

# ME400 Capstone Design 1 – Final Report

Group 9 20150222 김태연

지도교수: 오일권 교수님

## 목차

### **1. Timeline**

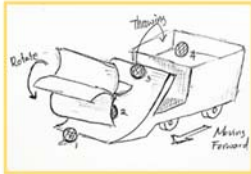

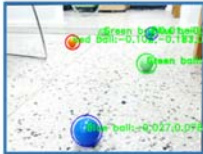

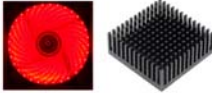
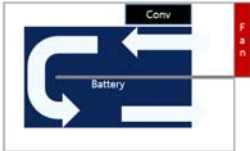




### **2. Our System IKOH**

- A. Heat transfer**
- B. Pick-up system**
- C. Motor control**
- D. Vibration control**
- E. Vision processing**
- F. ROS integration & Picking algorithm**

### **3. Result**

### **4. Progress report**

# 1. Timeline

●	3/13 우분투, ROS설치	
●	3/20 Fan Rotating 방법으로 결정	
●	4/2 Detection code 초안	
●	4/4 LabVIEW로 Motor control	
●	4/5 Hardware 초안 (Wheel+Structure)	
●	4/12 컨버터 구매	
●	4/13 ROS+LabVIEW 통합	
●	4/15 OpenCV+ROS 통합	
●	4/20 PMS 구현	
●	4/24 방열판, 플링겐 구매 Dynamixel 추가구매	
●	4/28 Picking Algorithm 초안	
●	5/2 Software 통합 (ROS+LabVIEW+OpenCV)	
●	5/3 Cooling duct 초안	
●	5/4 Hardware 통합 (Picking Blade+Wheel+Heat Duct)	
●	5/8 Picking code 초안	
●	5/18 Detection code 최종	
●	5/19 All system 통합(+NUC)	
●	5/22 최초 Mission 성공	
●	5/26 Cooling duct 최종	
●	5/27 All system 최종	
●	5/31 Picking code 최종	
●	6/1 3차발표 및 데모	

## 2. Our system IKOH

### A. Heat transfer

이번 미션은 공을 집어오는 미션을 수행한 후 Power managing system(이하 PMS)의 열 발생에 의한 system part의 최고 온도를 70도 미만으로 낮추는 것이었다. 본 조는 냉각 온도 목표를 room temperature로 잡아 온도 상승을 최소화하려고 시도하였다.

온도를 낮추는 방법에는 냉각제 사용, 공기 대류 등이 있었는데, 먼저 소모되는 냉각제를 사용하는 것은 반칙으로 판단하였고, 에너지 효율을 따져보았을 때 본 조는 대류에 의한 냉각에 thermal fin과 fan cooler, duct structure를 보조적인 역할로 사용하기로 하였다.



그림 1.a Cooling fan

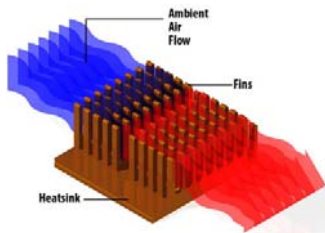
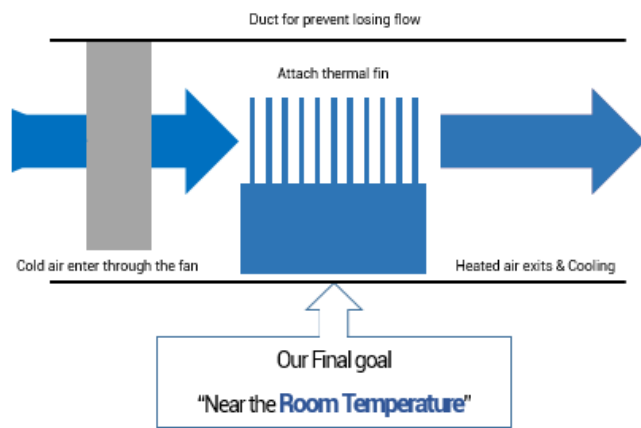


그림 1.b Thermal fin



1.c Duct system

#### 1) U-shaped duct

배터리에 대한 많은 주의사항이 전달되었고, 특히 과열에 의한 폭발을 막아야 한다는 경고에 의해 배터리의 온도를 낮추기 위한 duct를 설계하였다. 하나의 fan으로 효율적인 cooling을 하기 위해서 배터리의 각 면을 식힐 수 있도록 U자형으로 굽은 duct를 제작해 가운데 벽면에 배터리를 끼웠다. Converter역시 온도가 높았기 때문에 duct에 내장하여 냉각시켰다.

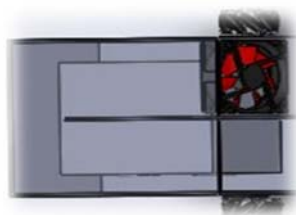


그림 2.a U-shaped duct

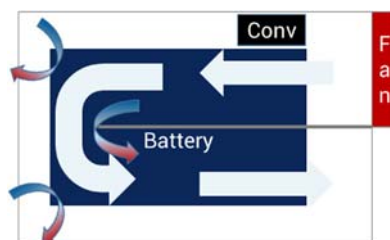


그림 2.b Problem of U-shaped duct

하지만 U-shaped duct에는 문제점이 있었다. 내부 공간에 flow가 꺾이는 모서리가 많다 보니, flow가 내부를 통과하며 energy loss가 발생하였다. 치명적인 오류는 배터리 자체의 온도가 높지 않았다는 것이다.

이러한 사실을 바탕으로 우리는 converter를 비롯한 여러 부품들을 cooling할 수 있는 compact한 duct를 새로 design하였다.

## 2) IKOH duct

먼저 converter를 효과적으로 냉각시키기 위해 대류뿐만 아니라 전도까지 사용하는 CPU cooler를 탑재하였다(그림 2.a). Thermal pad로 converter와 cooler의 접촉면의 전도를 최대화하였다. 12V와 15V converter에 각각 cooler를 달고, 이 cooler의 유량을 맞추기 위한 fan을 2개 더 사용하였다. 최종적으로 그림 2.b와 같은 duct를 설계할 수 있었다.

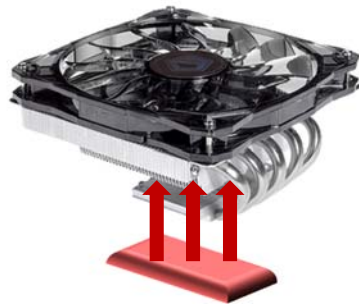


그림 3.a CPU cooler



그림 3.b 제작한 IKOH duct

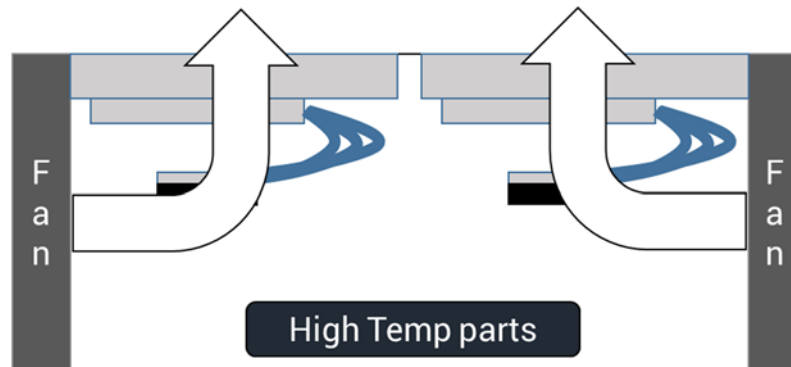


그림 3.c IKOH duct design

그 결과, 온도의 냉각은 효과적으로 이루어졌다. 그림 4의 그래프에서 IKOH duct를 설치하였을 때, 두 컨버터의 온도가 상온 근처에서 유지됨을 확인할 수 있다.

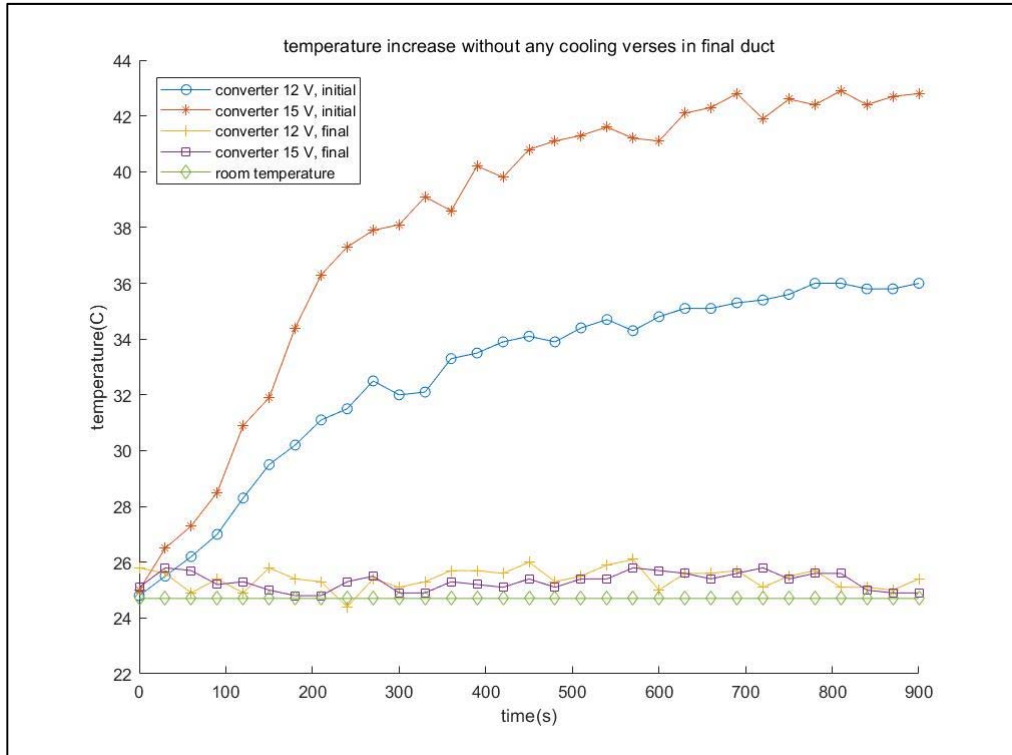


그림 4 초기 컨버터 온도 상승 및 최종 IKOH duct 설치 후 온도 상승 그래프

## B. Pick-up System

본 조에서는 pick-up system으로 rolling blade method를 사용하였다(그림 5). Rolling blade method에는 여러 가지 장점이 있다. 첫번째로, pick-up 도중에 전체 system이 정지할 필요가 없다. 직진하는 도중 blade를 회전시키면 땅에 있는 공이 blade에 의해 차체 내부로 들어온다. 두번째로, 제작과 제어가 쉽다. 여러 개의 관절이 있는 팔이 아니라 blade 하나만 회전시킴으로써 공을 줍는 mechanism이기 때문에, 오직 하나의 actuator만이 필요하다. 마지막으로, ball의 직경보다 넓은 범위를 커버하는 blade를 사용함으로써 중심을 맞추고 직진하는 도중 발생하는 오차가 있더라도 충분히 pick-up이 가능하다.

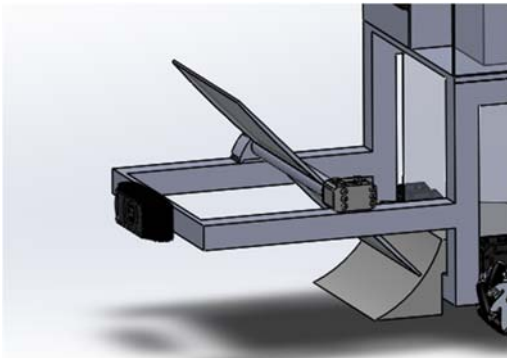


그림 5.a Solidworks로 blade 설계

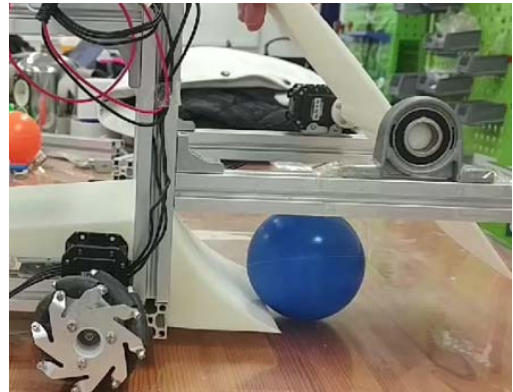


그림 5.b 제작한 blade

하지만 blade를 장착해 구동해본 결과 문제점이 있었다. Blade가 회전하며 공을 집을 때, 공이 땅과 blade 사이에 끼어버리는 것인데, 이는 blade가 공에 힘을 가하는 각도가 지면과 거의 수직하기 때문이었다. 따라서 이를 개선하기 위해 blade에 curvature를 추가하였다(그림 6).

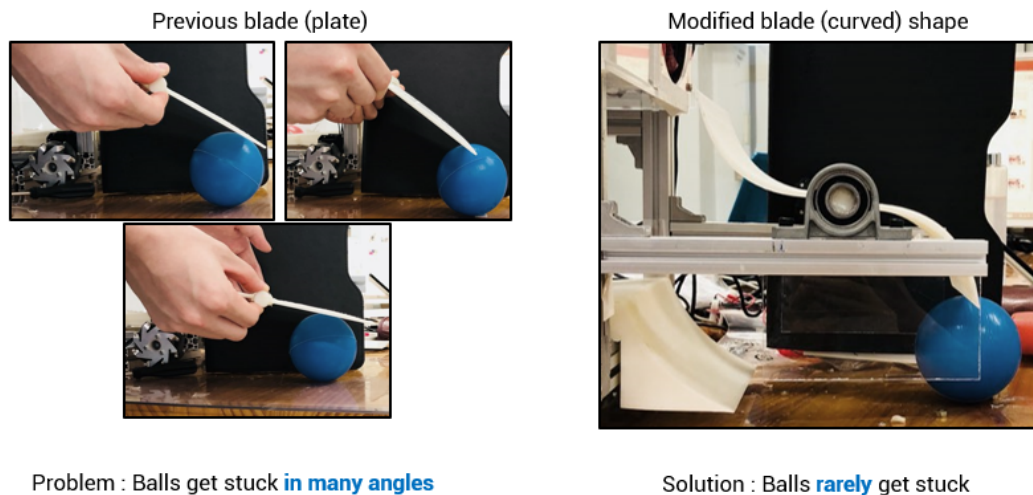


그림 6 Blade shape modification

Ball releasing의 경우 non actuator mechanism을 사용하였다. 바구니의 문턱 높이 특성을 고려하여, 특수한 형태의 문과 자석을 이용하여 별다른 actuator가 필요없이 공이 적재된 차체 뒤쪽의 문이 열려 바구니로 공이 들어가도록 하였다. 문은 위에서 아래로, 밖으로 열리는 구조이며, 문 아래에 튀어나온 돌기가 바구니의 문턱에 걸리도록 설계되었다. 차체가 바구니를 향해 후진하여 문턱이 걸리면 돌기가 힘을 받고, 힘이 토크로 변환되어 자석의 인력을 넘을 때 문이 열리는 구조이다.

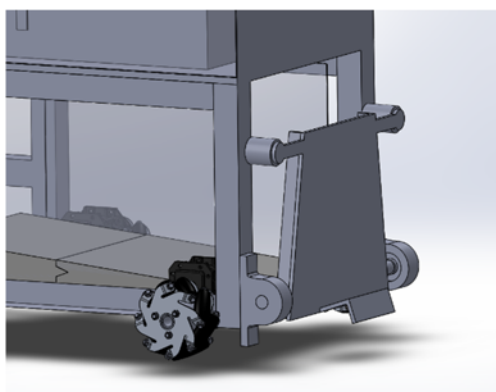


그림 7.a 닫힌 상태의 문

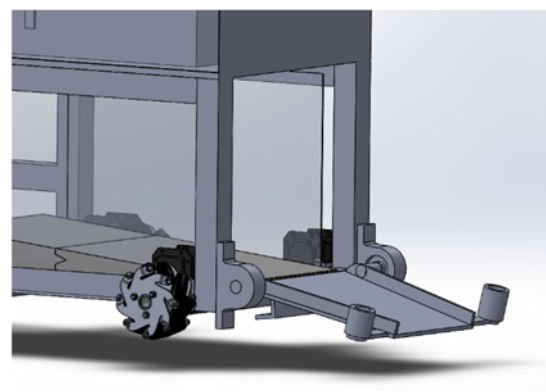


그림 7.b 열린 상태. 차체 내부 slope와 이어진 slope가 형성되어 공이 굴러내려온다.

## C. Motor control

차체가 원하는 경로를 따라 공을 pickup하기 위해선 바퀴의 rpm에 따른 차체 전체의 속도와 각속도를 구해야 한다. 따라서 메카넘휠의 'wheel kinematics'를 이용하였는데, wheel kinematics는 차체에 부착된 네 개의 메카넘휠의 rpm값을 차체의 속도값과 각속도 값으로 바꿔주는 상수행렬이다. 이를 이용해 system의 움직임 중 90도 회전이나 2m 직진과 같이 real world에서 절대값을 요구하는 명령을 정확히 줄 수 있게 되었다.

또한 모터의 속도(rpm)를 명령할 때, low pass filter를 이용하여 속도 변경에 damping을 주었다. 이러한 방법을 사용한 이유는, 급제동과 급가속을 하는 경우 차체가 앞뒤로 크게 진동하며 webcam의 세밀한 detection에 문제가 발생하기 때문이다. 예를 들어 파란공이 시야에 보이면 빠르게 직진하고, 공을 먹은 뒤 제자리에서 다음 공을 찾기 위해 회전하라는 명령을 준다고 하자. 이 때 회전을 시작하는 순간, 차체의 직선속도는 0으로 급감하고, 차체의 각속도는 회전속도만큼 급증한다.

이러한 계단형의 속도명령값을 방지하기 위한 것이 low pass filter인데, 그림 8.a와 같은 step input을 그림 8.b와 같이 매끄럽게 만들어준다. 차체의 속도 변경 명령의 최단주기를 파악하였더니 약 2초 가량이 나왔다. Damping이 너무 과할 경우 명령하고자 하는 거리 또는 각도에 한참 못미치는 값만큼 구동하므로, damping의 시상수가 전체 주기의 40분의 1이 되도록 하였다. 이를 통해 아래 식과 같이 transfer function  $H(s)$ 를 구하고, Z transform 을 통해 discrete system에서의  $y(n)$ 을  $y(n-1)$ 과  $u(n)$ 으로 나타낼 수 있었다. 이를 ROS code 명령문에 대입함으로써 damping을 구현하였다.

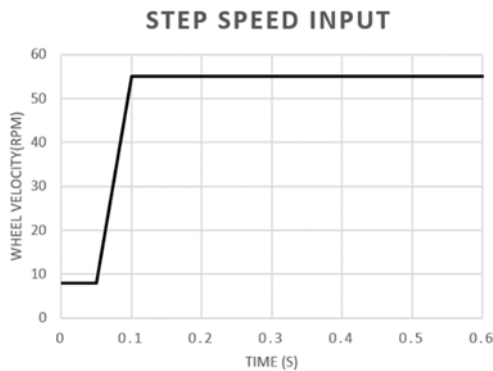


그림 8.a 계단형 속도 input

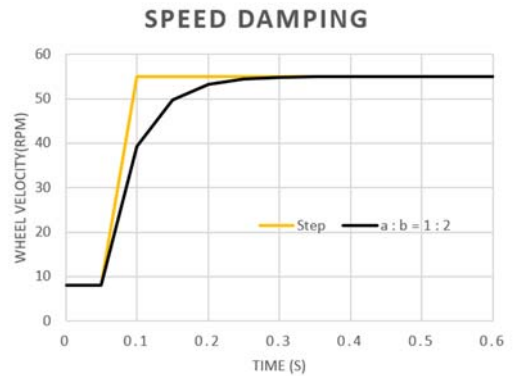


그림 8.b Damping을 적용한 속도 input

$$T = n\tau = 2(s), n = 40 \gg \tau = 0.05 \gg H = \frac{1}{\tau s + 1}$$

$$\frac{Y}{U} = \frac{0.6321}{z - 0.3679},$$

$$(1 - 0.3679z^{-1})Y = 0.6321z^{-1}U$$

$$y(n) - 0.3679y(n-1) = 0.6321u(n-1)$$

$$\therefore y(n) = 0.3679y(n-1) + 0.6321u(n) \simeq \frac{y(n-1) + 2u(n)}{3}$$



## D. Vibration control

Motor control에서 언급한 것과 같이, system의 소프트웨어에서 급가속과 급제동에 대한 vibration control이 실행되고 있다. 이번 장에서는 하드웨어에서 어떻게 웹캠의 vibration을 감소시켰는지 설명하고자 한다.

직진 시 바퀴의 각속도를 55rpm으로 놓는데, 이 때 일정한 주기를 가지는 소음이 관측되었다. 분석한 결과는 다음 식과 같다.

$$f_{ex} = 7.159Hz$$
$$\omega_{f,ex} = 2\pi f_{ex} = 2\pi \times 7.159Hz = 44.981rad/s$$

본 조에서는 이 소음의 원인이 하중에 의한 메카넘휠 회전축의 기울어짐에 의한 메카넘휠 돌기와 바닥 사이 접합면의 요철이라고 판단하였다. 바퀴의 돌기가 8개이고, 55rpm으로 돈다고 하였을 때 예상되는 system vibration의 진동수는 다음과 같다.

$$\omega_{f,the} = 2\pi f = 2\pi n\omega_{wheel}$$
$$= 2\pi \times 8 \times (55rpm/60sec) = 46.077rad/s$$

계산 결과, 주 진동의 원인이 메카넘휠의 요철임을 파악하였고, 이 진동수에 해당하는 transmissibility를 줄여야 한다고 판단하였다. 이 때 system이 공을 찾기 위해 회전하거나, 회피하는 동안에는 웹캠의 ball detection이 정밀할 필요가 없다. 하지만 직진할 시 진동에 의해 웹캠 화면 축이 범위 내를 진동할 경우 정확히 공을 향해 가질 못하고 엇나가게 된다. 따라서 본 조는 직진, 즉 46.077rad 주변의 진동을 집중하여 저감하기로 한다.

스펀지의 물성치를 특정 무게에 대한 변위값으로 측정하였고, 그 결과는 다음과 같다.

$$k_{sponge} = \frac{F}{\Delta x} = \frac{0.82kg \times 9.8m/s^2}{0.5mm} = 1.607 \times 10^4 N/m$$

그 다음으로 눌리는 깊이 h와 눌리는 넓이 d값을 측정하여 국부적인 부피에 대한 탄성계수를 다시 구하였다(그림 9.b,c)

$$k = k_{sponge} \times \frac{dh/2}{\pi D^2 h/4} = 247.421 N/m$$

이후 그림 9.a의 real system을 analyzable한 system으로 modeling하였다(그림 9.d). 이를 1차원 진동계로 변환하여, 1차원 진동계에서 basement excitation transmissibility를 구하려고 하였다. 식은 아래 그림 9.e와 같다.



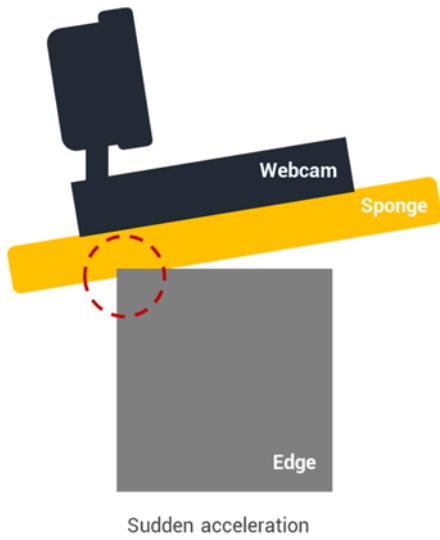


그림 9.a 급가속 시 웹캠 상태

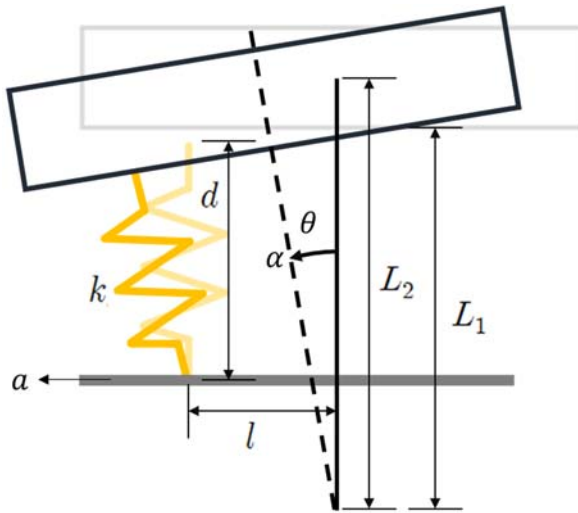


그림 9.d Modeling



그림 9.b 스펀지가 눌리는 모양

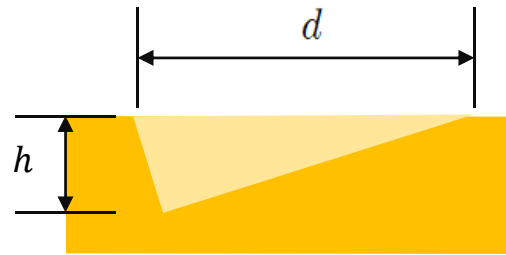


그림 9.c 분석을 위한 수치화

$$\alpha = L_2 a$$

$$x = -l$$

$$x' = -L_1 \sin \theta - l \cos \theta$$

$$y = d$$

$$y' = -L_1 (1 - \cos \theta) + d - l \sin \theta$$

$$\Delta x = x - x'$$

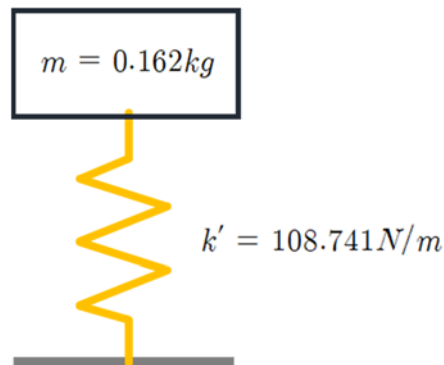
$$\Delta y = y - y'$$

$$m L_2^2 \ddot{\theta} = -k (\sqrt{\Delta x^2 + \Delta y^2}) l \cos \theta$$

$$\therefore m \ddot{\theta} = -k' \theta$$

$$k' = 108.741 \text{ N/m}$$

그림 9.e 1차원 진동계 변환 분석 수식



$$\omega_n = \sqrt{\frac{k'}{m}} = 25.908 \text{ rad/s}$$

그림 9.e 1차원 진동계로 변환한 결과

그 결과, 웹캠의 질량이 0.162kg일 때 스폰지의 일부가 웹캠과 차량 사이 edge에서 국부적으로 눌릴 때와 같은 탄성계수  $k=108.741\text{N/m}$ 이다. 이를 기반으로 damping ratio가 0에 근접할 때 transmissibility를 계산한 결과, 약 0.5가 도출되었다. 또한 본 조에서는 가속도센서를 이용하여 스폰지가 있을 때와 없을 때 직진에 따른 웹캠의 진동을 측정하였는데, 아래 그래프(그림 11.a,b)와 같이 반 가량 줄어듦을 확인할 수 있다.

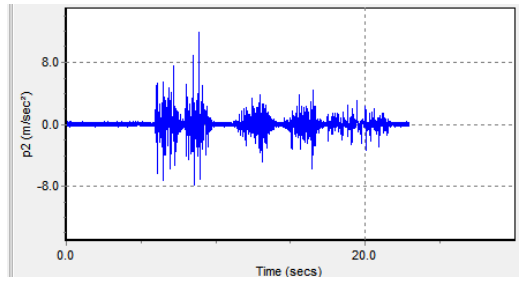


그림 11.a 스폰지가 없을 때

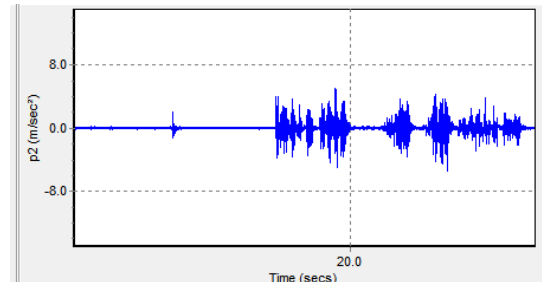


그림 11.b 스폰지가 있을 때

## E. Vision processing

공을 detect하는 데에 있어 본 조에서는 contour method를 사용하였다. 몇 가지 문제점이 있었고, 이를 모두 해결하여 system이 mission을 수행함에 있어 전혀 문제되지 않도록 하였다.

첫 번째 문제점은 천장에 설치된 광원이 매끈한 고무재질의 공 표면에 비칠 경우, 해당하는 부분이 하얗게 촬영되어 마치 구멍난 원으로 인식한다는 것이다. 이는 morphological process; 인식하는 pixel 자체를 키우는 보정을 통해 해결할 수 있었다.

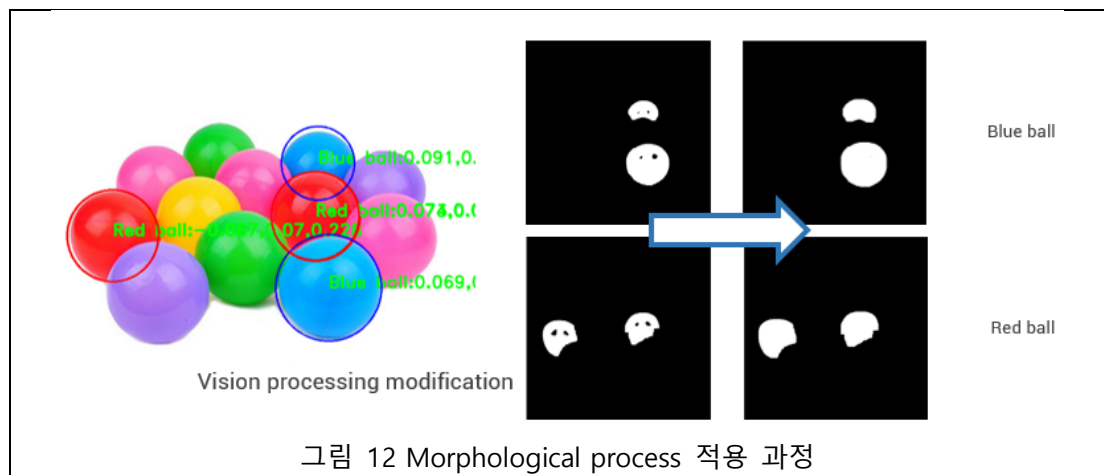
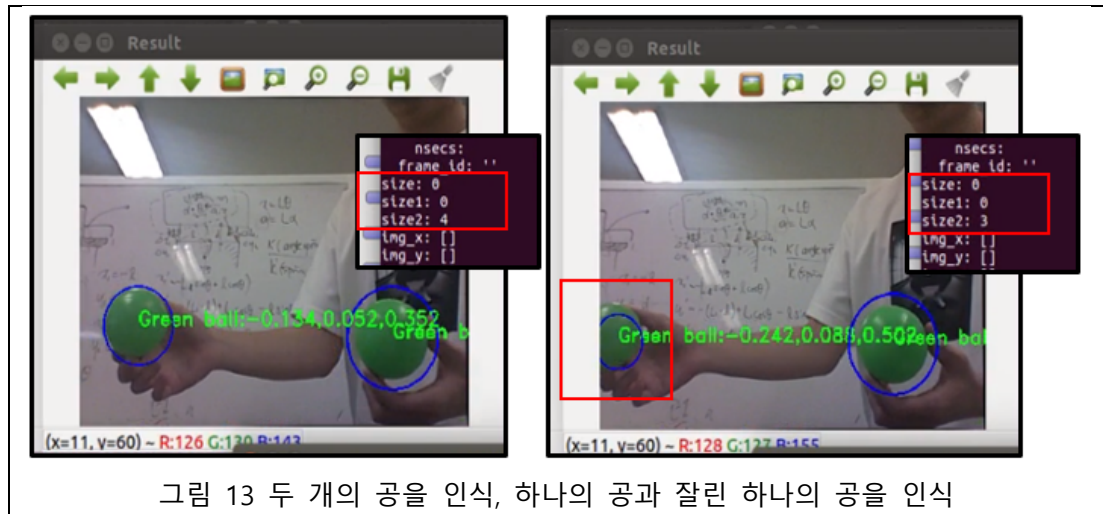


그림 12 Morphological process 적용 과정

두 번째 문제점은 공을 항상 두 개로 인식하는 것이다. 화면에서 공이 잘린 채로 촬영될 경우 공이 하나로 인식되는 대신 잘린 모양의 직경에 해당하는 공 크기로 인식한다. 이는 ROS code 내에서 공의 개수가 홀수일 경우를 따로 나누어 화면에서 잘렸다는 상태

를 알 수 있도록 하였다.



## F. ROS Integration & Picking algorithm

웹캠이 받은 image 정보를 image publisher에서 처리하여 image data를 ball detecting node로 보내고, ball detecting node에서 공의 화면상에서의 position을 ros integration node로 보내고, 이 position을 가지고 계산한 차체의 속도를 system의 모터에 명령하는 것이 system의 총 data flow sequence이다.

Picking algorithm에서 본 조는 가장 오른쪽 공을 집어오는 방법을 썼다. 가장 가까운 것을 먹는 것과 같이 다른 방법에 비해 이 방법이 효율면에서 떨어지지 않는 이유는, 목표물이 3개이기 때문에 항상 비슷한 삼각형 형태의 경로를 이동할 것이기 때문이다. 상세한 순서는 다음과 같다.

- 1) 가장 오른쪽 파란공을 향해 직진한다. 중간에 장애물(빨간공)이 있으면 피한다.
- 2) 파란공과 거리가 일정 수치 이상으로 가까워지면 blade를 돌리며 직진한다. 즉 공을 먹는다.
- 3) 공을 집은 다음 CCW로 돌며 다음 파란 공을 찾는다.
- 4) 이러한 과정을 두 번 더 반복한다. 마지막 스캐닝에서 360도를 돌 때 까지 파란공이 없다면, 초록 공을 보도록 회전한다.
- 5) 오른쪽 초록공을 향해 간다.
- 6) 일정 거리 이상 가까워지면, 두 초록공의 가운데를 보도록 & 두 공까지 거리가 같도록 움직인다.
- 7) 180도 회전한 뒤 후진하며 ball을 release한다.

### 3. Result

기록: 1분 40초

잡은 공: 파란공 3개 / 빨간공 0개

### 4. Progress report

<b>ME400 Capstone Design I</b>  <b>Progress Report – Group 9</b>	<b>Student Number</b>	<b>20150222</b>
	Name	김태연
	Part	LabVIEW

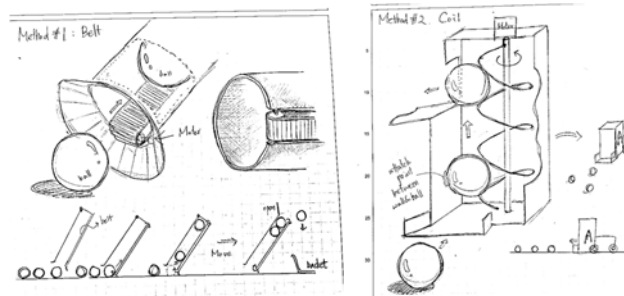
1<sup>st</sup> week (3/19 ~ 3/25)

3/13(화) 19:00 조별 회의를 통해 파트별로 배경지식을 조사하여 발표하기로 하였다.

3/14(수) ICL 특강: 발표 능력 피드백을 받았다.

3/15(목) 21:30 최민우 조원과 LabVIEW Assignment #1 – myRIO 예제 수행: myRIO의 가속도 센서와 LED, 버튼을 LabVIEW로 작동시키고 코딩하는 방법을 배웠다.

3/16(금) 19:00 조별 회의를 통해 다음주 화요일 회의까지 개인당 1~2개씩 pickup method를 조사하기로 하였다. 아래는 이후 회의에 가져간 method이다.



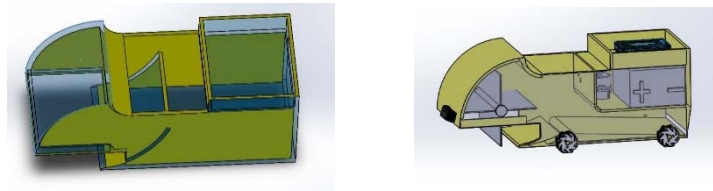
3/20(화) 19:00 조별 회의: Fan Rotating 방식의 ball collecting method를 우리 조의 system에 적용시키기로 하였다. 조의 조장을 맡았고, 이후 1차발표까지의 대략적인 계획을 잡았다.

3/20(화) 20:00 최민우 조원과 LabVIEW Assignment #2 – myRIO 예제 수행: myRIO의 LED등을 주어진 규칙(Time interval / Threshold of accelerometer)에 맞게 점멸하는 코드를 LabVIEW로 만들었다.

3/23(금) 09:00 장세용 과장님 LabVIEW 강의 수강: Control/Constant/Indicator, Node, Shift-register, Data flow following, Array & Cluster, Structures(While, For), State machine, Communication에 대한 내용을 들었다.

3/23(금) 13:00 조별 회의: Solidworks팀이 가져온 초안을 바탕으로 회의를 진행하였다. 상시 fan을

돌리는 방식의 energy consumption의 문제점을 지적하고, 공을 인식한 후 body에 contain할 때만 fan을 돌리도록 제안하였다.



2<sup>nd</sup> week (3/26 ~ 4/1)

3/25(일) 20:00 조별 회의: 회의가 진행되는 동안 결정된 1차발표의 흐름 및 klms의 공지사항 등을 요약하고 정리하여 조원들에게 공지하였다. 회의 결과 이예성 조원과 함께 1차발표 ppt 제작 및 발표를 맡았다.

3/26(월) 22:00 이예성 조원과 함께 1차발표 ppt 초안을 작성하였다.

3/27(화) 18:00 조별 회의: 오일권 교수님과 함께 식사 후 이예성 조원과 모의 발표를 진행했고, 피드백을 받았다. 피드백을 바탕으로 수정하여 수정본을 조원들에게 공지하였다.

3/28(수) ICL 특강: 1차발표 ppt 수정본을 수업시간에 영어로 발표하고, 교수님께 내용/발표능력 피드백을 받았다. 평가내용과 타 조의 idea들을 취합했고 이를 조원들에게 전달하였다.

3/29(목) 22:30 조별 회의: ICL 수업시간에 받은 피드백 등을 바탕으로 재 수정된 ppt로 조원들과 모의 발표를 진행하였고, 최종 피드백을 받아 ppt 최종본을 만들었다.

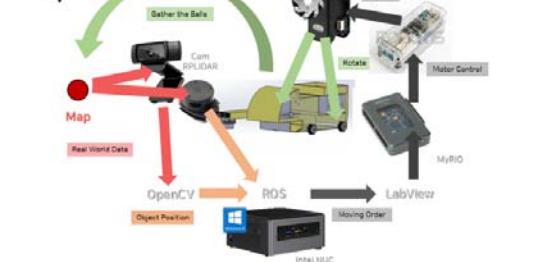
## Ball Picking System

Capstone Design I : 1<sup>st</sup> Presentation

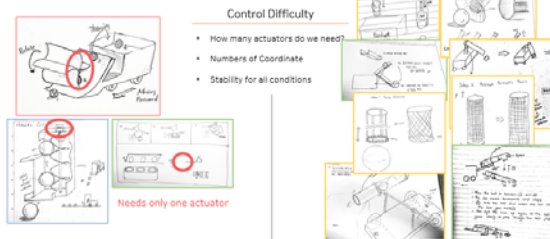


Group 9  
Prof. ILKWON OH  
TA JAESEONG OH, KANGKYU LEE  
Team member TAEYON KIM, TAEHONG KIM, VIVEK URADHAYA, YESUNG YU, SUHYUN LIM, WOOSAEK JUNG, MINWOO CHOI, JINYE HAN

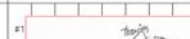
## 1. System Definition



## 2-2. Design Comparing

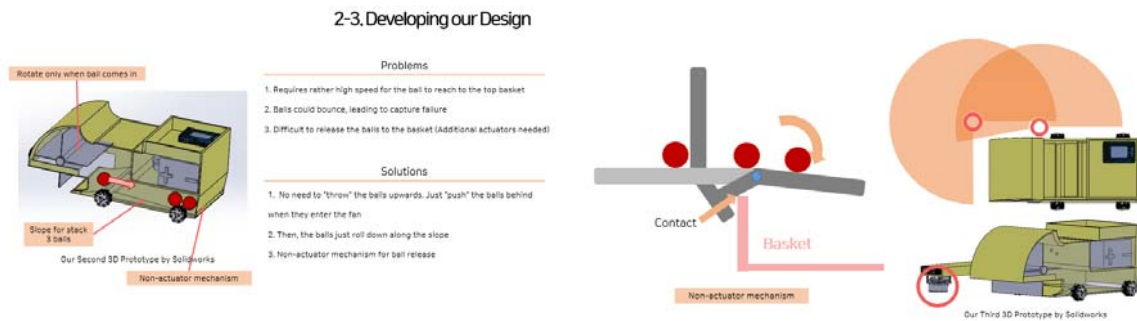


## 2-2. Design Comparing

Pugh's Decision Matrix		Idea #1~11										
Issue: Choose Picking-up function & form		#1									#10	#11
Overall Operation Speed	30	7	30	40	90	35	42	65	90	30	80	10
Energy Consumption	30	6	30	40	90	35	42	65	90	30	80	10
Robustness for Picking Up	20	7	30	40	90	35	42	65	90	30	80	10
Control Difficulty	10	5	30	40	90	35	42	65	90	30	80	10
Manufacture Cost	10	6	30	40	90	35	42	65	90	30	80	10
Total		31	64	43	63	42	50	54	50	45	60	71
Weighted total		64	43	63	42	50	54	50	45	60	40	71

Result of Pugh's Decision Matrix

#11 is the best method for picking up the balls.  
There are certain strengths and weaknesses.  
Identified a need to eliminate the weaknesses by further development

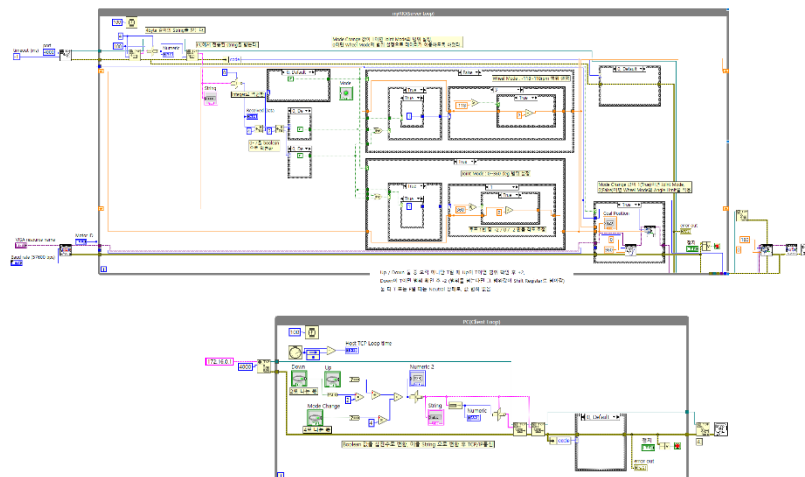


1차 발표 ppt의 일부

3/30(금) 1차 발표

3<sup>rd</sup> week (4/2 ~ 4/8)

4/2(월) 17:30~24:00 Labview assignment#3을 수행. PC와 myRIO사이의 TCP/IP통신을 통해 모터의 mode를 변환, 속도와 각도를 조절하는 client.vi와 server.vi 구현.



4/3(화) 19:30 조별 회의: 향후 일정 확인 및 2차 발표까지의 목표 및 상세 계획 확정. 회의 내용을 정리 후 조원들에게 공지. Labview part 목표: 모터 4개 구동 시험을 수행하고, ROS와 TCP/IP통신을 할 수 있도록 하는 것을 4/8(일)까지 수행

4/4(수) 16:00~21:30 Dymanixel 4개 ID 부여, 각 모터 별로 joint mode / wheel mode 제어 가능한 코드 설계. Wheel mode에서 Joint mode로 전환 시 구동한 만큼 영점으로 돌아가는 것을 확인, 이를 보정하기 위해 shift register로 wheel mode에서 position값을 저장하는 방식을 구현. DC/DC converter 구매를 위해 각 system에서 사용하는 전력 파악 Dynamixel과 myRIO가 합쳐서 약 100W, NUC와 RPLIDAR, webcam이 합쳐서 약 80W를 사용한다고 판단하였고, 이를 계산한 후 구매 담당 학생에게 자료를 송부하고 구매 전까지 지속적으로 Specification에 대한 논의 진행

4/5(목) 14:30~20:30 Server.vi를 만들어 NUC의 client에서 송출하는 xbox control 신호를 수신하여

motor 구동 시도(실패)

1. 예제를 변형하여 xbox controller 신호를 array 로 가정, 24 개의 array 를 string 으로 변환하여 수신한 뒤 어떤 데이터를 받으면 일단 모터가 돌아가도록 설계하였다.
2. 데스크탑 컴퓨터를 사용하지 않고 server 를 켜놓을 수 없음을 깨달았고, myRIO 자체에 startup vi 를 직접 구워서 바로 실행시키는 방법을 Youtube 영상에서 찾았다.
3. 직접 구워서 server 를 열었고, NUC 의 ROS 인터페이스에서 연결을 성공하였다는 메시지를 확인하였다. 하지만 데이터 수신은 되지 않았고, data type 변환의 문제라 판단하였다.

#### 4<sup>th</sup> week (4/9 ~ 4/15)

4/10(화) 19:30 조별 회의: 지난주에 계획했던 목표에 대한 각 파트 진행상황 보고 후 시험기간 전까지 목표 설정(Labview part: xbox – motor 연결 완성). 이강규조교님의 합류로 현재 상황 간단히 설명 후 회로 설계와 전반적인 system에 관한 조언을 들음.

4/11(수) 지속적으로 컨버터 구매를 위한 회의 진행. Specification을 다시 파악하여 12V/18V 두가지 컨버터만 필요하다고 판단하였고, 조사 결과 Murata사의 DC/DC컨버터를 구매하기로 결정.

4/13(금) 16:00~18:00 Xbox controller의 신호가 24\*4 또는 24\*4\*8 단위인 것을 공지받고 이를 적용하여 통신을 다시 시도함.(성공)

1. 기존의 TCP/IP 예제는 두 개의 TCP Read 함수를 썼는데, 데이터 길이를 계속 시험해보고 맞는 걸 채택하는 방식에서 이는 필요없다고 판단하여 지웠다.
2. Bytes to read 에 넣는 수를 다양하게 바꿔보았고, 데이터가 어떻게 잘리고 밀리는지 관찰하였다.
3. 24\*4 를 해보니 데이터가 정확히 수신됨을 확인하였다. 하지만 데이터가 잘게 잘리는 과정에서 버퍼가 발생(약 1000 개 이상씩 잘리면 버퍼가 없음)하였고, 이를 해결하기 위해 기존의 100ms 이던 loop time 을 10ms 로 바꾸어 delay 없는 통신에 성공하였다. 이는 myRIO 에 무리를 줄 수 있기에, 대부분의 계산은 ROS 에서 하기로 결정했다.
4. 모터를 연결해서 임의의 크기와 방향을 가진 joystick 을 차체의 움직임과 대응되도록 움직이는 것을 확인하였다.

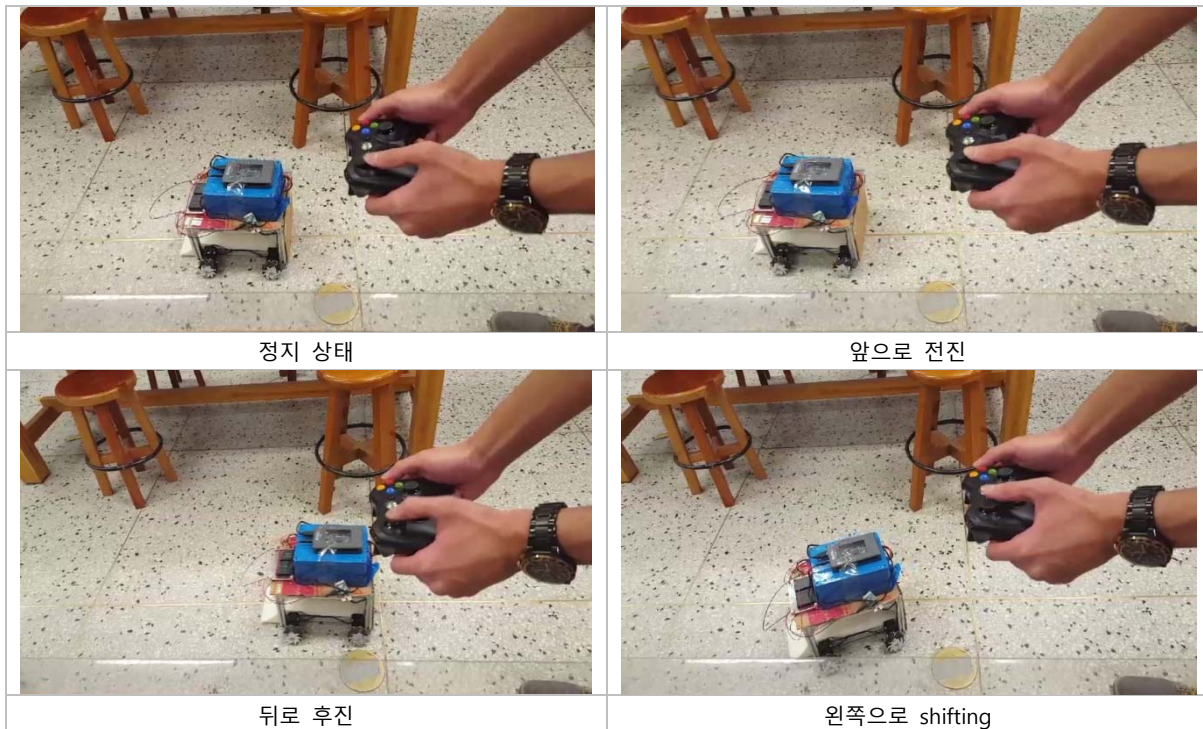
#### 5<sup>th</sup> week (4/16 ~ 4/22) – 시험기간



활동한 내용이 없습니다.

6<sup>th</sup> week (4/23 ~ 4/29)

4/23(월) 16:10~19:00 Solidworks팀이 PMS를 포함한 차체 제작을 완료하였기 때문에 Xbox controller로 조종을 시도. 제작 과정에서 이전에 가정했던 front-rear, left-right 바퀴 넘버링이 뒤틀려 이를 조정 후 모터 재연결. 이 과정에서 3pin-connector의 길이가 부족하여 전선 3개를 꽃아 해결. Xbox controller의 신호를 받아서 차체를 움직이는 것에 성공, 영상 촬영(아래 캡처화면). 하지만 shifting 시 움직임이 원활하지 않았고, 지면의 상태와 차체의 무게중심에 의한 오차일 것이라 판단.



4/24(화) 19:30~22:00 조별 회의 내용을 정리하여 조원들에게 공지함. Labview파트는 Xbox controller의 영상을 다시 찍기 위해 xbox controller로 회전 또한 가능하도록 코드를 수정하고, Xbox controller를 통해 전송하는 24개의 input 대신 ROS에서 직접 주는 4개의 motor speed(rpm)값으로 받는 코드를 제작하기로 함. 또한 한 명이 담당하고 있던 path planning 에 대한 조모임을 만들어 일정을 조정. 회의 중 나온 추가 구매 목록을 작성하였고, 우리 조의 system에 맞는 제품을 찾는 일을 맡음.

4/24(화) 22:00~24:00 히트싱크, 쿨러, 서멀패드, 적외선 센서를 찾아 주소를 넘기고, 추가적인 팬 구성을 위해 고속 회전에 특화된 (약 450rpm) 다이내믹셀(MX 12W)의 스펙을 분석하여 주문하도록 함.



MX-12W > MX-시리즈 | 로보티즈 -  
[http://www.robotis.com/shop/item.php?it\\_id=902-0086-000](http://www.robotis.com/shop/item.php?it_id=902-0086-000)

- 4/25(수) 16:00~18:40 ROS팀의 이예성 군과 ROS코드를 같이 수정하며 24개의 신호를 주던 client를 4개의 데이터를 주는 client로 바꾸는 것을 시도함. 이전의 무게중심과 모터 축 뒤틀림 등의 문제점을 solidworks팀이 개선하였고, ROS에서 직접 송신하는 신호를 받아 차량이 움직이는 것을 확인함. 시간관계상 전진만 시도하였고, 추후 ROS코드로 구현한 4방향 이동, 회전, 대각선 이동을 촬영하기로 함.
- 4/25(수) 19:00~20:00 ICL특강 - 수업의 변경사항을 공지 받고 이를 조원들에게 전달함: 발표자가 수업을 수강하지 않더라도 발표자가 참석하여 교수님께 발표력에 대한 피드백을 받는 것이 가능했으나, 이제는 발표를 하지 않더라도 발표자의 script로 수업 수강자가 피드백을 받는 것으로 변경됨.
- 4/26(목) 14:30~16:30 이예성 군이 제작해온 코드로 각 방향 이동 및 회전 이동을 테스트하였으나, 차량이 이상하게 움직였음. 살펴본 결과 모터와 바퀴 사이의 연결부에 문제 - 회색 연결부의 모터 접촉부분 기어가 가 생겼고, 모터 축이 헛도는 것을 확인.
- 4/28(토) 19:00~22:00 조별 회의를 통해 '카메라가 공을 완벽하게 인식한다'는 가정 하에 차체가 과제를 수행함에 있어 발생할 수 있는 모든 문제점을 파악하고, 이를 대처하는 방법을 모색하였고, 회의 내용을 바탕으로 순서도를 작성해봄. 작성 후 눈에 보이는 또 다른 예외사항들을 점검.



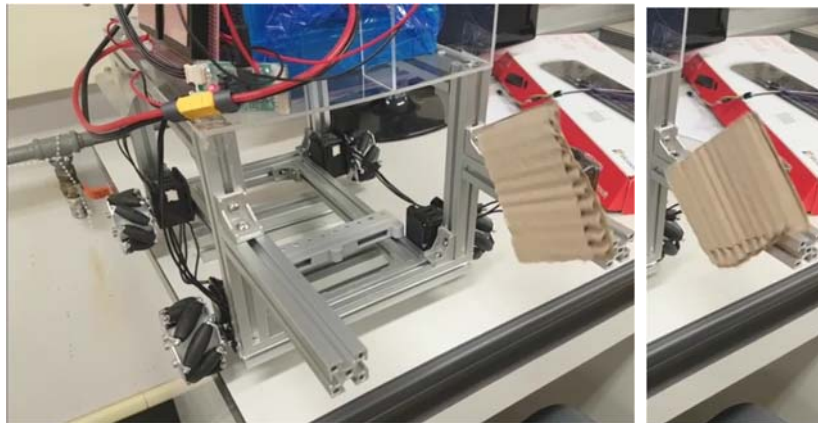
5/2(수) 15:20~16:30 이전에 발생했던 메카닉 휠 연결부의 문제를 solidworks팀의 vivek군이 수리했고, 작동을 확인하기 위해 ROS팀과 함께 정상 구동을 확인. 이후 Xbox 구동영상을 재촬영(회전 포함)하고, ROS에서 자체적인 code로 control 하는 영상을 성공적으로 촬영. ROS팀에 합류하여 ROS내 사용하는 code를 이해하고 같이 수정하는 작업을 시작함.

5/2(수) 18:00~19:40 조원들끼리 모여 ppt 수정 작업.

5/2(수) 19:50~20:30 PPT초안을 바탕으로 ICL 특강 수업에서 모의 발표 진행.

5/2(수) 20:30~22:30 새로 온 Dynamixel MX-12W에 ID를 부여. 모터 5개를 control하도록 Labview code를 수정한 후 전체 연결을 시도했으나 연결 방법의 문제로 실패. U2D2나 모터 전원칩 간의 연결 문제라 판단

5/3(목) 01:00~02:00 다시 생각해본 연결방법으로 MX-12W 연결을 재시도하여 성공. 이로써 System 전면부의 fan, 즉 ball picking system 구현이 가능해짐.



5/3(목) 14:40~16:30 오일권교수님 lab의 FRF실험기자재를 통해 system의 진동 특성을 파악.

5/3(목) 16:50~17:20 Hubo lab에 방문하여 열화상카메라로 system을 일정 시간 구동 시 각 부분의 온도를 파악. 예상 외로 회로 상에서 연결한 기기(NUC)이 제외되었음에도 15V converter에서 가장 높은 열 발생.

5/3(목) 17:30~5/4(금) 03:00

- "파란공이 가까울 경우 3초간 fan을 회전시키며 직진" 구현 시도: Labview code 내에서 시간 delay를 만들고, 통신 데이터 수를 늘려 mode를 전환할 수 있도록 한 후 ROS팀에 합류하여 TCP/IP통신의 client 코드를 함께 수정. 하지만 delay이후 송신데이터가 이상해지며, 통신 자체의 delay가 다시 발생하였고, 이를 수정.
- OpenCV파트에 합류: 창시구실에 맞는 HSV값을 직접 조정, 이후 ball detect 후 ball을 collecting하는 상황을 ROS팀과 함께 점검.
- LabVIEW파트 ppt를 재수정.

- System의 바퀴가 다시 헐거워졌고, 문제를 직접 파악해본 결과 부품의 마모가 있더라도 볼트를 조이면 구동에는 문제 없을 정도로 고정하는 것이 가능하였음.

- 마지막 발표 점검 및 코멘트

5/4(금) 10:00~13:00 2차 발표 진행.

#### 8<sup>th</sup> week (5/7 ~ 5/13)

5/8(화) 19:30~22:00 조별 회의: 발표에 대한 피드백과 함께 앞으로 해야 할 task를 점검. System의 이동속도를 늘리기 위해 Gear box 대신 메카넘휠의 직경을 늘리기로 함. Labview는 이후 할 일이 많지 않아 전에 하던 대로 자연스럽게 소프트웨어 보완팀에 합류함. Labview에서 delay를 주는 방식에는 통신적으로 문제가 발생하였고, sleep()함수를 이용한 delay를 제안하였고, 성공적으로 구현됨.

5/9(목) 21:00~23:00 소프트웨어팀 회의: OpenCV의 code와 ROS code를 계속 고쳐가며 system이 ball을 온전히 picking하기 위한 tuning 진행

5/10(금) 10:00~13:30 소프트웨어팀 회의: Tuning 이어서 진행. Data integrating node의 코드 상 문제점을 발견하였고, ROS 담당자가 공 3개 중 선택하는 code를 추가하기로 함과 동시에 각자 code상의 문제점을 파악하기로 함.

#### 9<sup>th</sup> week (5/14 ~ 5/20)

5/15(화) 16:00~18:00 소프트웨어팀 회의: 빨간공을 대각선으로 이동하면서 피하고, 파란공 3개를 성공적으로 모으는 것을 확인. 하지만 초록공을 디텍트하고 직진하는 코드에서 에러가 발생함을 확인.

5/16(수) 19:00~21:00 소프트웨어팀 회의: Wheel kinematics를 사용하여 현재 system이 initial point에서부터 얼마나 돌아갔는지 계산하는 코드를 고안. 계산한 결과를 matlab 함수로 작성하였고, 돌아가야 하는 각도와 모터 각속도에 따른 loop 횟수와 1번의 loop당 돌아가는 각도를 계산함. 이를 적용하여 실제로도 잘 맞음을 확인.

5/17(목) 19:30~21:00 하드웨어팀에서 새로 주문한 바퀴(D=152mm)를 달았고, 이 system이 가진 새로운 수치로 wheel kinematics 보정. 하지만 system을 구동하던 도중 하드웨어팀이 새로운 바퀴를 알루미늄 프로파일에 1축 고정을 시켜 모터가 하중을 받아 돌아가는 문제가 발생하였고, 바퀴 연결도 불안정함을 확인하여 볼트를 이용한 새로운 연결 방법을 제안.

5/18(금) 12:00~14:00 초록공을 detect하고 잡으러가는 상황을 시간 관계상 system의 하드웨어 없이 확인해야 했고, 현재 바퀴의 각속도에 따른 system의 속도 및 각속도를 적분하여 시

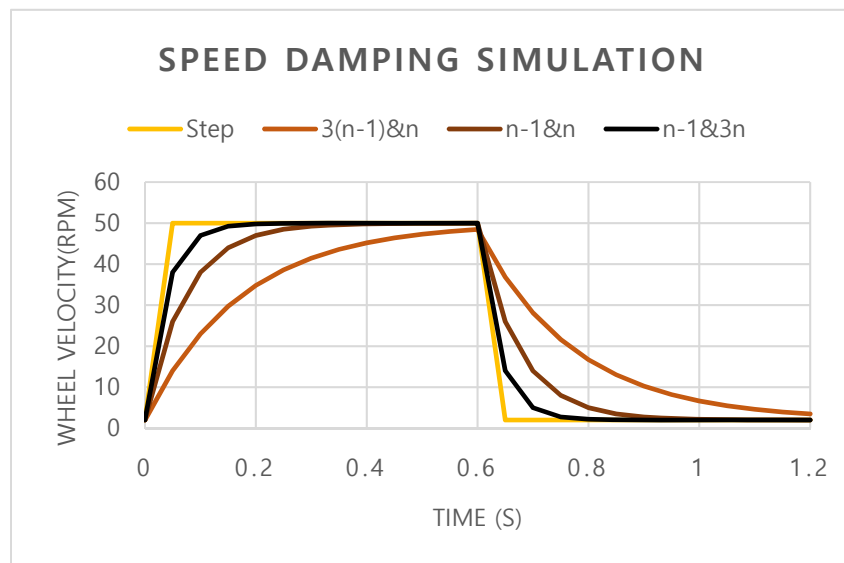
야에 보이는 초록공에 따른 차체의 위치 변화를 알려주는 코드를 제작.

5/18(금) 19:00~20:00 공동강의실 데모: 어떠한 position에 있더라도 공을 일정하게 detect할 수 있도록 demo 경기장 바로 위의 조명을 제외한 나머지를 끄도록 제안하였고, HSV값을 조정한 이후 먼 거리에서도 모든 ball이 잘 detect됨을 확인.

5/20(일) 16:00~19:00 소프트웨어 팀 회의: 차체 바퀴 높이가 일정하지 않아 분해 후 재조립. 그럼에도 이전에 잘 맞던 각도 계산이 잘 맞지 않음 - 바퀴가 커서 wheel과 바닥 사이에 불규칙한 slip이 발생하고 있다고 추측. 각도를 예상해 초록공이 "있을" 위치를 바라보게 하는 코드를 빼고, ball releasing에 반드시 필요한 180도 회전은 trial error로 계속 보정. 초록공의 detect에는 문제가 없으나 코드 상의 문제로 원하는 움직임을 보이지 않음.

10<sup>th</sup> week (5/21 ~ 5/27)

5/21(월) 19:00~21:00 소프트웨어 팀 회의: 이예성 군이 수정해온 "초록공으로 가기"가 원활히 작동됨을 확인. 급제동, 급가속의 문제로 캠을 비롯한 차체가 덜컹거리는 것을 해결하기 위해 ROS에서 보내는 모터 rpm에 해당하는 변수에 바로 상수를 대입하던 것과 달리,



이전에 저장된 값과 명령을 주는 값(위 그림은 3:1, 2:2, 1:3의 비중에 해당하는 속도 그래프)을 평균 내어 새로 대입함으로써 damped velocity 개형을 구현. 이를 적용한 결과 가속, 제동에 의한 덜컹거림이 사라짐을 확인. 하지만 알고리즘 상으로 공이 가운데 올 때까지 회전하는 구간에 damping이 많으면 오히려 역효과를 부르기에 이 구간의 damping값을 줄임.

5/22(화) 13:30~15:00 소프트웨어 팀 회의: 수정을 거듭하여 최초로 전 과정을 성공. 약 1분 30초 소요. Ball releasing 부분을 하드웨어팀이 다시 만들고 있어 system의 뒤부분과 초록공의 중점(바구니)가 맞닿는 것까지 확인.

5/23(수) 20:00~21:30 소프트웨어 팀 회의: 코드 상의 값을 계속 tuning함. 이예성 군이 지속적인 수정을 해서 와주었기 때문에 수월하게 진행. 차체의 바퀴 buckling이 심해져 Xbox로 실험한 결과 직진의 경우 한쪽으로 휘고, 좌우 shifting은 차체 뒷부분이 더 많이 이동 (아주 낮은 속도에서는 적당히 평행이동함을 확인하여 코드 내 shift 이동 속도를 대폭 낮춤). 다시 분해 후 재조립하고, 일단 모터 고정판이 돌아감을 막기 위해 임시방편으로 보충재를 끼워넣음. 하드웨어 팀이 전면 보완 예정.