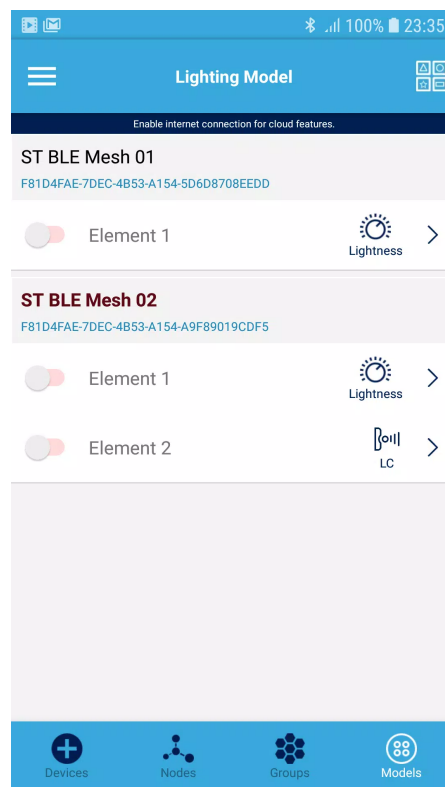
```
ACC_X[0]: -7, ACC_Y[0]: -10, ACC_Z[0]: 1010  
GYR_X[0]: 210, GYR_Y[0]: -70, GYR_Z[0]: -210
```

[그림 10. 터미널 출력문]

3.1.3 BLELighting 예제

BLE Lighting 예제는 Stm32에서 공식적으로 Bluetooth MESH를 경험하기 위해 Mesh 기술을 사용하여 여러 보드의 LED를 제어하는 예제이다. 이 예제를 통해 Bluetooth Mesh 사용법을 익힐 수 있다. BLELighting에는 3가지의 예제가 있는데, 그중 PRFNode와 Provisioner 예제를 사용했다. PRF Node는 Proxy, Relay, Friend 기능을 지원하는 노드를 생성하는데 사용하며 Provisioner는 현재 Mesh 망에 등록되지 않는 디바이스(Unprovisioner)를 등록하는데 사용되는 노드를 생성할 때 사용할 수 있다. 하나의 디바이스에는 Provisioner를 다른 디바이스에 PRFNode를 보드에 다운로드한다. 이후 Provisioner의 터미널을 통해 특정 명령어를 입력하면 근처에 Unprovisioner node를 검색하고 망에 등록할 수 있다. 이 예제는 LED를 제어하는 예제로 한 보드에서 LED를 키면 다른 보드도 LED가 켜지는 기능을 지니고 있다.

Provisioner 예제를 사용하지 않아도 PRFNode를 두 보드에 다운로드 시킨 후, STM32 공식에서 제공하는 ST BLE MESH 앱을 사용해 Mesh 망을 생성하고 보드를 조작할 수도 있다.



[그림 11. ST BLE MESH]

3.1.4 FP-SNS-Blemesh1 활용

FP-SNS-Blemesh1은 BLE Mesh Sensor Model 생성을 지원하며 보드와 연결된 컴퓨터에 Virtual COM를 통해 센서 값을 전달하거나 ST BLE MESH APP을 통해 센서값을 읽을 수 있다. 본 졸업과제는 BLE Mesh로 센서값을 전달한 뒤 컴퓨터와 연결된 보드에서 **Sensor Fusion** 후, 그 값을 시리얼 통신을 통해 컴퓨터로 전달하는 것이 목표이므로 FP-SNS-Blemesh1을 활용할 예정이다. 컴퓨터와 연결된 보드는 다른 센서 보드에서 얻은 값을 **Sensor Fusion**을 진행하고 그 값을 컴퓨터로 시리얼 통신을 통해 전달해야하며, 센서 보드들은 자신의 센서 값을 망에 있는 보드에게 전달해야한다. 이런 기능들은 위에서 진행한 예제를 활용해 구현할 예정이다.

3.2 유니티 3D

3.2.1 사용할 모델 선정



[그림 12. Space Robot Kyle]

모션 캡처에 사용할 3D 인체 모델은 유니티 공식 에셋스토어에서 제공하는 **Space Robot Kyle**을 사용할 예정이다. 모델에는 각 관절 부위가 모두 분리되어 있어 상세한 모션을 캡처할 수 있다.



[그림 13. Kyle 모델 구조]

4. 개발 일정 및 역할 분담

4.1 갱신된 과제 개발 일정

월요일을 기준으로 주를 나누어 일정을 정리하면 다음과 같다.

5월			6월				7월				8월					9월			
3주	4주	5주	1주	2주	3주	4주	1주	2주	3주	4주	1주	2주	3주	4주	5주	1주	2주	3주	4주
센서 보드 스터디																			
			모션 캡처 스터디																
			Unity 3D 스터디																
						스터디 내용 정리													
							알고리즘 설계												
								중간 보고서											
									BLE MESH망 구성										
									센서 간의 통신 데이터 변환										
									데이터 퓨전 처리										
											Unity와 연동								
															최종 점검				
																최종 보고서, 발표 심사 준비			

완료 , 진행 중 , 미진행 .

4.2 구성원별 진척도

이름	진척도
박해미	<ul style="list-style-type: none">• 유니티 3D 모델 기본 설정• 보드와 컴퓨터 사이의 시리얼 통신 구성
박민수	<ul style="list-style-type: none">• BLE Mesh 예제를 이용하여 분석• IMU 센서 예제를 이용해 Terminal 및 Fusion 분석
공동	<ul style="list-style-type: none">• 스터디 진행• 중간보고서 작성• 중간평가 시연 준비