2017학년도
기계종합설계 최종 보고서

과제명 : LHD 장비의 리모트 컨트롤 적용에 대한 연구

(2016년 9월 1일 ~ 2017년 6월 15일)

팀명: 스티브 짭스

기계공학 설계프로젝트 최종보고서를

붙임과 같이 제출합니다.

2017. 6

대구대학교 기계공학부(기계설계전공)

제 출 문

대구대학교 기계공학부 학부장 귀하

본 보고서를 대구대학교 기계공학부 설계프로젝트 과제 ‘ LHD 장비의 리모트 컨트롤 적용에 대한 연구(과제기간 : 2016. 09. 01 ~ 17. 06. 15)’의 결과보고서로 제출합니다.

2017. 6.

지도교수 : 김 홍 석 (인)

대표학생 : 원 세 진 (인)

참여학생 : 류 관 현 (인)

유 이 건 (인)

보고서 작성 윤리 서약서

대구대학교 기계공학부 학부장 귀하

본인은 보고서를 작성함에 있어 다음과 같이 연구 윤리 및 보고서 작성 윤리를 준수하였음을 서약합니다.

1. 본인은 다른 학생의 보고서를 복사(copy)하지 않았습니다.
2. 본인은 다른 사람의 보고서 내용 중 전부 또는 일부를 무단으로 도용하거나 인터넷에서 내려받기(download)하여 대체하지 않았습니다.
3. 본인은 보고서에 참고자료를 인용할 경우 원본의 출처를 반드시 표시하였습니다.

2017. 6.

대표학생 : 원 세 진 (인)

참여학생 : 류 관 현 (인)

유 이 건 (인)

목 차

최종보고 요약문

과제명	REMOTE CONTROL을 이용한 LHD 장비개발
팀명	스티브 짭스
팀원	원 세 진, 류 관 현, 유 이 건
과제기간	2016 년 9 월 1 일 ~ 2017 년 6 월 15 일

1. 개발내용 및 목표

‘ 위험한 광산, 안전 할 수은 없을까?’ 라는 생각을 가지고 연구를 해 보았다.
전 세계의 광산현장에서는 끊임없이 폭발 및 붕괴로 인하여 사망사고가 발생을 하고 있으며 광산현장의 위험성에 대해서 다시 한 번 알 수 있었다.
통계자료에 따르면 우리나라에서도 매년 광산사고가 일어나고 있다는 것을 알 수 있었으며, 이에 ‘광산현장은 안전하지 않다.’ 라는 결과를 알 수 있었고, 여기에 따른 해결방안으로 광산 안에서 적재와 하역을 동시에 수행하는 LHD(Loader + Haulage + Dump) 장비를 개발하여 인명피해가 없이 광산작업을 수행하는 것을 목표로 두었다.

2. 개념설계 및 상세설계

1) LHD의 목표를 구현하기 위한 구조 설계
2) Remote Control을 이용한 LHD 구동

3. 제작

1) Remote Control을 구현하기 위한 아두이노 실험 및 제작
2) LHD 장비 구동 목표를 위한 구조 설계 및 제작
3) 구조 설계 및 제작이 완료된 제품을 Remote Control을 이용한 구동

4. 시험 및 평가

1) 아두이노를 이용하여 서보모터 4개를 동시에 제어 할 수 있는지 평가
2) 기능부 1(Dump)의 가동범위에 이상이 없는지 평가
3) 기능부 2(Bucket)의 가동범위에 이상이 없는지 평가
4) 아두이노를 이용하여 기능부 제어(블루투스 송‘수신)가 이상 없이 작동하는지 평가

5. 세부 연구개발 내용 및 실적

1) LHD장비의 이름에 맞게 자가 적재가 가능한지에 대한 문제점 검토
2) 붐과 암 그리고 버켓이 덤프에 적재를 할 수 있게 할 수 있는 최적의 절점계산 및 버켓의 최대 적재량 계산

요약 1. 부품 및 제작비 사용내역

순번	부품 구매 및 제작 내용 상세	수량	소요예산(원)
1	한국형 아두이노 지니어스키트 스타터팩	1	116,000
2	감속기어모터	3	15,000
3	모터드라이브 모듈	2	5,500
4	블루투스 HC-06	7	12,500
5	샤오미 액션캠	1	160,000
6	개발용 원격 무한궤도	1	49,2000
7	리니어 서보 엑추에이터	5	744,000
8	무연납	1	31,000
9	인두기	1	59,100
10	절연선	1	1,300
11	아두이노 우노	6	12,000
12	아두이노 블루투스 조이스틱 모듈	4	24,000
13			
14			
총 액			1,672,400
예산지원 사업목록	• 기계공학부 실험실습비: 1,500,000 • LINC사업 캡스톤디자인 제작지원: 200,000 • 건설기계부품 특성화트랙 부품/시제품 제작비: 450,000 • CK-1 뿌리산업 특성화트랙 산학형종합설계 제작지원: 290,000		

요약 2. 설계구성요소 일람표

구 분		적용 내용	적용 여부	적용
설계구성요소	설계 목표 설정	비젼부를 구동부에 부착시켜 사고로 인한 인명 피해를 줄이고, 구동부에 기능부 Dump와 Bucket을 장착시켜 적재, 하역, 운반을 동시에 수행을 할 수 있게 목표	○	1.2절 pp. 9
	합성	1. 기구학 적으로 기능부 2(Bucket)이 스스로 자가 적재를 할수 있는지 파악 2. 서보모터로 버켓이 물체를 담아서 들 수 있는지 파악	○	1.2절 pp. 28~34
	분석	Remote Control의 보편화, 시장성 면에서도 완성도가 높음, 기존제품에 적용되어있는 유압과 기구학적인 형상을 더욱 응용하여 제품에 보다 좋은 기능을 추가하기 위한 기술 분석, 쾌적한 환경에서도 작업 할 수 있음으로 작업자의 능률 변화	○	1.1절 pp. 5~6
	제작	1. 구동부와 비젼부는 완제품을 구입 2. 버켓은 3D프린트로 제작 3. 덤프는 아크릴판을 레이져 커팅기로 제단 후 제작 4. 아두이노를 이용해서 기능부 리모컨을 별도로 제작하여 기존에 있는 구동부 리모컨에 결합	○	3.1절 pp. 35~41
	시험	1. 적재, 하역, 운반이 동시에 작동 하는지 시험 2. 보이지 않는 곳에서도 간접적으로도 사용이 가능한지 시험	○	4.1절 pp. 42
	평가	1. LHD장비가 적재, 하역, 운반이 동시에 가능한지 평가 2. 목표가 사람 인명피해를 줄이는 장비를 개발 하는 것인 만큼 보이지 않는 곳에서도 작동을 할 수 있는지 평가	○	4.1절 pp. 42~43

요약 3. 현실적 제한조건 일람표

구 분		적용 내용	적용 여부	적용
현실적제한조건	원가	중장비이고 덤프와 굴삭기 기능이 한꺼번에 탑재가 되어서 가격 부분에서는 비용이 나가겠지만, 두 장비를 구입한 가격 인건비, 안정성과 LHD장비를 구입 한 비용과 인건비, 안정성을 생각해보면 LHD장비가 비용 부분에서는 훨씬 저렴하다.	○	1.1절 pp. 5
	안전성	제품을 설계를 하고 제작을 완료한 뒤에 간단한 동굴 효과를 낼 수 있는 미로에서 오로지 비젼부 만을 통해 작동을 하며 구동과 적재, 하역을 동시에 수행하는 것을 성공적으로 해내었고 이로써 인명피해를 줄 일 수 있고, 그러면 안전성도 보장이 된다.	○	4.1절 pp. 42
	신뢰성	제품을 설계를 하고 제작을 완료한 뒤에 간단한 동굴 효과를 낼 수 있는 미로에서 오로지 비젼부 만을 통해 작동을 하며 구동과 적재, 하역을 동시에 수행하는 것을 성공적으로 해내었다.	○	4.1절 pp. 42
	윤리성	우리나라 광산현장부터 시작해서 세계 광산현장까지 광산사고로 인한 인명 피해를 줄이고 작업자의 효율을 늘리기 위함이다.	○	1.1절 pp. 5
	미학	기존 중장비의 형상에서 크게 벗어나지 않는 형상이고 화려한 디자인은 아니지만 성능과 기능에 충실하다는 느낌을 주기 위해서 블랙 화이트로 디자인 하였다.	○	2.2절 pp. 41
	사회에 미치는 영향	원격 중장비가 보편화 된다면 더 이상 광산현장에서의 인명피해는 없을 것이다. 사각지대가 사라지고 중장비로 위험한 작업을 수행하더라도 그 범위에서 벗어나 작업함으로 작업자 자신의 신변의 안전을 보장받을 수 있게 된다.	○	1.1절 pp. 5

제 1 장 설계목적 및 목표

제 1 절 목적 및 필요성

‘ 위험한 광산, 안전 할 수은 없을까?’ 라는 생각을 가지고 연구를 해 보았다.
전 세계의 광산현장에서는 끊임없이 폭발 및 붕괴로 인하여 사망사고가 발생을 하고 있으며 광산현장의 위험성에 대해서 다시 한 번 알 수 있었다.

<그림 1-1> 광산 사고 사례

아래 (표1-1)의 통계자료와 같이 우리나라에서도 매년 광산사고가 일어나고 있다는 것을 알 수 있다. 2005년부터 2014년 까지 통계자료인데 평균적으로 매년 53.3번의 사고가 발생하고 있다. 이에 ‘광산현장은 안전하지 않다.’ 라는 결과를 알 수 있었고, 여기에 따른 해결방안으로 광산 안에서 적재와 하역을 동시에 수행하는 LHD(Loader + Haulage + Dump) 장비를 개발하여 인명피해가 없이 광산작업을 수행하는 것을 목표로 두었다.

<표 1-1> 광산 작업 중 사고 발생 현황

구분	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014.6	계
낙반,붕락	11	16	28	25	18	12	18	32	20	5	185
가스,폭발	0	0	0	0	0	0	0	5	0	0	5
발파,화약	4	2	4	2	4	1	1	1	0	0	19
운반	13	10	28	14	16	12	4	11	17	4	129
기계,전기	4	3	7	5	6	7	10	3	6	3	54
추락,전도,전석	4	2	8	8	11	10	8	5	9	1	66
기타	9	10	10	5	9	15	7	3	5	2	75
계	**45**	**43**	**85**	**59**	**64**	**57**	**48**	**60**	**57**	**15**	**533**

구분	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014.6	계
사망	5	1	6	4	10	7	5	9	6	3	56
중상	18	14	41	29	25	24	24	25	27	9	236
경상	22	28	38	26	29	26	19	26	24	3	241
계	45	43	85	59	64	57	48	60	57	15	533

제 2 절 과제의 목표

이번 프로젝트의 목표는 광산을 폭파시켜서 만들어 낸 공간은 결코 안전하다고 볼 수가 없으며, 광산 안에서 적재와 하역을 동시에 수행하는 LHD장비를 Remote Control로 조종을 하여 전 보다 광산 현장이나 위험한 현장에서 인명 피해를 줄이고 작업자들의 작업환경과 작업능률을 늘릴 수 있게 함을 이번 프로젝트의 목표를 두고 작업을 하였다.

제 3 절 기대효과 및 활용방안

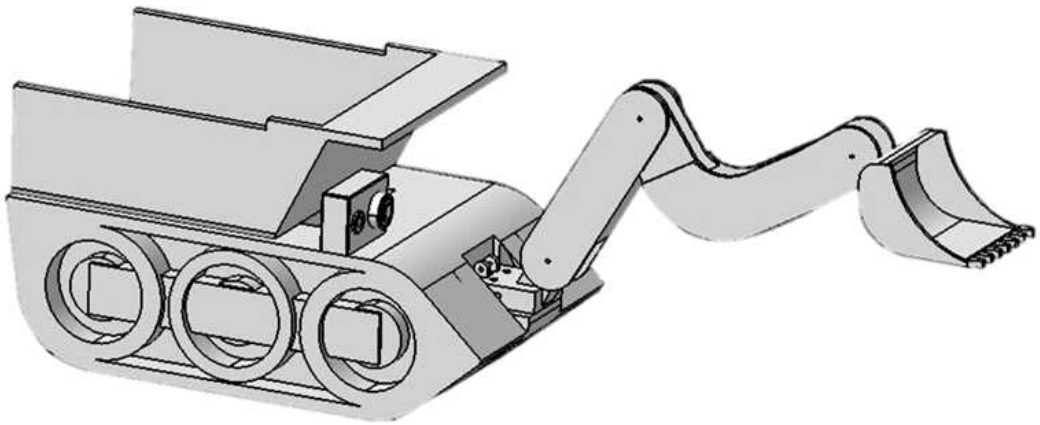
Remote Control를 사용함으로써 총 3가지를 중점으로 두었다. 첫 번째는 안정성 두 번째로는 작업성 세 번째로는 효율성 크게 총 3가지를 기대효과로 두었는데 우선 안정성으로는 간접적으로 탄광 작업을 함으로써 많은 사고로부터 안전하여 인명피해를 최소화 시킬 수가 있고 다음 작업성으로는 보통 탄광 작업을 하면 사람들이 빛도 못보고 몸에 안 좋고 목숨을 보장 받을 수 없는 작업 환경으로부터 벗어나 작업을 Remote Control로 작업을 함으로써 얻게 되는 보다 나은 작업 환경으로 얻게 되는 건강과 목숨을 보장 받을 수 있다는 정식적인 문제점도 개선을 할 수가 있어 작업자들이 작업 능률이 보다 늘어날 수가 있고 마지막으로 효율성으로 앞서 말했듯이 위험해서 작업을 할 수 없었단 중장비의 사각지대가 사라짐으로 작업량을 증가시킬 수가 있다.

제 2 장 개념설계 및 상세설계

제 1 절 개념설계

LHD장비를 간단히 설명을 하자면 적재를 하는 L(Loader) 운반을 하는 H(Haulage) 하역을 하는 D(Dump) 광산현장에서 일정한 구간을 반복적으로 적재, 운반, 하역을 수행하는 것을 말한다. 운반기능을 수행하는 구동부, 영상을 수신 할 수 있는 비젼부는 현재 시중에 보편화가 되어있고 쉽게 구할 수가 있어 자체 개발 제작하는 것은 시간적 여유가 부족할 뿐만 아니라 완성도면에서도 많은 차이가 있을 것이라 보여 완제품을 구입을 하였고. 굴삭, 적재를 할 수 있는 기능부1은 버켓, 암, 붐으로 나누어 유압 실린더를 대체할 서보 엑추에이터가 암과 붐 사이의 공간으로 3개를 내장시켜 작동할 수 있게 하였다.

하역을 하는 기능부2는 구동부에 핀으로 연결을 하여 고정을 시켰고 서보 엑추 에이터를 1개를 사용하여 하역을 수행 할 수 있게 만들었다.그리고 이 모든 기능부는 아두이노를 이용하여 구동 시킬 수 있어 적재, 운반, 하역을 동시에 수행할 수가 있다.



<그림2-1> 개발 목표인 LHD 개념도

제 2절 상세설계 및 보완

구동부는 아두이노와 블루투스 통신으로 DC모터를 제어해 제작할 계획이었지만 제작기간 단축을 위해서 개발용 및 실험 플랫폼으로 시중에 판매되는 궤도를 구입하게 되었다. 제작기간 단축뿐만 아니라 완성도 면에서도 확연한 차이가 있을 것이다.

1) 구동부

<그림 2-2> 구동부 모듈의 형상

<표 2-1> 구동부 모듈의 치수

크기	190 X 295 X 110 (mm)
무게	3.7kg
배터리 함 크기	90 X 180 X 40 (mm)
궤도 폭	40mm
배터리 용량	5200mA
배터리 종류	리튬 이온 배터리
전력	12V

2) 비젼부

비전부 또한 완제품을 구입 하였다. 제품은 샤오이 액션캠으로 블루투스를 이용해 스마트폰에 실시간 영상을 제공한다. 이 액션 캠으로 원거리에서 실시간 영상을 통해 서보 엑추에이터를 제어할 수 있다.

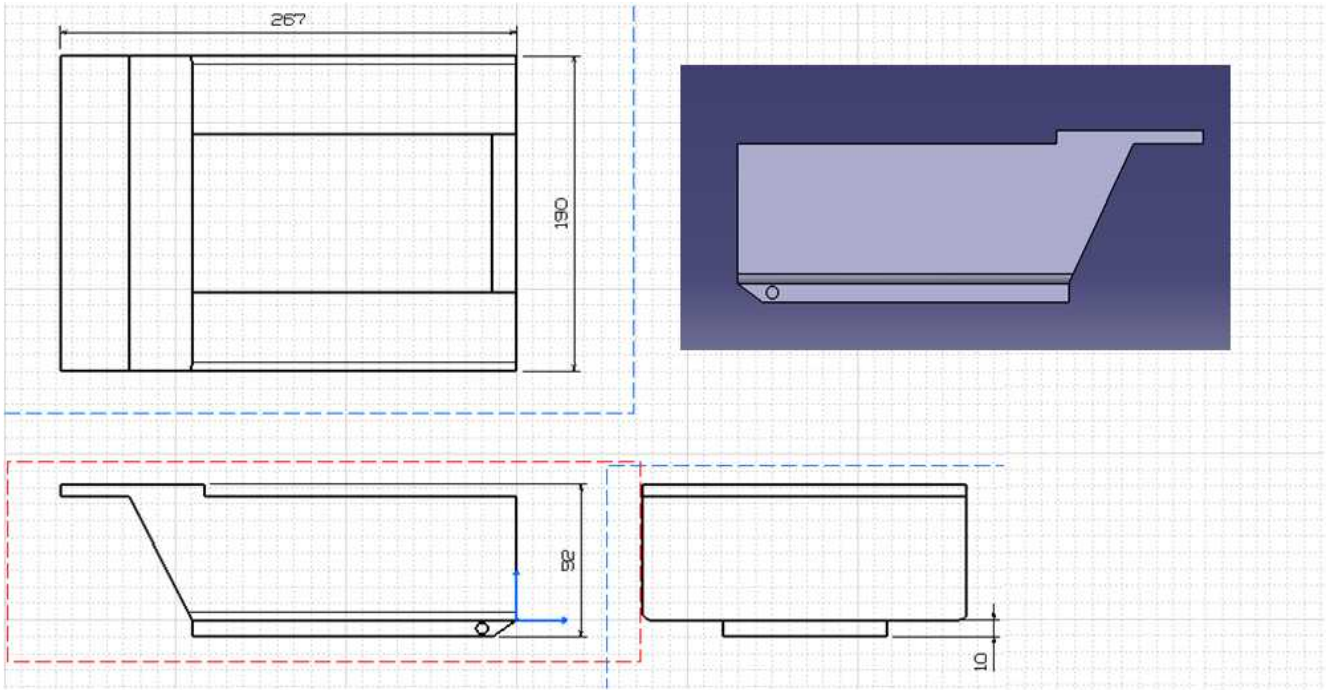
<그림 2-3> 비젼부 모듈의 구성도

<표 2-2> 비젼부 모듈 개요

제품명	샤오이 액션캠
해상도	1080p 60fps지원
무게	72g
배터리 사양	3.7V 1.010mAh
지원플렛폼	스마트 폰 원격제어가능 동영상 및 사진 촬영 가능

<그림 2-4> 비젼부 모듈의 장착 형상

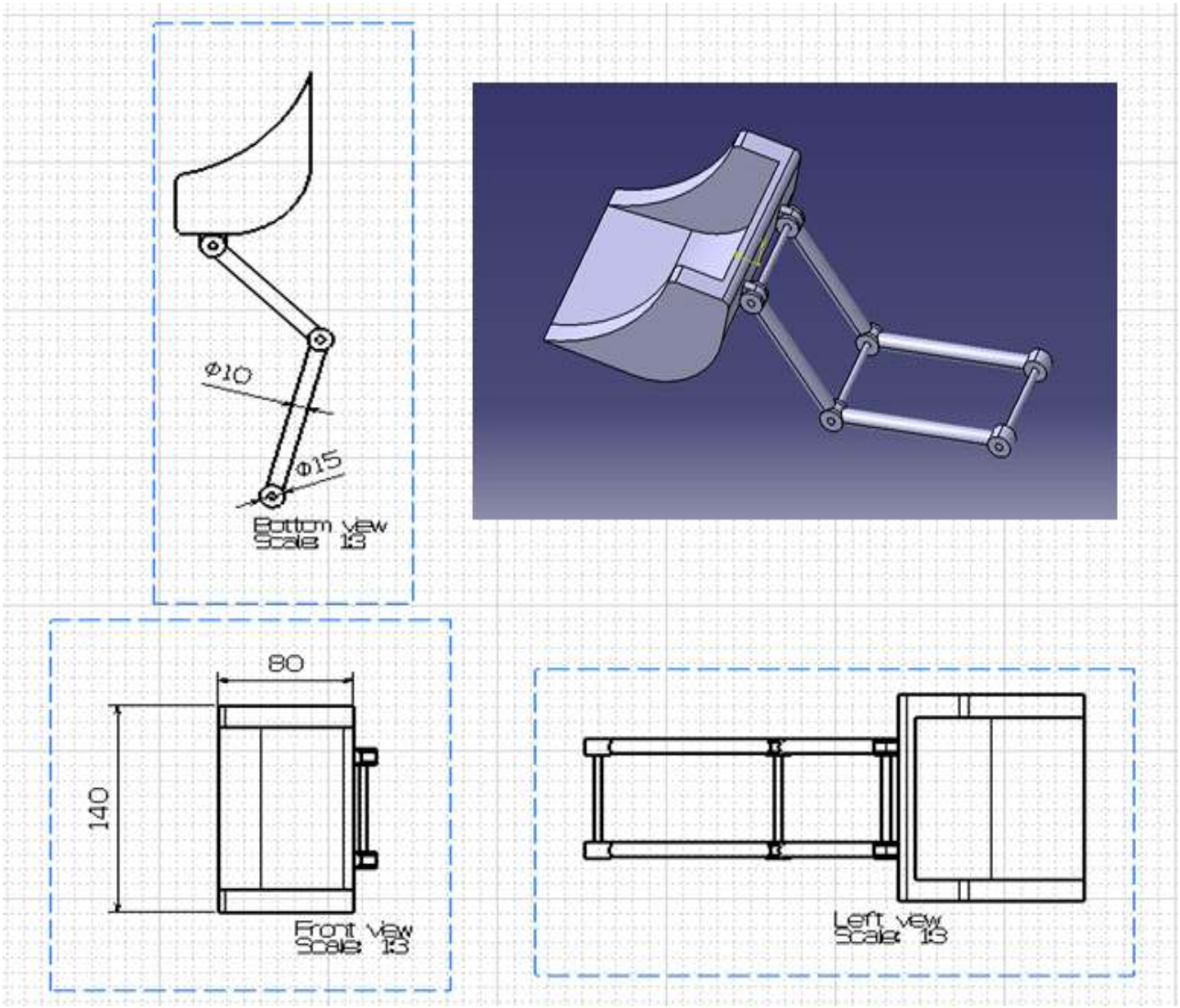
3) 기능부 1(Dump)



<그림 2-5> 기능부 덤프의 형상

덤프의 크기는 190mm X 267mm X 92mm 로 선정을 하였고 덤프의 최대 적재량은 0.0047 mm^3로 선정 하였다. Dump의 재질은 아크릴 판으로 제작되었고 가공은 레이저컷팅기로 가공하였다. 구동부와 Dump사이에 핀을 연결하고 서보엑추에이터가 상하 운동을 함으로서 Dump가 작동된다.

4) 기능부 2(Bucket)



<그림 2-6> 기능부 버킷의 형상

초기 도안은 각 관절마다 서보모터를 3EA를 이용하여 기능부를 제어하기로 계획을 해보았지만 Bucket에 작업물을 담아서 들어올리기 위한 힘을 가지려면 서보모터가 자동으로 크기가 커지게 됨으로 초기도안에서 수정하기로 했다.

<수정된 도안>

<그림 2-7> 버킷의 설계 변경

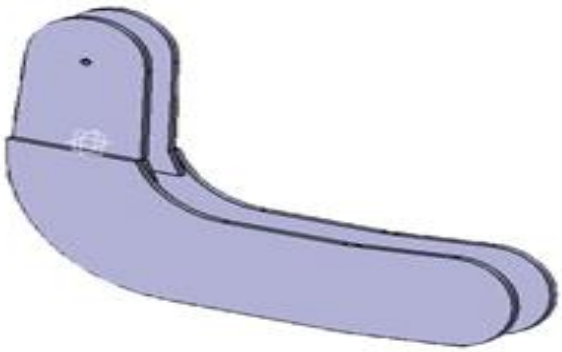
수정된 도안은 중장비에서 많이 쓰는 유압실린더로 작동하는 방식으로 구동시키기 위해 서보 엑추에이터를 쓰기로 했다. 같은 서보모터 이지만 회전운동을 직선으로 변환 시켜 작은 모터로 큰 힘을 낼 수 있는 모터이다. 암(Arm)과 붐(Boom)으로 구성되어 있고 사이 공간에 서보 엑추에이터가 들어가 원하는 운동을 하게 된다.

그림(2-8)

<그림 2-8> 버킷의 기구적 형상(http://www.hanaro-tech.com)

① 암(Arm)

암은 버켓과 붐사이를 연결하며 서보엑추에이터에 의해 버켓을 구동하게 만드는 역할을 한다. 크기는 폭 20mm 가로 높이로 선정 했다.



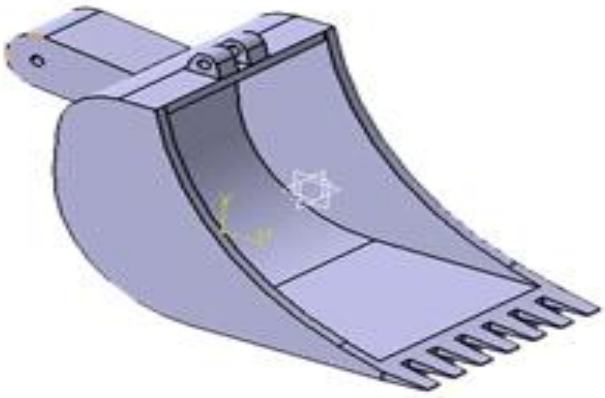
<그림 2-9> 설계 변경된 암(Arm) 형상

② 붐(Boom)

붐은 암과 굴삭기의 몸체를 연결하며 서보엑추에이터 의해 암이 구동하게 만드는 역할을 한다. 또한, 굴삭이나 무거운 제품을 운반 시, 하중을 지지하는 중요한 역할을 수행 한다. 암과(Arm)과 같이 폭은 20mm이고 길이는 mm 높이는 mm 로 선정했다.

<그림 2-10> 설계 변경된 붐(Boom) 형상

③ 버켓(Bucket) 적재 용량의 계산



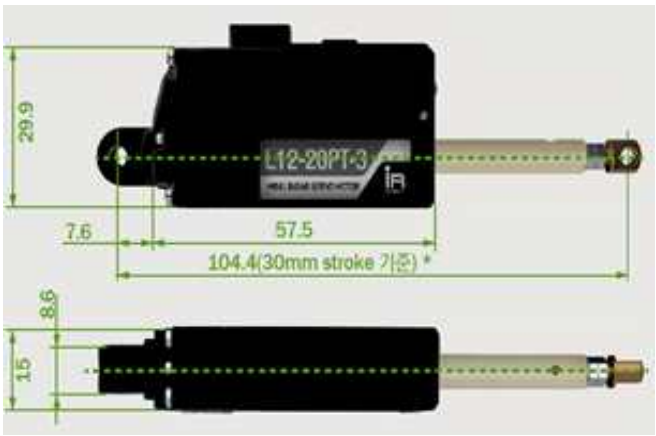
<그림 2-11> 설계 변경된 붐(Boom) 형상

버킷의 적재 용량을 계산 하기 위해 우선 버킷 각도를 계산하였다. 버킷의 각도는 $\tan\alpha = \frac{h}{b}$에서 구할 수 있는데, 여기서 h는 높이 b는 폭을 말한다. 버킷의 높이와 폭을 적용하면 각도 α는 33.7° 가 나오게 된다. 따라서 버킷의 적재 용량은 (1)식으로 계산할 수 있다.

$$q = \frac{1}{2} \times \frac{h^2}{\tan\alpha} \cdot L \cdot E \tag{1}$$

(1)식에서 L은 길이, E는 평상시의 평균 작업효율인데, 버킷 각도를 계산해 나온 값을 대입해서 계산하면 $17.21cm^3$, $17.21 \times 10^{-6} m^3$ 값이 나오게 된다.

암(Arm), 붐(Boom), 버켓(Bucket) 모두 3D프린터로 제작될 예정이며 재질은 플라스틱이고 각 조인트마다 핀으로 연결해 서보모터와 결합될 예정이다. 기능부 1, 2를 제어할 서보엑추에이터의 기본 사양은 다음 그림에 나타낸 바와 같다.



<그림 2-12> 기능부에 적용될 서보엑추에이터

<표 2-3> 서보엑추에이터의 사양

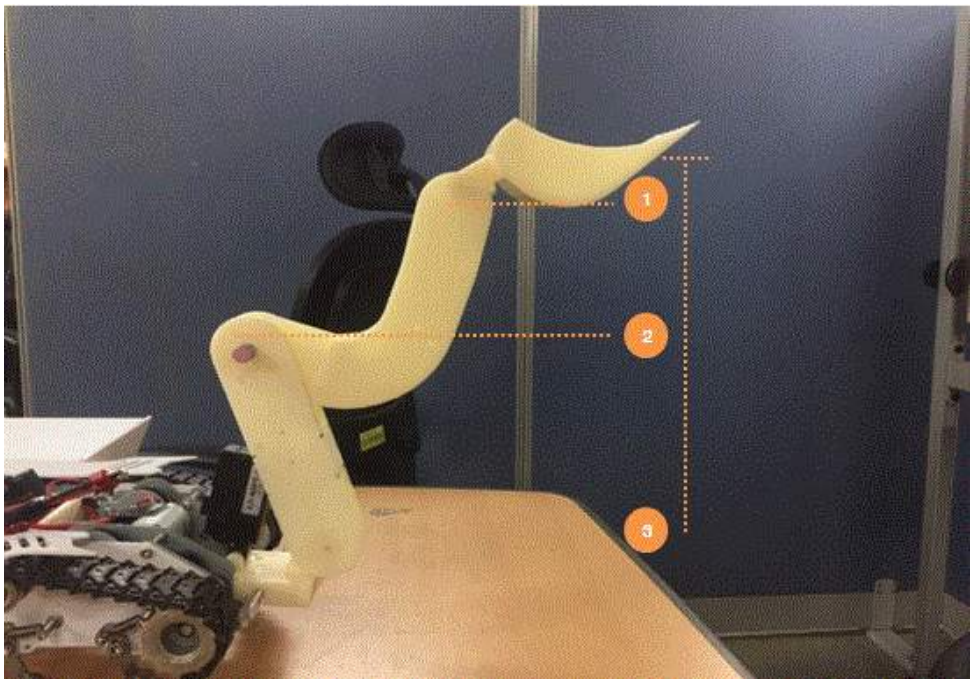
정격 전압	7.4V
출력	40N
특징	서보 모터의 회전 운동을 직선 운동으로 변화시킨 전동 피스톤으로 크기에 비해 출력이 좋다.

위 표(2-3)의 서보엑추에이터의 출력과 Bucket적재용량을 계산한 결과값을 이용해 토크값을 구할 수 있다. 모멘트 공식인 M=F • d을 적용하는데, 여기서 F는 하중 d는 길이를 나타낸다. 첫 번째 길이는 Bucket만의 길이 두 번째는 Bucket에서 Boom까지 세 번째는 Bucket에서 Arm까지의 길이를 계산한다.

첫 번째 토크값은 $\frac{40N\times125mm}{1000}=5N\bullet m$ 이 나오고

두 번째 토크값은 $\frac{41.55N\times250mm}{1000}=10.39N\bullet m$, 마지막으로

세 번째 토크값은 $\frac{42.45N\times390mm}{1000}=16.56N\bullet m$ 이 나온다.



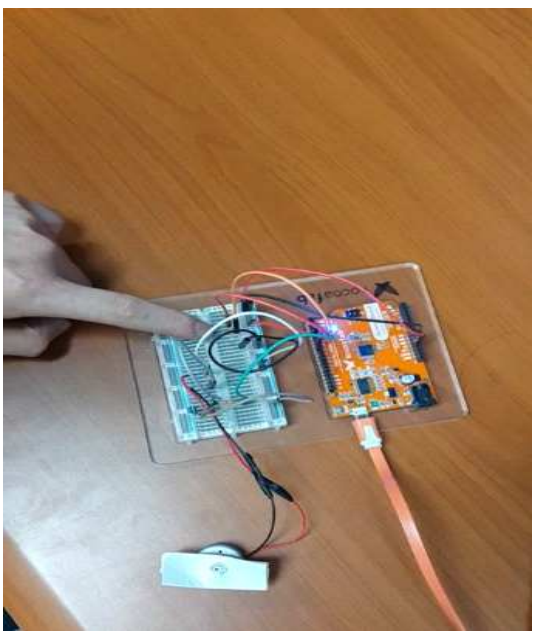
<그림 2-13> 버킷부의 모멘트 계산

5) 기능부 제어(Arduino)

기능부 제어는 아두이노 우노와 아두이노 나노의 블루투스 통신으로 제어가 이루어진다. 아두이노란 비전공자도 쉽게 접근할 수 있게 만든 킷트 개념의 기판이라고 할 수 있다. 오픈소스도 온라인상에서 쉽게 얻을 수 있으며 약간의 c언어를 다룰 줄 안다면 얼마든지 자신이 원하는 기기를 만들어 낼 수 있다는 장점이 있다. 하지만 조원들 중에서는 제어부분에 대한 지식이 전혀 없는 상태였기 때문에 기초부터 공부를 하고 서보엑추에이터 4개를 동시에 블루투스 통신으로 제어하기 위해서는 충분한 실험과 연습 과정이 필요했다. 우선 DC모터를 on off하는 기초적인 실험부터 해보았다.

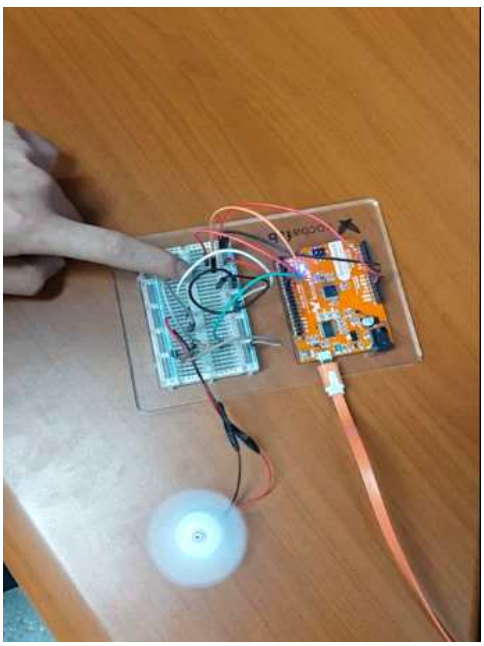
<그림 2-14> 기능부 제어에 활용된 아두이노우노

<실험-1> DC모터 제어



(off)

→



(on)

<소스코드-1>

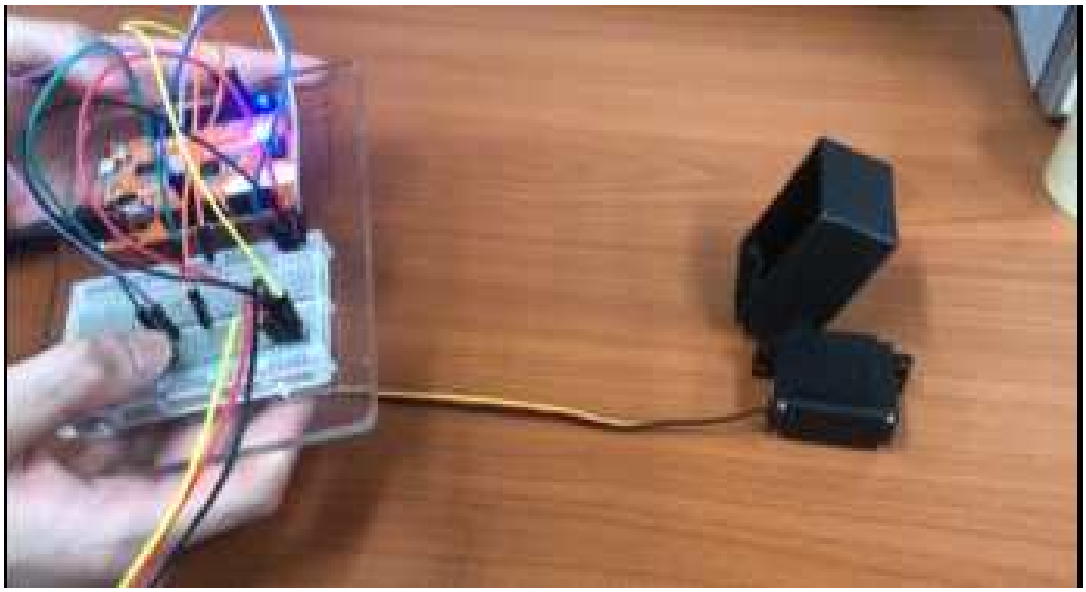
```
// DC모터를 3번 핀으로 설정합니다.
int motor = 3;
// 스위치를 6번 핀으로 설정합니다.
int sw = 6;

// DC모터의 회전 속도를 정의합니다.
// 디저털 핀으로 0~255 범위의 값으로 DB모터의 속도를 제어할 수 있습니다.
// 모터가 제대로 동작하지 않는다면, speed 값을 상향조정합니다. (예 200)
int speed = 127;

// 실행시 가장 먼저 호출되는 함수이며, 최초 1회만 실행됩니다.
// 변수를 선언하거나 초기화를 위한 코드를 포함합니다.
void setup() {
	// DC모터가 연결된 핀을 OUTPUT으로 설정합니다.
	pinMode(motor, OUTPUT);
	// 스위치가 연결된 핀의 모드를 INPUT_PULLUP 상태(초기 로직레벨을 HIGH로 설정)로 설정합니다.
	// 설정된 디지털 핀은 아래와 같은 값을 반환합니다.
	// 스위치가 열려있다면 (누르지 않은 상태) HIGH
	// 스위치를 닫혀있다면 (누른 상태), LOW
	pinMode(sw, INPUT_PULLUP);
}

// setup() 함수가 호출된 이후, 호출되는 함수입니다.
// 블록 안의 코드를 무한히 반복 실행합니다.
void loop() {
	// 스위치가 연결된 핀의 로직레벨이 LOW라면,
	// 스위치가 닫혀있는 상태(누른 상태) 이므로, 아래의 블록을 실행합니다.
	if (digitalRead(sw) == LOW) {
		// DC모터가 연결된 핀으로, 지정된 speed로 회전하도록 설정합니다.
		// 본 예제에서 사용된 127이란 값은, 디지털 핀으로 출력 할 수 있는 최대값 255의 절반에 해당되므로,
		// DC모터가 5V 전류로 낼 수 있는 최대 회전 속도의 절반으로 해석 할 수 있습니다.
		// 이는 디지털로 아날로그 신호를 보내는 펄스 폭 모듈레이션(PWM)에서 duty-cycle이 50%인 것으로 설정됩니다.
		// 오렌지보드 디지털 핀의 PWM 주파수를 약 500Hz로 가정하면, 1초에 255번만 HIGH 신호를 보내는 것과 같습니다.
		analogWrite(motor, speed);
```

```
        }
        // 스위치가 연결된 핀의 로직레벨이 HIGH라면,
        // 스위치가 열려있는 상태(누르지 않은 상태) 이므로, 아래의 블록을 실행합니다.
        else {
                // DC모터가 연결된 핀으로, duty-cycle을 0%로 설정하여 DC모터를 멈추게 합
니다.
                analogWrite(motor, 0);
        }
}
```

소스 출처-코코아팹 https://kocoafab.cc/tutorial/view/353

<실험-2> 서보모터 제어

다음으로는 서보모터를 원하는 각도로 on off 스위치로 제어하는 실험을 해보았다. 이 실험으로 인해 뒤에 최종적으로 우리가 원하는 서보모터의 움직임을 얻는데 큰 도움이 되었다. 원하는 각도만큼 값을 입력 하고 한번 스위치를 눌렀을 때 원하는 각도만큼 서보모터의 각도가 유지되고 스위치에서 손을 떼면 서보모터의 값이 '0' 으로 초기화 되어 원래 상태로 돌아오게 만들었다.



<소스코드-2>

```
#include <Servo.h>

//서보 모터를 사용하기 위한 Servo 객체를 생성
Servo myServo;
//버튼을 5번핀에 연결
int button = 5;

void setup() {
        pinMode(button,INPUT);
        myServo.attach(3);
}

void loop() {
        //만약 버튼이 눌러졌다면
        if (digitalRead(button)) {
            myServo.write(180); //180도로 이동
        }
```

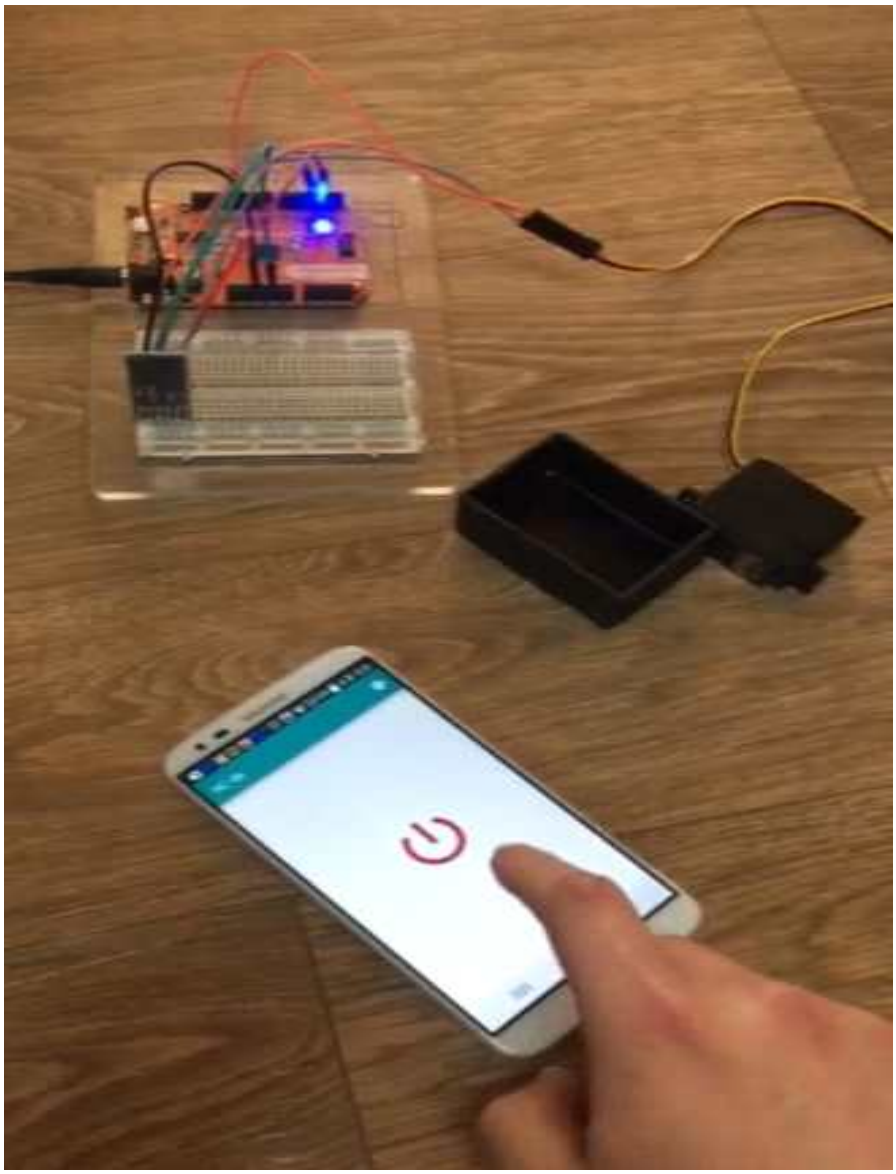
```
        //만약 버튼이 눌러지지 않았다면
        else {
                //0도로 이동
            myServo.write(180);
          }
}
```

소스 출처-코코아팹https://kocoafab.cc/tutorial/view/587

<실험-3> 블루투스 통신을 이용한 서보모터 제어

본격적으로 블루투스 원거리 통신을 하기 위해 스마트폰과 블루투스 HC-06를 사용하여 서보모터를 제어해 보았다. 블루투스 통신은 안드로이드 기반으로 한 스마트폰만 아두이노와 호환되기 때문에 안드로이드 스마트폰을 사용했고 코코아팹에서 제공하는 앱을 이용해 서보모터를 제어 했다. 블루투스는 슬레이브와 마스터가 있는데 쉽게 설명하자면 슬레이브는 출력해주는 모드라고 보면 되고 마스터는 입력해주는 모드라고 보면 된다. 여기서 아두이노의 블루투스는 슬레이브, 스마트폰은 마스터라 할 수 있다.

HC-06은 주로 슬레이브로 제작되어 나오지만 AT창에서 마스터 모드로 변경할 수 있다. 비밀번호와 블루투스 이름도 따로 설정 가능하다.



<소스코드-3>

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(2, 3);
void setup() {


  Serial.begin(9600);
  BTSerial.begin(9600);
  Serial.println("ATcommand");  //ATcommand 시작
}

void loop() {
  if (BTSerial.available())                    // 시리얼에서 읽은 것을 bt시리얼에서 읽고 bt시리얼에서 읽은 것은 시리얼에서 읽도록 한다.(시리얼은 아두이노와 컴퓨터간 통신,bt시리얼은 블루투스)
    Serial.write(BTSerial.read());
  if (Serial.available())
    BTSerial.write(Serial.read());
}
```

소스코드 코딩 후 AT창에서 AT 명령어를 입력해 이름 및 비밀번호 설정

AT+NAME 이름 설정 ex) 'AT+NAMEsejin=OKsejin' 이라고 뜸

AT+PIN 비밀번호 설정 ex) 'AT+PIN1234=OKsetpin' 이라고 뜸

<블루투스 서보모터 제어 소스코드>

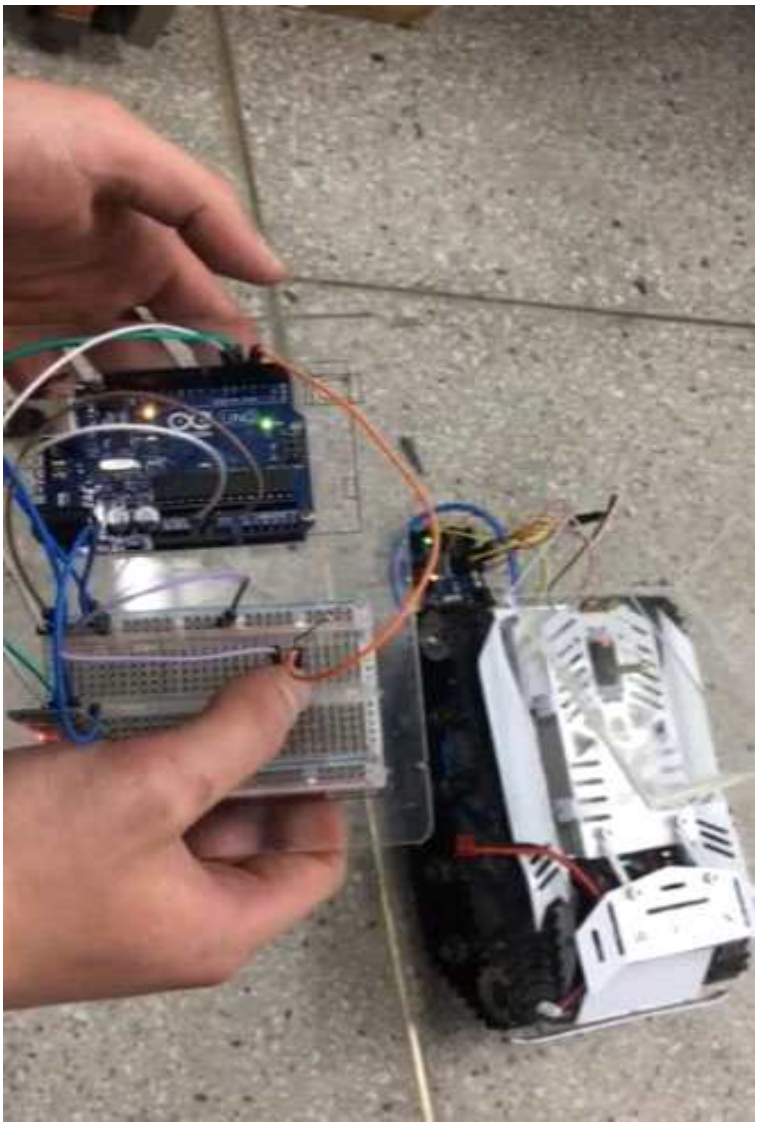
```
#include <SoftwareSerial.h> //시리얼 통신 라이브러리 호출
#include "Servo.h" //서보 라이브러리
Servo myservo; //서보객체
int blueTx=2;   //Tx (블투 보내는핀 설정)
int blueRx=3;   //Rx (블투 받는핀 설정)
SoftwareSerial mySerial(blueTx, blueRx);  //시리얼 통신을 위한 객체선언
String myString=""; //받는 문자열
void setup() {
  myservo.attach(12);   //서보 시그널 핀설정
```

```
  myservo.write(0);        //서보 초기각도 0도 설정
  mySerial.begin(9600); //블루투스 시리얼 개방
}
void loop() {
  while(mySerial.available())  //mySerial 값이 있으면
  {
    char myChar = (char)mySerial.read();  //mySerial int형식의 값을 char형식으로 변환
    myString+=myChar;    //수신되는 문자열을 myString에 모두 붙임 (1바이트씩 전송되는 것을 모두 붙임)
    delay(5);                //수신 문자열 끊김 방지
  }
  if(!myString.equals(“”))  //myString 값이 있다면
  {
    Serial.println(“input value: “+myString); //시리얼모니터에 myString값 출력
      if(myString==“on“)  //myString 값이 ’on’ 이라면
      {
        myservo.write(60);       //각도 60도로 움직임
      } else {
        myservo.write(0);    //각도 0도로 움직임
      }
    myString=“”;  //myString 변수값 초기화
  }
```

출처: http://deneb21.tistory.com/347 [Do It Yourself!]

<실험-4> 블루투스 양방향 통신

앞전의 실험에서 스마트폰이 마스터였다면 마스터부분을 따로 설정해 리모콘을 만들 수 있는지 실험해 보았다. 블루투스 설정을 마스터 모드로 변환해 주고 마스터 블루투스와 슬레이브 블루투스의 통신 속도를 동일하게 설정해 준다. 슬레이브는 서보엑추에이터를 사용해보았고 소스코드는 앞전 실험과 동일하게 코딩했다. 이 실험을 통해서 아두이노간 블루투스 통신의 이해도가 가장 높았고 서보엑추에이터 4개 동시제어하기 전 단계까지 왔다고 할 수 있다.



<자작 리모콘 마스터 소스코드>

```
#include <SoftwareSerial.h> //시리얼 통신 라이브러리 호출
#include "Servo.h" //서보 라이브러리
Servo myservo; //서보객체
int blueTx=2;    //Tx (블투 보내는핀 설정)
```

```
int blueRx=3;    //Rx (블투 받는핀 설정)
SoftwareSerial mySerial(blueTx, blueRx);  //시리얼 통신을 위한 객체선언
String myString=“”; //받는 문자열
void setup() {
  myservo.attach(12);    //서보 시그널 핀설정
  myservo.write(0);      //서보 초기각도 0도 설정
  mySerial.begin(9600); //블루투스 시리얼 개방
}
void loop() {
  while(mySerial.available())  //mySerial 값이 있으면
  {
    char myChar = (char)mySerial.read();  //mySerial int형식의 값을 char형식으로 변환
    myString+=myChar;     //수신되는 문자열을 myString에 모두 붙임 (1바이트씩 전송되는 것을 모두 붙임)
    delay(5);             //수신 문자열 끊김 방지
  }
  if(!myString.equals(“”))  //myString 값이 있다면
  {
    Serial.println(“input value: “+myString); //시리얼모니터에 myString값 출력
      if(myString==“a“)  //myString 값이 ’on’ 이라면
      {
        myservo.write(60);       //각도 60도로 움직임
      } else {
        myservo.write(0);    //각도 0도로 움직임
      }
    myString=“”;  //myString 변수값 초기화
  }
}
```

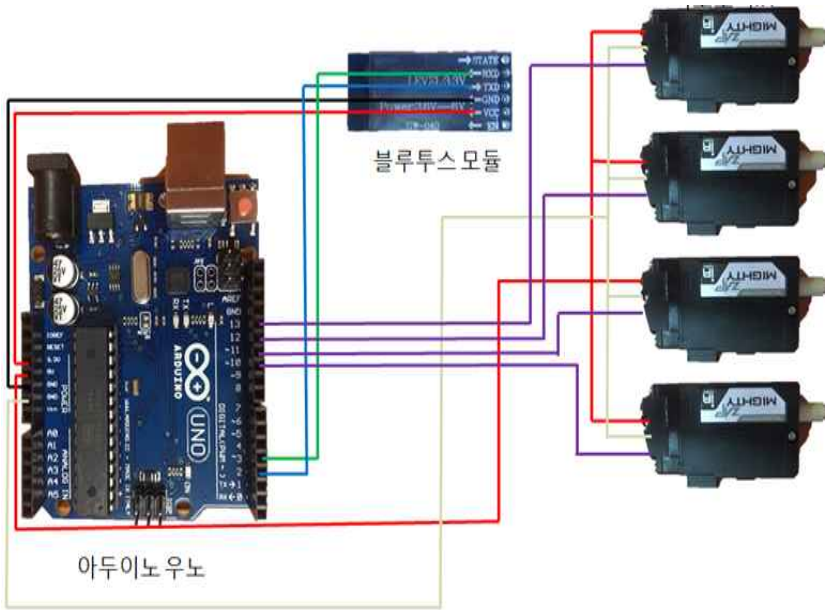
버튼으로 제어 하려 했지만 체터링 현상 때문에 서보엑추에이터가 제대로 작동하지 않았다. 체터링 현상이란 번튼을 누를 때 사람이 보는 관점에서는 한번 눌렀지만 기계적 특성으로는 한번 눌렀을 때 미세한 진동이 일어나 신호가 튀는 현상이 발생하게 된다.
한번 누르면 on 한번더 누르면 off라고 입력을 했다면 한번 눌렀을 때 on이라는 값이 3~4회 연속으로 값이 입력되어 버리는 것이다. 이를 해결하기 위해서는 하드웨어적인 방법과 소프트웨어적인 방법이 있는데 하드웨어적인 방법은 회로도를 다 바꾸어야하기 때문에 비효율적이라 판단되어 제외하였고 소프트웨어적인 방법은 소스코드를 수정하게 되면 체터링이 완화되게 된다. 체터링 방지 코드는 다음과 같다.

```
delay_ms(100);  //스위치 누를 때의 채터링 방지용 딜레이
while(!PIND.?);  //스위치가 ?인 동안 대기
delay_ms(100);  //스위치가 손을 뗄 때의 채터링 방지용 딜레이
```

적용해본 결과 채터링 현상이 완화되긴 했지만 여전히 떨리는 현상이 발생해 결국 버튼에서 조이스틱으로 변경하기로 했다. 조이스틱은 한 방향으로 밀고 있으면 값이 유지되어 원하는 만큼 서보엑추에이터의 각도조절이 가능하고 여러 방향 동시에 움직일 수 있다는 것이 장점이다.

이제 실험과 연습과정은 충분 하다고 생각되었고 조이스틱을 마스터로 하고 서보엑추에이터를 4개 동시에 제어할 수 있는 설계를 해보았다.

회로도는 다음과 같다.



<그림 2-15> 블루투스와 서보엑츄에이터의 구성도

블루투스는 RX TX 교차로 연결 해주어야 하며 블루투스 전원은 3.5V 에 연결한다. 사용하는 배터리는 12V이지만 아두이노에 외부전력 가용 범위는 5V~12V이고 별도의 전압 변환기능이 있어서 3.5V와 5V를 나눠 블루투스와 서보 엑추에이터에 각각 공급이 가능하다. 서보엑추에이터의 정격 전압은 7.4V이라 작동하는 속도가 약간 느려지긴 했다.

<서보모터 4개 동시제어 소스코드> 슬레이브

```
#include <SoftwareSerial.h> //시리얼 통신 라이브러리 호출
#include "Servo.h" //서보 라이브러리

Servo myservo1; //서보객체
Servo myservo2; //서보객체
Servo myservo3; //서보객체
Servo myservo4; //서보객체

int blueTx=2;    //Tx (블투 보내는핀 설정)
int blueRx=3;    //Rx (블투 받는핀 설정)
SoftwareSerial mySerial(blueTx, blueRx);  //시리얼 통신을 위한 객체선언
String myString=""; //받는 문자열

void setup() {
  myservo1.attach(10);    //서보 시그널 핀설정
  myservo2.attach(11);
  myservo3.attach(12);    //서보 시그널 핀설정
  myservo4.attach(13);
  myservo1.write(0);      //서보 초기각도 0도 설정
  myservo2.write(0);      //서보 초기각도 0도 설정
  myservo3.write(0);      //서보 초기각도 0도 설정
  myservo4.write(0);      //서보 초기각도 0도 설정
  mySerial.begin(9600); //블루투스 시리얼 개방
  Serial.begin(9600);
}

int rightJoystickX = 0;
int rightJoystickY = 0;
int leftJoystickX = 0;
int leftJoystickY = 0;

void loop() {
  if (mySerial.available()) {
    int value = mySerial.read();

    switch(value) {
      case 1:
        if(rightJoystickX<180) {
          rightJoystickX+=1;
        }
```

```
        break;
      case 2:
        if(rightJoystickX>0) {
          rightJoystickX-=1;
        }
        break;
      case 3:
        if(rightJoystickY<180) {
          rightJoystickY+=1;
        }
        break;
      case 4:
        if(rightJoystickY>0) {
          rightJoystickY-=1;
        }
        break;
      case 5:
        if(leftJoystickX<180) {
          leftJoystickX+=1;
        }
        break;
      case 6:
        if(leftJoystickX>0) {
          leftJoystickX-=1;
        }
        break;
      case 7:
        if(leftJoystickY<180) {
          leftJoystickY+=1;
        }
        break;
      case 8:
        if(leftJoystickY>0) {
          leftJoystickY-=1;
        }
        break;
    }

}
myservo1.write(rightJoystickX);
  myservo2.write(rightJoystickY);
```

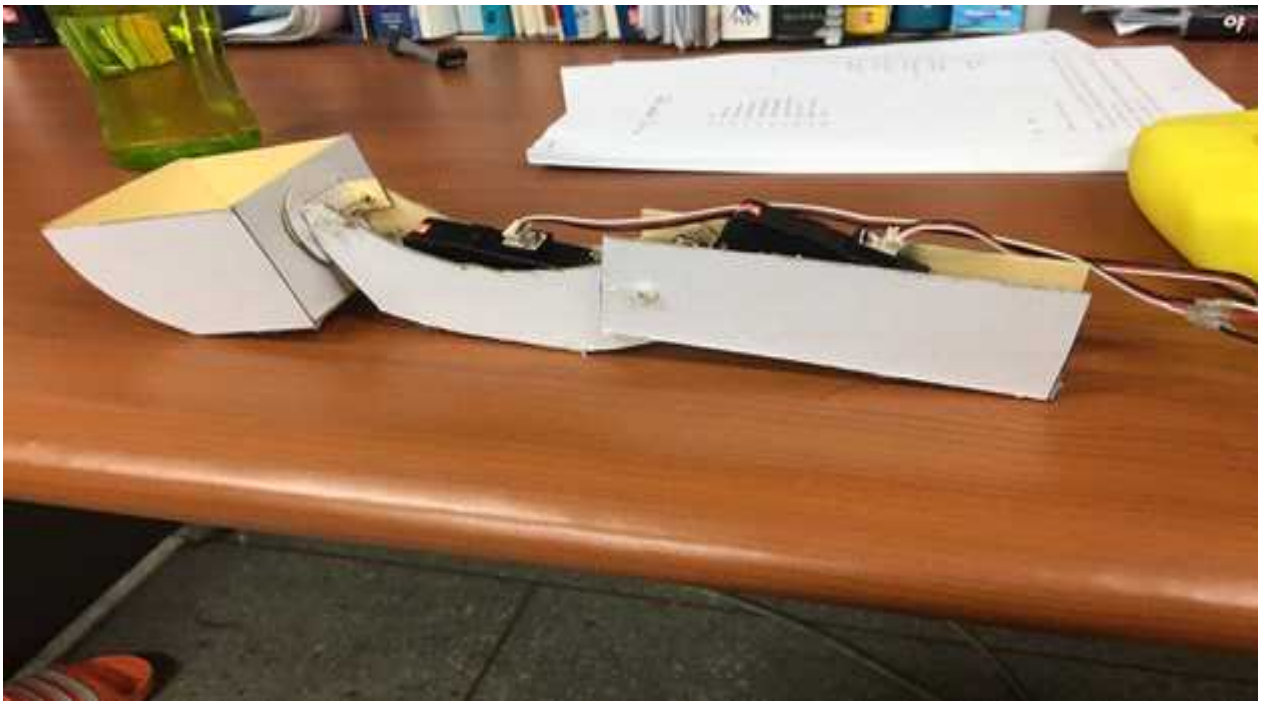
```
    myservo3.write(leftJoystickX);
    myservo4.write(leftJoystickY);
}
```

<조이스틱 소스코드> 마스터

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(2, 4);

const int L_JOYSTICK_X = A0;
const int L_JOYSTICK_Y = A1;
const int R_JOYSTICK_X = A2;
const int R_JOYSTICK_Y = A3;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  BTSerial.begin(9600);
  pinMode(7, OUTPUT);      //HC-05
  digitalWrite(7,LOW);
  pinMode(L_JOYSTICK_X, INPUT);
  pinMode(L_JOYSTICK_Y, INPUT);
  pinMode(R_JOYSTICK_X, INPUT);
  pinMode(R_JOYSTICK_Y, INPUT);
}

void loop() {
  int rightJoystickX = analogRead(R_JOYSTICK_Y);
  int rightJoystickY = analogRead(R_JOYSTICK_X);
  int leftJoystickX = analogRead(L_JOYSTICK_Y);
  int leftJoystickY = analogRead(L_JOYSTICK_X);

  Serial.print(rightJoystickX);
  Serial.print(“, “);
  Serial.print(rightJoystickY);
  Serial.print(“, “);
  Serial.print(leftJoystickX);
  Serial.print(“, “);
  Serial.println(leftJoystickY);
```

```
  if(rightJoystickX < 300) {
    BTSerial.write(1);
  }else if(rightJoystickX > (1024-300)) {
    BTSerial.write(2);
  }

  if(rightJoystickY < 300) {
    BTSerial.write(3);
  }else if(rightJoystickY > (1024-300)) {
    BTSerial.write(4);
  }

  if(leftJoystickX < 300) {
    BTSerial.write(5);
  }else if(leftJoystickX > (1024-300)) {
    BTSerial.write(6);
  }

  if(leftJoystickY < 300) {
    BTSerial.write(7);
  }else if(leftJoystickY > (1024-300)) {
    BTSerial.write(8);
  }

  delay(20);
}
```

마지막으로 이때까지의 설계로 붐과 암, 버켓이 잘 작동 될 수 있는지 간단한 테스트를 해보았다. 하드보드지로 붐과 암, 버켓을 만들고 이쑤시개로 핀을 만들어 기능부를 제작해 구동 시켜보았다.



<그림 2-17> 회로 및 소스코드 확인을 위한 테스트 장치

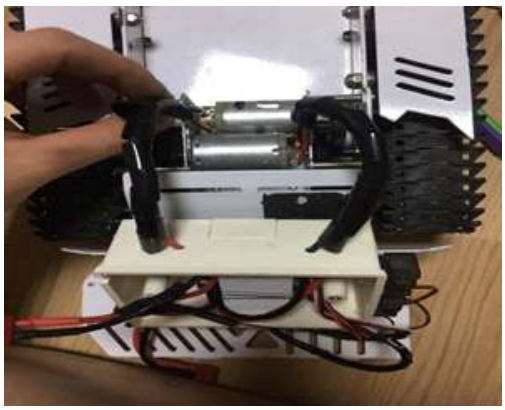
테스트 결과는 정상적으로 잘 작동하나 암과 붐, 버켓을 연결하는 핀의 위치가 어디에 위치하느냐에 따라 원하는 각도만큼 움직일 수 있다. 절점위치를 계산하는 방법을 찾아 정확한 위치에 구멍을 내고 핀을 박아야 하지만 방법을 찾지 못해 수차례 여기저기 구멍을 뚫어 본 결과 원하는 절점 위치를 확인할 수 있었다.

제 3 장 제 작

제 1 절 공정도

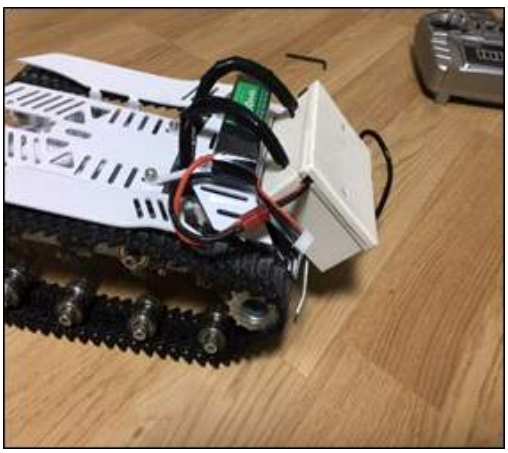
1) 구동부

<그림 3-1>에서 외부수신기를 분해하고 내장하는 과정을 나타내고 있다. 구동부 외부에 외부수신기가 부착이 되어 있었고 외부 수신기 부분이 비젼부와 기능부2(Bucket)를 설치 할 부분으로 선정을 하여 외부수신기를 분해를 하였고, 외부수신기를 분해한 뒤 구동부 안쪽에 내장 시켜 외부공간을 확보했다.



외부수신기 분해

외부수신기 모듈



수신기 내장 전

수신기 내장 후

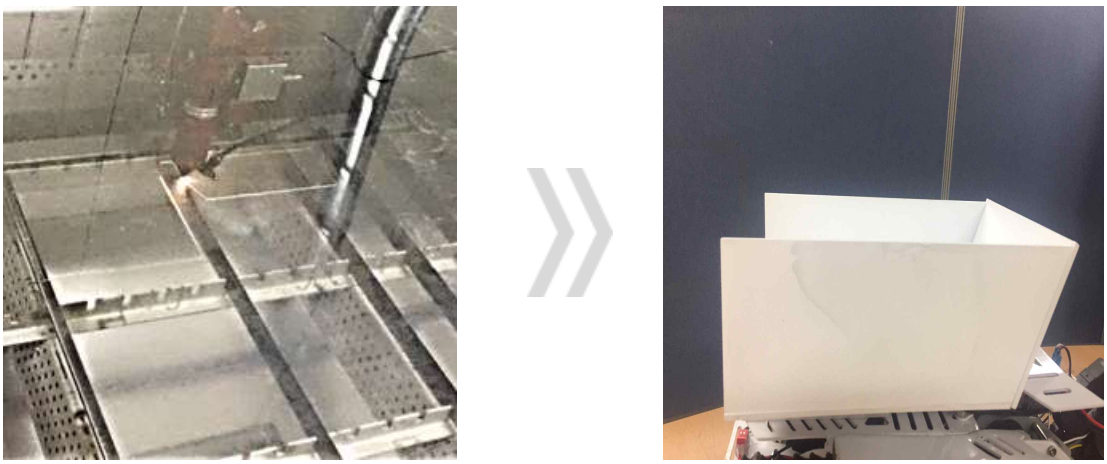
<그림 3-1> 구동부 구성 및 개선

2) 비젼부

비젼부는 기존에 구동부의 외부수신기가 부착이 되어 있던 곳을 떼어내고 그 위치에 비젼부를 부착하였다.

3) 기능부1(Dump)

덤프는 아크릴 판을 이용하여 제작하였다. <그림 3-2>에 나타낸 바와 같이 아크릴 판을 아크릴 레이저 커팅기로 제단을 한 뒤 접착을 시켜 덤프 제작을 진행하였다. 그리고 완성된 덤프에 도색을 진행하였다. 또한, 구동부에 덤프를 부착시키기 위해 <그림 3-3>에 나타낸 바와 같이 구동부에 경첩을 우선 3D프린터로 제작 후 부착한 후에 3D프린터로 경첩과 덤프사이를 고정시킬 핀을 따로 제작하였다.



<그림 3-2> 기능부의 구성 및 제작

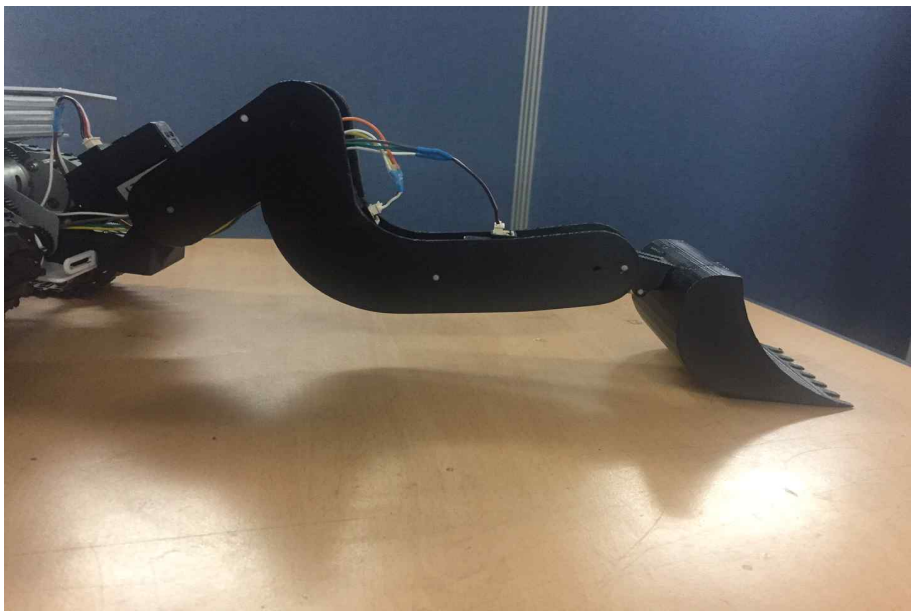
<그림 3-3> 경첩부의 구현을 위한 3D 프린팅 적용

4) 기능부2(Bucket)

두번째 기능부의 구성품인 암, 붐, 버킷의 3 가지는 3D프린터로 출력했다. 또한, 제작된 암, 붐, 버킷을 앞서 말했던 핀을 사용하여 결합을 하고 도색 후 구동부와 결합하였다.

암(Arm) 붐(Boom) 버켓(Bucket)

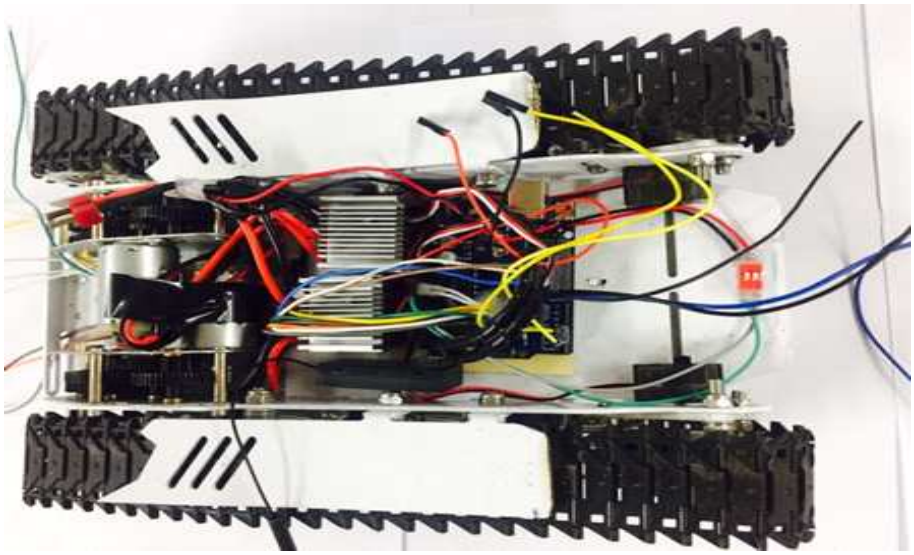
<그림 3-4> 경첩부의 구현을 위한 3D 프린팅 적용



<그림 3-5> 완성된 버킷 기능부

5) 배터리 결합

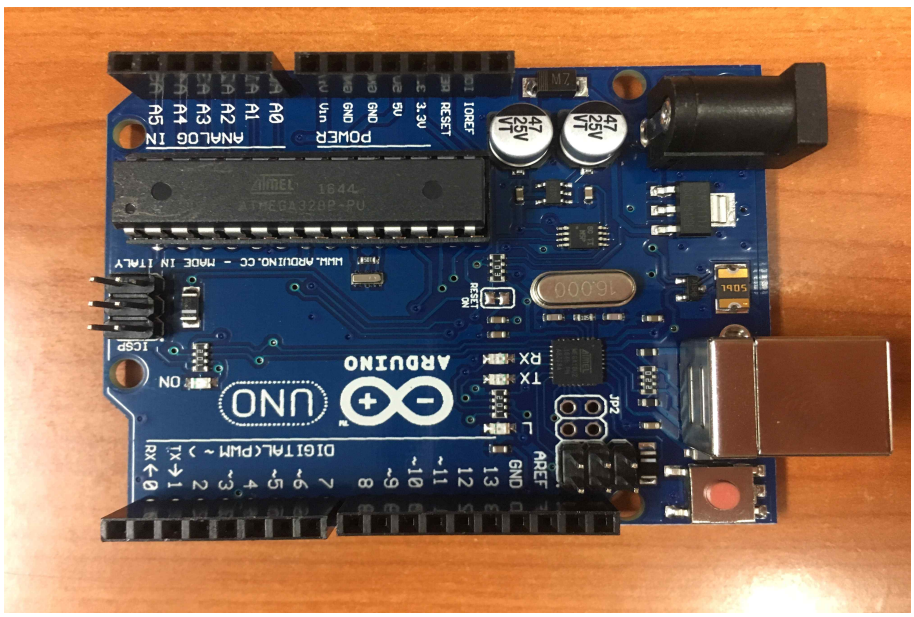
배터리는 구동부와 굴삭 부분에서 총 12V의 전력이 사용이 되어 아두이노에 따로 추가적인 배터리를 다는 대신에 기존 구동부에 장착이 되어있는 12V 배터리로 아두이노와 전력을 공유할 수 있게 외부전력을 주었다.



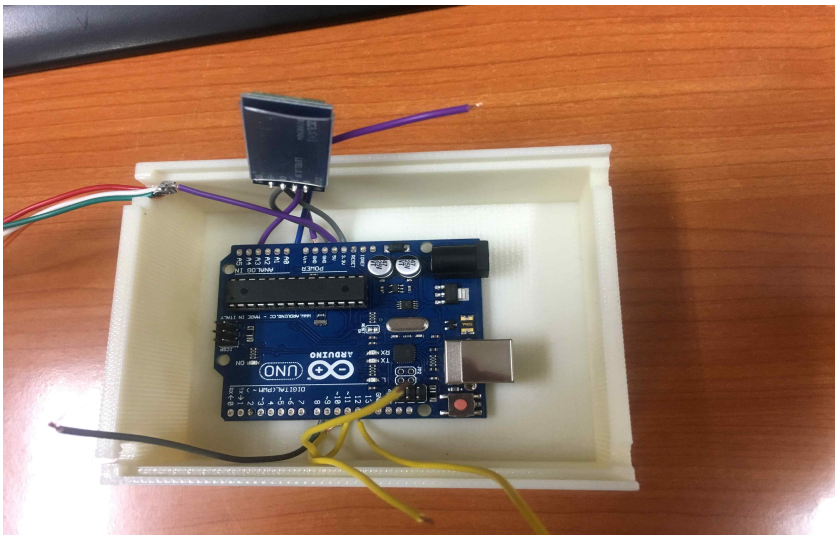
<그림 3-6> 구동부 내면 및 배터리 장착

6) 제어기 구성

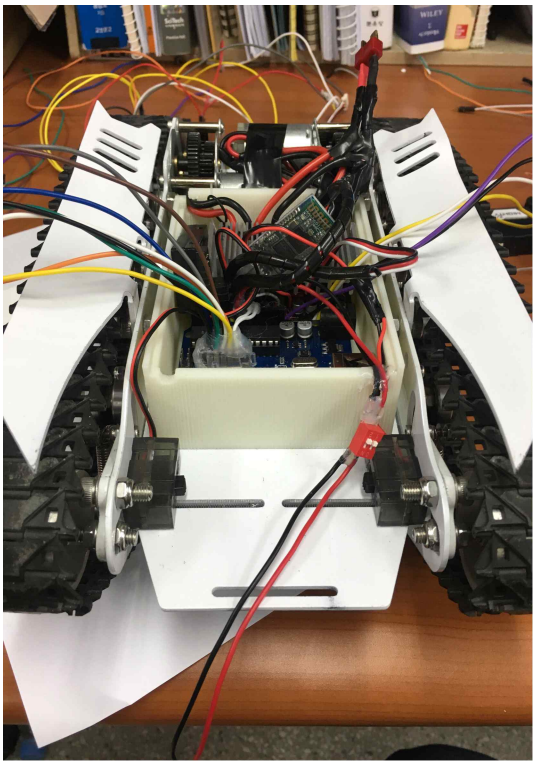
우선 아두이노 우노를 가지고 블루투스 모듈을 장착하고 납땜을 하였다. 그리고 서보모터와 연결 할 수 있는 전선도 납땜을 하였다.



<그림 3-7> 사용된 아두이노 우노 기판

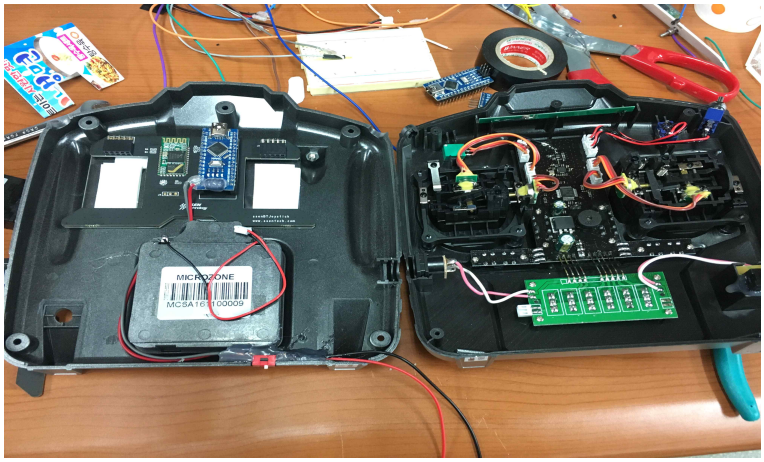


<그림 3-8> 아두이노 우노 기판의 조립을 위한 케이스 제작(3D 프린터)



<그림 3-9> 외부 수신기와 납땜을 완료한 아두이노 우노의 구동부 내장

구동부를 움직이게 하는 기존의 Remote Control을 분해하여 기능부1(Dump), 기능부2(Bucket)을 조종하는 Remote Control을 만들어 구동부와 기능부의 Remote Control을 통합하였데, 그 구성품은 <그림 3-10>에 나타낸 바와 같다.



<그림 3-10> 리모트 컨트롤러 분해 및 개조

제4장 시험 및 평가

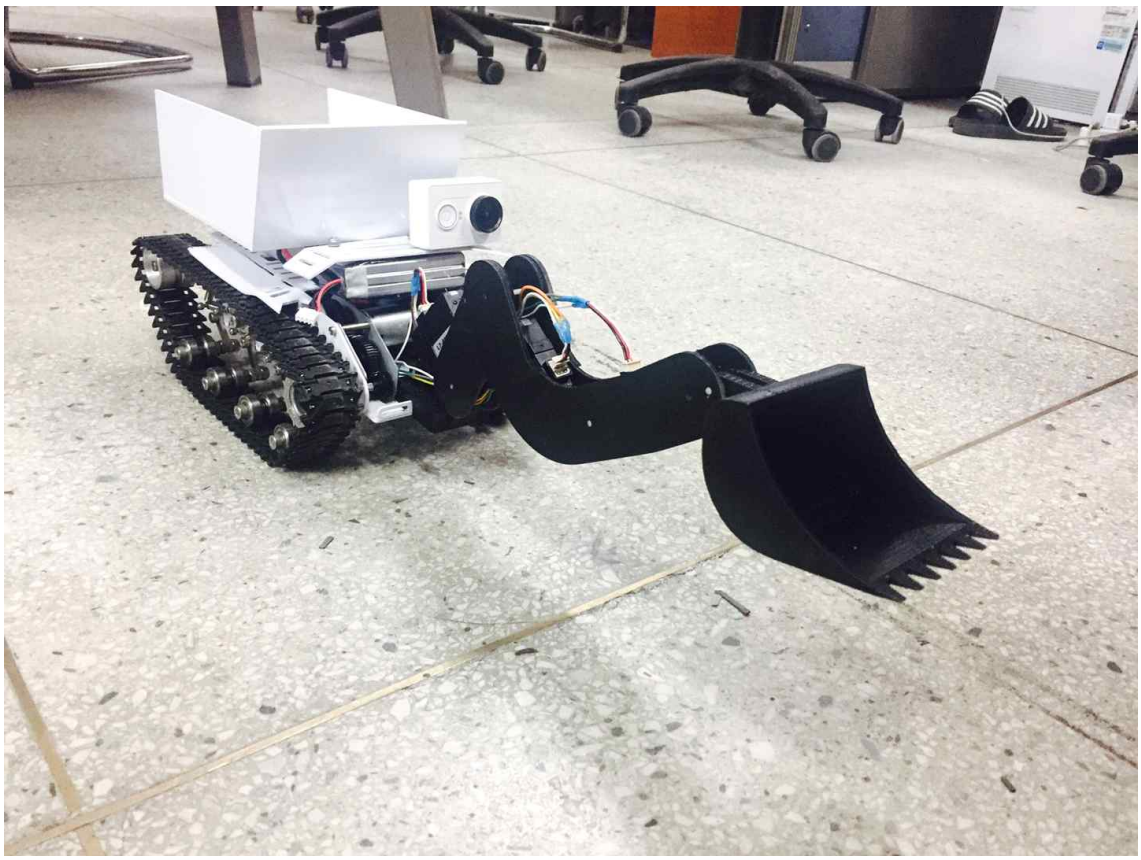
제 1절 운용 및 시험 요구조건

1) 적재, 하역, 운반이 동시에 작동 하는가?

정확한 계산 없이 절점을 잡았기 때문에 약간의 오차는 있지만 원하는 목표를 퍼 올려서 적재까지 가능했다. 하지만 서보엑추에이터의 떨림 현상과 블루투스의 꺼짐이 발생되어 미흡하다고 볼 수 있다. 서보엑추에이터의 떨림 현상은 조이스틱을 한 방향으로 밀어줄 때 정확하게 밀어주어야 하는데 약간이라도 다른 방향으로 휘어지게 되면 다른 서보엑추에이터에 조금씩 값이 입력되어 버려 동시 다발 적으로 서보엑추에이터가 움찔하게 된다. 이때 순간적으로 전류를 많이 소비하게 되어 아두이노에 무리가 가게 되어 블루투스의 꺼짐 현상도 같이 발생하게 되는 것 같다.

2) 보이지 않는 곳에서도 간접적으로도 사용이 가능한가?

사용가능하다. 다만 액션캠이 저가 중국산 제품이라 실시간 영상이 끊어지게 보여 어려움이 있긴 하지만 근거리에서는 무리 없이 작동 가능하다. 실시간 영상만으로 작업을 하려면 어느정도의 조작 연습이 필요하다. 또 미처 생각 하지 못한 부분이 어두운 굴 속으로 들어가 작업하는 중장비 인데 조명을 생각하지 못했다는 것이 아쉽다.



<그림 4-1> 최종 완성된 리모트 컨트롤형 LHD 장비

제 5 장 결론

제 1 절 문제점 분석 및 처리결과

1) LHD장비를 개발하면서 문제점이 블루투스가 반복적으로 꺼지는 현상이 발생 하였다. 이 현상은 아두이노에서 서보모터로 전력을 공급하기 때문에 순간적으로 전압이 낮아져서 생기는 현상이라 생각이 되어 외부전력을 아두이노와 서보모터에 따로 전력을 공급하면 해결될 것이다.

2) 서보모터 떨림 현상은 채터링 현상과 비슷한데 조이스틱에서는 가령 x축으로 민다고 생각했을 때 실제로는 정확한 x방향이아니라 약간 대각선으로 밀게 되어 x값만 이외에 다른 방향값을 읽게 되어 다른 서보엑추에이터 들이 움찔움찔 하게 되는 것이다. 이 부분은 조이스틱이 한 방향만을 인식 할 수 있게 명령어를 재구성을 하면 해결될 것이다.

제 2 절 향후과제

LHD를 만들고 난 후 가장 큰 문제점이 블루투스 꺼짐 현상 이였다. 처음에는 원인도 모르고 블루투스만 계속 교체했었다. 거듭 반복한 결과 원인은 외부전압 주는 방식과 조이스틱 소스코드, 배터리에 문제가 있었다. 외부전압을 주는 방식은 아두이노와 서보 엑추에이터에 따로 전력을 공급해 주면 훨씬 안정적인 작동을 할 수 있다. 서보 엑추에이터도 12V 용으로 구매해 배터리에서 바로 연결 했더라면 무리가 덜 갔을 것이다.

하지만 서보엑추에이터의 가격이 만만치 않기 때문에 회로도와 부품을 중간에 교체할 수는 없었다. 가장 중요한 점을 간과한 것이 있는데 어두운 동굴 안에서 작업을 할 중장비에 조명이 없다는 것이 아쉬웠다. 처음 생각했던 것 보다 작품이 잘 나왔고 작동도 잘해 주었지만 제작과 작동하는 것에만 급급해 디테일한 부분을 놓친 점이 아쉬웠다.

보고서 후기

전자제어에 관한 지식이 전혀 없는 상태였지만 아두이노와 온라인상에 공유 되어있는 오픈소스를 하나하나 찾아감을 통해 제어에 흥미를 느낄 수 있었고, 기능부를 작동을 시킬 때 서보모터를 좀 더 부드럽게 제어하지 못한 부분이 아쉽게 만 느껴졌습니다. 기존에 있는 중장비 형상의 틀에서도 벗어나지 못하고 더 새로운 형상이 나오지 못한 점 또한 많이 아쉽습니다.

무작정 인터넷을 보면서 제어에 필요한 오픈소스를 찾기만 할 것이 아니라 제어를 다뤄 봤던 우리 학과 선, 후배들을 한번이라도 더 찾아가서 물어보고 배우면서 이번 설계 프로젝트를 진행 했으면 보다 문제점도 개선이 되었을 것이고 지금보다 훨씬 더 좋은 결과가 나왔을 것인데 그렇게 하지 못 했다는 게 가장 아쉬웠던 것 같습니다.

구동부와 비젼부 까지도 직접 제작해 보고 싶었지만 우선적으로 제어를 공부를 하고 저희 설계프로젝트 스티브 잡스 조원 전체가 겨울 방학 때 현장실습을 다녀옴으로써 방학에도 준비를 할 수 있었던 2달간의 공백의 틈이 다른 팀원들 보다 연구를 못 하였고 저희 조 내부에서도 작년 9월 달에 첫 설계 프로젝트 수업을 수강하면서 남은 조원들끼리 각 조원들이 분담했던 업무를 취업계로 인하여 안타깝게 빠지게 된 전 조원들의 연구까지 인수인계를 받고 이해하면서 준비하는 그 짧았던 시간마저 이제는 아쉽게 느껴졌습니다.

이번 프로젝트는 여러 부분에서 아쉬운 점이 많이 남는 프로젝트가 아니었나 싶습니다. 나중에 다시 동기들끼리 한 주제를 정하여 그 주제를 연구하고 연구를 진행하는 이런 기회가 두 번 다시는 오지는 않겠지만 나중에 사회에 나가서 회사 내의 같은 부서 직장상사와 직장 후배들과 이런 비슷한 프로젝트를 준비를 할 때 지금 이 순간이 많이 생각이 날것이고, 그때는 후회가 없는 프로젝트를 진행 할 수 있을 것이란 생각이 듭니다. 마지막까지 힘들어도 끝까지 포기하지 않고 서로 다독여주면서 여기까지 온 우리 스티브 잡스 조원 모두 서로에게 고맙고, 주마다 저희가 준비하는 과정을 봐주고 같이 문제점을 찾아 고치는데 신경을 써주시고 도움을 주신 김홍석 교수님, 임학규 교수님께도 감사하는 말씀을 드리면서 1년 동안 준비한 설계프로젝트 최종 레포트를 마무리 하겠습니다.