

ENGINEERING NOTE
TALOS
#5073

2023-24 KOREA ROBOT CHAMPIONSHIP
KOREA SCIENCE ACADEMY OF KAIST

#5073

CONTENTS

ABOUT US	... 5
ROBOT & STRATEGY	... 13
EN - BUILDER	... 19
EN - PROGRAMMER	... 47
CONCLUSION & PLANS	... 67

TEAM TALOS

ABOUT US

2023-24 KOREA ROBOT CHAMPIONSHIP
KOREA SCIENCE ACADEMY OF KAIST

TALOS 소개

TALOS는 KAIST 부설 한국과학영재학교의 로봇공학 연구회 KROS에서 출전하는 FTC 팀으로, 수리정보과학부 김호숙 선생님과 학부의 지원을 받고 있습니다.

TALOS는 그리스 신화 속 헤파이스토스가 만들어 낸 최초의 로봇으로, 헤파이스토스가 이성을 통해 창조한 탈로스처럼 멋진 로봇을 만들어 내겠다는 의지를 표방합니다.

연혁



**2018-19 SEASON
Rover Ducks**



**2019-20 SEASON
SKYSTONE**



2021-22 SEASON FREIGHT FRENZY



2022-23 SEASON POWERPLAY

2023-24 SEASON CENTERSTAGE #5073



팀원 소개

TALOS는 n명의 Programmer와 n명의 Builder로 구성되어 함께 작업합니다. 2023-24 시즌의 경우 23학번 2명, 22학번 8명, 21학번 5명으로 구성되었습니다.

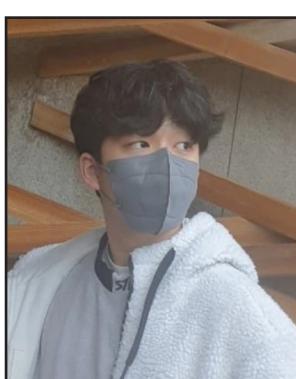
Captain - 팀 리더



민지홍
Programmer, 21
:: 프로그래밍



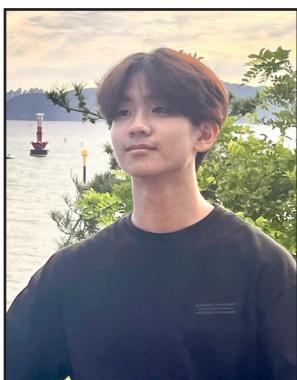
유태우
Programmer, 22
:: 프로그래밍, Auto 개발
:: 드라이버



홍정우
Builder, 23
:: 리니어 슬라이드 제작
:: 비행기 발사대 제작

Programmer

프로그래머는 로봇의 코드를 작성합니다. 컨트롤러와 로봇 간의 통신이 가능하도록 하며, Auto 모드 동안 로봇이 지정된 미션을 수행하도록 합니다.

**김준성****Programmer, 23**

:: 프로그래밍

:: 엔지니어링 노트 편집

Builder

빌더(엔지니어)는 로봇을 설계 및 제작합니다. 본체, 리니어, 이터를 제작하는 세 팀으로 나누어 작업했으며, 전반적인 설계와 제작, 조립 등을 맡습니다.

**박상현****Builder, 21**

:: 차체 디자인

**박찬****Builder, 21**

:: 차체 디자인 및 본체 제작



심규서
Builder, 21

:: 차체 디자인 및 본체 제작



오민준
Builder, 21

:: 이터 설계



강현빈
Builder, 22

:: 리니어 슬라이드 제작
:: 비행기 발사대 제작



김민권
Builder, 22

:: 3D 프린팅 담당
:: 종이비행기 테스트



김수기
Builder, 22

:: 이터 설계 및 제작



이규진
Builder, 22
:: 차체 제작 및 디자인



장호원
Builder, 22
:: 이터 설계 및 제작
:: 드라이버



정의진
Builder, 22
:: 차체 제작 및 디자인
:: 포스터 및 발표준비



최성빈
Builder, 22
:: 아크릴 프린팅 담당
:: 설계 및 디자인

지도

김호숙 선생님 (Advisor)

KAIST 부설 한국과학영재학교 수리정보과학부

이승찬 (Advisor, 19)

KSA 19학번, 2021-2022 시즌 캡틴

SPONSER



KAIST부설

한국과학영재학교



과학기술정보통신부

Ministry of Science and ICT

TEAM TALOS는..

KAIST 부설 한국과학영재학교의 연구회 KROS의 연구회원으로 구성되며, 본 연구회는 한국과학영재학교와 과학기술정보통신부의 지원 하에 운영되고 있습니다.

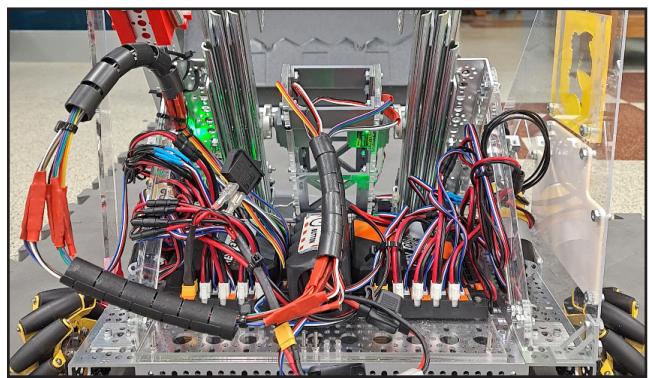
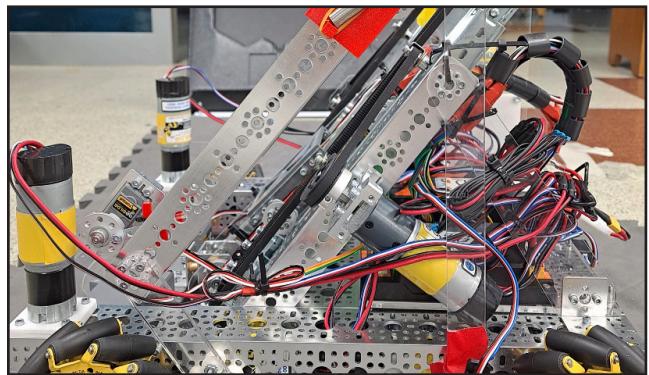
TEAM TALOS

ROBOT

& STRATEGY

2023-24 KOREA ROBOT CHAMPIONSHIP
KOREA SCIENCE ACADEMY OF KAIST

TALOS 로봇 소개 : “우이”



FTC 2023-2024 시즌 - 로봇 “우이”

제작 기간 : 총 13일 (1월 12일 ~ 24일)

TEAM TALOS는 주어진 미션을 최대한 효율적으로 수행할 수 있도록 로봇 “우이”를 설계하고 제작하였습니다. 해외 팀들의 경기 영상을 분석하여 경기 방식과 현실적인 문제점들을 확인하고, TALOS만의 독창적인 아이디어로 로봇을 구상하였습니다. 총 13일의 기간 동안 모든 팀원의 협업과 노력을 바탕으로 완성도 높은 로봇의 제작을 마무리할 수 있었습니다.

본 로봇은 goBILDA의 차체를 기반으로 플라스틱 (3D 프린트), 아크릴 평판 등 다양한 부품들을 결합해 제작하였으며, 무엇보다 정해진 규격 내에서 좌우 대칭과 무게 중심이 맞으면서 안정적인 플레이를 할 수 있도록 만드는 데 심혈을 기울였습니다.

로봇 “우이”는 본체, 리니어 슬라이드, 집게, 이터, 그리고 비행기 발사대로 구성됩니다.

1. 본체

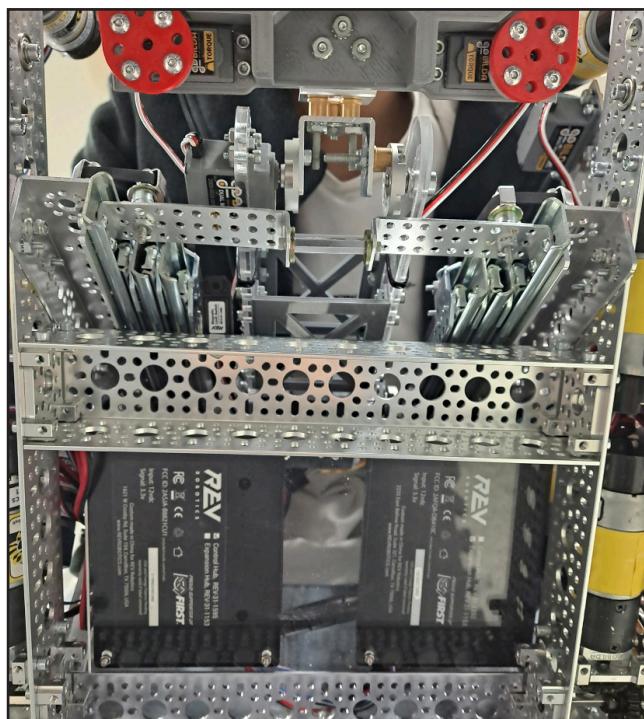
차체의 기초 프레임과 바퀴로 구성되어 있는 구조체이며, 리니어 슬라이드와 이터, 비행기 발사대 등이 연결되어 있습니다. 이 외에도 Control Hub, Expansion Hub, 광학 거리 센서, 배터리가 탑재되어 있습니다. Autonomous Period에서 회전하지 않고 Team Prop을 인식할 수 있도록 광학 센서를 우측과 정면에 장착하였습니다.

2. 집게

픽셀을 집고 넘기는 부품으로, 리니어에 서보 모터로 연결되어 있습니다.

- 팔 (Arm) : 집게 파트를 로봇의 앞, 뒤로 회전
- 손목 (Wrist) : 집게와 백보드의 각도가 일치하도록 각도 조절
- 손가락 (Finger) : 픽셀을 잡고 놓는 역할

각 파트는 서보 모터로 구동되며, 다양한 버전을 테스트하며 육각형 모양의 픽셀을 가장 안정적으로 집을 수 있도록 설계하였습니다.



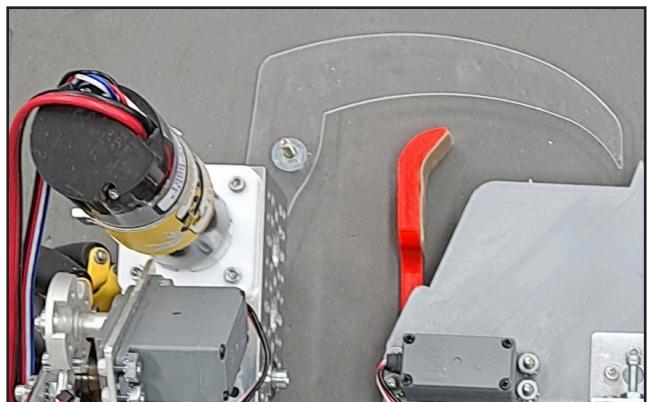
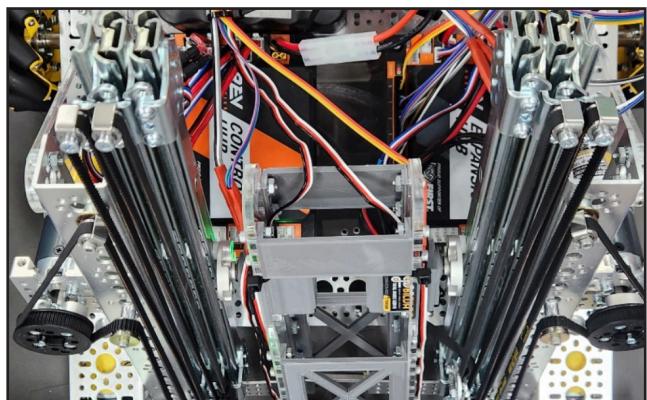
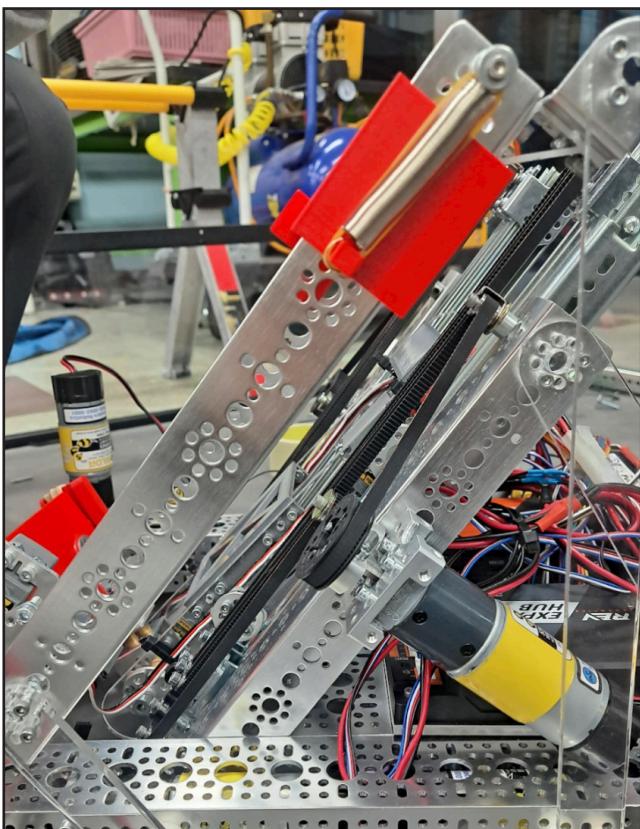
3. 리니어 슬라이드 : 4-stage viper-slide (belt-type, GoBlida)
 백보드에 픽셀을 놓을 수 있도록 집게를 적절한 높이까지 올리는 역할과
 엔드게임에서 로봇 차체를 Truss에 걸어 올리는 역할을 합니다. 차체 양
 쪽의 DC 모터에 체인을 연결해 최대 3단까지 확장됩니다.

4. 스택 이터

스택으로 쌓여 있는 픽셀 혹은 집게로 바로 집기 어려운 위치의 픽셀을 모아주는 역할을 합니다. 아크릴로 제작된 낫 형태의 구조물이 DC 모터에 연결되어 구동되며, 정해진 각도 내에서 회전합니다.

5. 비행기 발사대

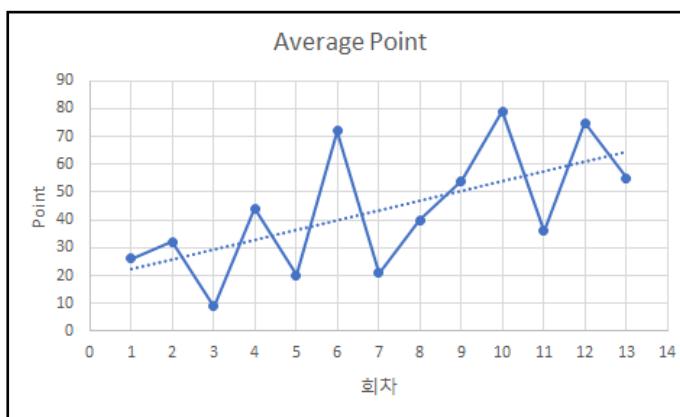
로봇에 미리 장착된 종이 비행기를 발사하는 역할을 하며, 본체에 고정되어 있습니다. 3D 프린트한 비행기 케이스는 고무줄과 용수철에 연결되어 있으며, 서보 모터를 이용해 위치를 고정하였습니다. 종이비행기가 일정한 위치에 도달할 수 있도록 적절한 각도로 조정하였습니다.



STRATEGY : 플레이 전략

플레이 전략

TEAM TALOS는 Center stage에서 좋은 점수를 얻기 위하여 전략을 구상하고, 전략을 투영한 로봇을 제작하였습니다. 로봇이 완성된 뒤에는 꾸준히 연습하고 전략을 수정하는 것을 반복하였습니다.



TEAM TALOS는 연습을 여러번 진행하고 결점과 실수를 점점 줄여나가며 발전해나가고 있습니다. 현재 13번의 연습을 진행하였으며, 결과는 좌측의 사진과 같습니다.

연습을 진행하며 그래프에 표시된 바와 같이 점수를 올릴 수 있었습니다.

Center stage에서는 Driver controlled period도 중요하지만, Autonomous period와 Endgame period의 점수가 큰 비중을 차지한다고 생각하였습니다. 그렇기에 Autonomous period와 Endgame에서 받을 수 있는 점수를 최대한 가져가는 방향으로 전략을 구상하였습니다.

1. Autonomous Period

이 구간에서는 30초 안에 거리 센서로 오브젝트를 감지하고, 오브젝트가 있는 선에 보라색 픽셀을 위치하여 20점, 정해진 위치에 보라색 픽셀을 backdrop에 놓아 20점, 그 후 선 안에 주차하여 5점을 받아 45점을 받아가는 것을 목표로 하고 있습니다.

2. Driver Controlled Period

이 구간 1분 30초 동안은 모자이크를 완성하기 위해서 human player가 주는 픽셀을 두 개 가져와 모자이크를 하나 완성하고, 그 뒤에는 backdrop에 비교적 가까운 pixel stack을 스택 이터를 통해 가져오는 것을 반복하여 backdrop에 픽셀을 8개 두는 것을 목표로 운영할 계획입니다. 총 픽셀 8개로 24점, 모자이크 완성 1회로 10점, 높이 점수로 10점을 받아 총 44점을 받는 것을 목표로 하고 있습니다.

동맹 팀과 합을 맞춰 점수를 최대한 많이 받을 수 있도록 유동적으로 전략을 변경할 계획입니다.

3. Endgame

이 구간에 해당된 30초 동안은 적절한 위치에서 비행기를 발사하여 30점, 남은 시간 동안 봉에 매달리는 것을 시도하여 20점을 받아 총 50점을 받는 것을 목표로 하고 있습니다.

TEAM TALOS

ENGINEERING NOTE

1 BUILDER

2023-24 KOREA ROBOT CHAMPIONSHIP
KOREA SCIENCE ACADEMY OF KAIST

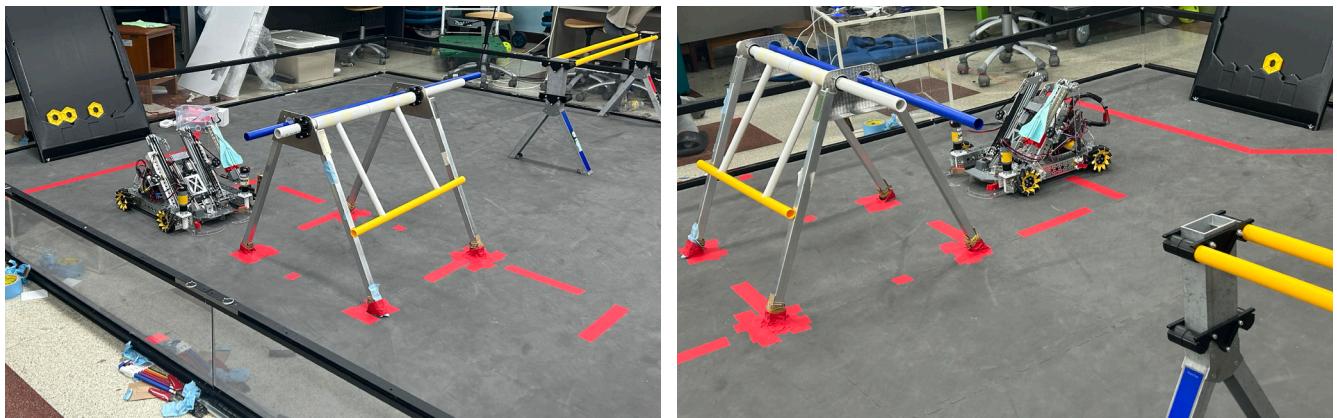
1월 13일 (토)

1. 리니어 슬라이드 제작

goBUILDAs의 4 Stage Viper-Slide (Belt-Type) 리니어를 구입 후, 사이트의 제작 설명서를 참고하여 제작하였다. 리니어 슬라이드를 로봇 차체 위에 설치할 경우 Truss의 높이인 35cm보다 높음을 확인하였다.

2. 경기장 제작

로봇을 제작하며 실제 경기와 유사한 환경에서 구동을 테스트하기 위해 경기장을 제작하였다. 경기장 세트의 1/4를 구입해 중앙의 Truss를 구축했으며, 작년의 경기장 세트 일부와 PVC 파이프 등을 이용해 작은 Truss 하나를 더 구축하였다.



1월 14일 (일)

1. 전체적인 로봇 설계

전반적으로 로봇을 설계하며 픽셀을 집고 배출할 수 있는 다양한 방식을 고민하고, 각각의 장단점을 고려하여 토의하였다.

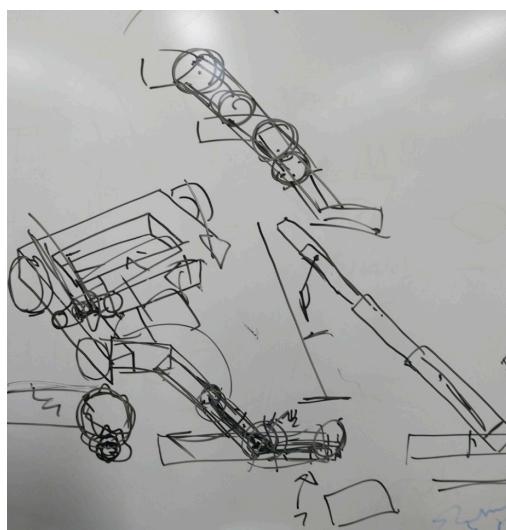
1) 집게형 vs 로터형

[집게형]

구조 :: 집게를 사용하고 리니어와 집게를 잇는 팔 형태를 사용

장점 :: Autonomous period 때 팔 형태를 사용하면서 리니어에서 멀리 떨어져서 픽셀을 드랍할 수 있음
 → 낮은 위치에서 떨어뜨리면 원하는 위치에 픽셀을 둘 수 있으며, 높은 위치까지 올릴 필요가 없으므로 두는 데까지 걸리는 시간이 감소함

단점 :: 팔 사용 시 토크의 증가로 인해 모터에 큰 부담이 가며, 불안정성이 증가함. 이를 해결하기 위해 기어를 사용하게 되면 집게를 돌리는 시간이 오래 걸림. 추가적으로 집게로 제작되었을 때, 너비에 맞추기 위해서는 먼저 제작한 가로로 만든 집게를 제작할 수 없음.



보완점 ::

기존 집게형의 경우 팔 부분과 손목 부분, 집게 부분 모두 서보모터가 필요하여 리니어 슬라이드에 고정된 서보모터에 무게로 인해 부하가 많이 걸리게 됨.
 또한 중력을 거스르는 방향으로 꾸준히 힘을 줘야 하므로 돌리는데 시간도 많이 걸림.

→ 집게를 리니어의 밑으로 통과시키게 되면 중력을 이용해서 서보모터를 돌리므로 모터에 가해지는 부하도 적고, 손목 부분 서보모터를 없애도 되므로 시간도 절약됨.

[로터형]

구조 :: 회전하는 로터가 바닥에서부터 픽셀을 잡아 트레드 밀 위로 올리면, 트레드밀이 돌면서 바구니까지 픽셀을 옮기고, 리니어에 고정된 바구니를 옮겨 판 위에서 밑을 열어 떨어뜨림

장점 :: 상대적으로 픽셀을 로봇에 로딩하기 위해 픽셀 근처로만 이동하면 되기 때문에 시간이 감축됨. 또한 팔이 없으므로 모터에 추가적인 부담이 가해지지 않음, 즉 집게형보다 안정적임

단점 :: 이터 부분에서 리니어 슬라이드의 바구니로 픽셀을 옮길 때 바구니에 정확히 로딩되는 것을 기대하기 어려움. 또한 바닥에서 바구니까지 동력원으로 로터 혹은 컨베이어 벨트를 사용해야 하는데, 무게가 증가하거나 있음. 또한 리니어 슬라이드에 바구니가 바로 고정되어 있어 판에 닿아 안전하게 픽셀을 두기 위해서는 리니어 슬라이드를 최대치까지 옮겨야 하므로 시간이 소요됨.

2) 바구니 가로 vs 세로

이터에서 올라온 픽셀을 담을 바구니의 목업 및 아크릴 도면 제작.

논의 부분 :: 이터에서 올라온 픽셀이 하나씩 들어갈 때, 원하는 위치에 가로로 정확하게 2개가 배치될 수 있는가.

대안 :: 가운데 부분에 픽셀 한개만 들어갈 수 있는 입구를 만든 뒤, 삼각형을 맞춰 픽셀이 움직일 수 있는 부분을 제한함. 또한, 이터에서 먹은 픽셀의 경로를 제한하여 단순한 직사각형 모형으로 만드는 것이 제안되었다.

리니어 슬라이드에 부착할 바구니의 크기를 최소화 하기 위해 직사각형 모형에 가림막을 만들어 박스로 목업을 제작.

추가적으로 아크릴로 사이드와 밑부분을 설계 제작. 각 부분의 나사 구멍을 추가해 바닥을 서보 모터로 여는 것을 고려하여 서보 모터를 부착할 곳을 정함. 추후 프린트 후 조립 예정.

[가로]: 가로로 2개를 로딩

장점 :: 높이가 낮아 리니어에 부착하기 편리함. 서보를 2개 사용하면, 픽셀을 하나씩 드랍할 수 있어 모자이크를 만들 때 미세 조종에 용이함.

단점 :: 픽셀 2개가 들어올 때 두개가 겹칠 가능성이 높음. 이를 개선하기 위해 여러 장치가 필요해보임.

[세로]: 세로로 2개를 로딩

장점 :: 걸릴 일이 없음

단점 :: 세로로 길어서 이터에서부터 픽셀 이동거리가 길어져, 이터의 힘만으로 올릴 수 없어 보임. 또한 리니어 슬라이드에 연결되기 어려움.



3) 바구니 → 픽셀 로딩 방법

[바구니의 뒷 부분으로 배출]

이터로부터 픽셀을 바구니로 받아 바구니의 각도를 고정한 상태로 리니어로 올린 뒤 바구니의 아래부분을 개방하여 픽셀을 배출

[바구니를 돌려서 배출]

이터로부터 픽셀을 바구니로 받은 뒤 리니어로 올린 뒤 바구니 자체를 뒤집어 픽셀을 배출

2. 리니어 슬라이드 수정 및 본체 제작

리니어 슬라이드를 4단에서 3단으로 수정하고 모터의 위치를 변경했으며, 35cm의 허들 높이를 준수할지 토의하였다. 토의한 결과 35cm를 넘지 않는 방법을 택하여 제작하였다.

1) 리니어 슬라이드 수정

i) 4 stage → 3 stage

판의 높이와 리니어 슬라이드의 높이를 비교해본 결과, 총 4 단 중 3단까지만 사용하여도 충분하며 무게를 경감할 수 있다고 판단함. 3단으로 리니어 슬라이드를 수정하여 제작함.



ii) 모터 위치의 변경 및 지지대 변경

리니어 슬라이드를 지지하고 있는 판의 크기가 를 필요가 없다고 판단. 또 한 리니어 슬라이드의 부피가 줄어들면 그만큼 많은 공간 확보가 가능. 따라서 지지대의 길이를 줄이고, 모터의 위치를 기존의 위치에서 차체에 연결할 수 있도록 바꾸었다. 이때 지지대를 자르는 방안도 고려했으나, 현실적으로 힘들다고 판단하였다.

2) 리니어 슬라이드 부착 위치

경기장 내 허들의 높이가 35cm로, 이동의 편의를 위해 이 높이보다 아래로 로봇 차체를 제작해야함. 그러나 리니어 슬라이드의 길이는 고정되어 있으므로 이를 맞추기 위해서는 리니어 슬라이드를 최대한 차체의 밑부분에 달아야 함.

[35cm 높이를 포기할 경우]

장점 :: 리니어 슬라이드를 양옆의 프레임 위에 달 수 있어, 가운데의 넓은 공간을 활용해 두 개의 픽셀을 가로로 한 번에 잡는 집게를 부착 가능함.

단점 :: 허들보다 로봇의 높이가 더 높아, 허들 밑으로 이동하는 것이 불가능함. 그렇기에 장애물 옆 지지대 부분으로 진입하여 반대편 까지 이동하여 통과, 즉 지그재그 형태로 장애물을 통과해야 됨. 장애물을 통과하는 과정에서 조종 시 오차가 누적될 경우 충돌할 가능성이 높아지며 그에 따른 큰 불편과 제약이 있을 수 있음.

[35cm를 준수할 경우]

장점 :: 이동이 매우 간편해지고 빠른 이동이 가능함.

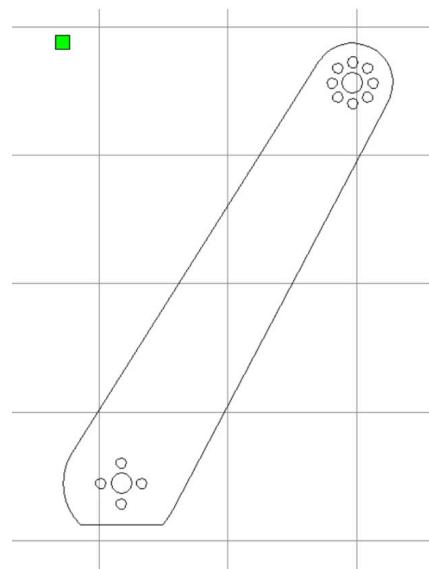
단점 :: 리니어 슬라이드 사이에 가로로 두 개의 픽셀을 집을 수 있는 집게가 들어갈 공간이 나오지 않음. 이 경우 가로로 두 개의 픽셀을 집는 집게를 사용하려면 집게의 팔을 최소 리니어 슬라이드 1단의 길이의 절반 이상 정도로 매우 길게 제작해야 함. → 안전성이 떨어짐

[결론] → 35cm 이하

이에 따라서 리니어의 하단부에서 모터와 모터의 지지대를 제거한 뒤 리니어와 연결된 판을 제거해 주었다. 남은 리니어 부분은 ㄷ자 프레임을 사용하여 연결하여 주었다. 이때 ㄷ자 프레임과 리니어 파트의 구멍 크기와 볼트의 직경이 맞지 않아 ㄷ자 프레임의 구멍을 나사산이 날카로운 볼트를 활용하여 넓혀 준 뒤 연결해 주었다. ㄷ자 기둥의 반대편과 본체의 옆 부분을 비스듬이 연결하여 기울기가 있는 판에 닿는 것이 용이하게 하였다. 이 때 1단 리니어가 내려오면서 빠지는 문제를 해결하기 위해 ㄷ자 기둥의 밑 부분의 ㄱ자 브래킷을 부착하여 주었으나 외부의 모터와의 연결을 용이하게 하기 위한 베어링 부착을 위해 작은 평판을 사용하여 공간을 만들고 동시에 리니어가 빠지는 문제까지 해결해 주었다.

3) 리니어 슬라이드 안정성 강화

리니어 슬라이드를 본체에 눕혀 부착하는데, 이를 본체의 프레임으로만 고정할 경우 안정성이 매우 떨어짐을 확인하였다. 그렇기에 리니어 슬라이드를 안전하게 고정할 수 있도록 아크릴로 제작하여 L자 브라켓과 함께 프레임에 부착하였다. 프레임과 아크릴이 삼각형 형태를 띠게 되며 안정성이 강화되었다.



3) 경기장 부품 제작

트러스 구조의 지지대를 박스로 제작 함. 트러스 구조에 기존의 봉을 구경에 맞는 pvc 파이프를 적절한 길이로 재 단하여 설치함.

기존의 봉과 여러 사이즈의 pvc 파이 프를 재단하고 부분적으로 가공 및 연 결하여 도어 (일방통행만 가능한 구조 물)의 기능을 구현함.



4. 차체의 앞은 어디인가

[픽셀을 뺏는 부분이다]

근거 :: 가장 어려운 부분을 수행하는 쪽이 앞이다.

가장 정밀해야 하므로 앞쪽을 픽셀 뺏는 부분으로 생각하는 것이 편하다.

[이터 부분이다]

근거 :: 우리 신체 또한 앞에 먹는 부분이 있고 뒤에 배출 부분이 있으므로 이게 당연하다.

차체의 “앞”을 어디로 정해야 할지에 대해 논의하였으며, 토론 결과 차체의 앞은 픽셀을 뺏는 부분으로 결정하였다.

1월 15일 (월)

1. 리니어 슬라이드 최종 수정 및 모터 장착

1) 리니어 슬라이드 최종 수정

i) 도르래 추가 및 위치 수정

본체의 리니어 슬라이드에서 타이밍 벨트의 방향을 바꾸어 주는 도르래가 각 단 별로 부착되어 있는데, 리니어가 빠지지 않도록 ㄱ자 브라켓을 부착하기 위해 도르래의 위치를 수정하며 그 간격이 최대가 아니게 되었다. 이로 인해 리니어 슬라이드가 최대 높이로 전개되지 않는다는 치명적인 오류를 발견하였으며, 리니어의 각 단에 부착된 도르래 이 외에도 모터에 타이밍 벨트가 잘 감기도록 도와주는 움직이지 않고 본체에 부착된 독립적인 도르래가 필요했다. 공간적 부족함을 해결 하기 위해 ㄷ자 프레임 두개를 사용하여 각각 ㄱ자 브라켓의 역할을 대신하도록 하였다. ㄷ자 2개의 프레임을 사용함으로써 독립적인 도르래 설치가 가능했으며, ㄱ자 브라켓 때문에 최대 높이로 올라갈 수 없는 문제점을 도르래의 위치를 옮김으로써 해결할 수 있었다.

또한 리니어 슬라이드를 부착한 프레임 가운데 부분에 큰 도르래를 하나 더 추가하여 모터에 타이밍 벨트를 더 맞물리게 하여 모터 힘을 더 잘 받게 하였다.

ii) 구조 보강

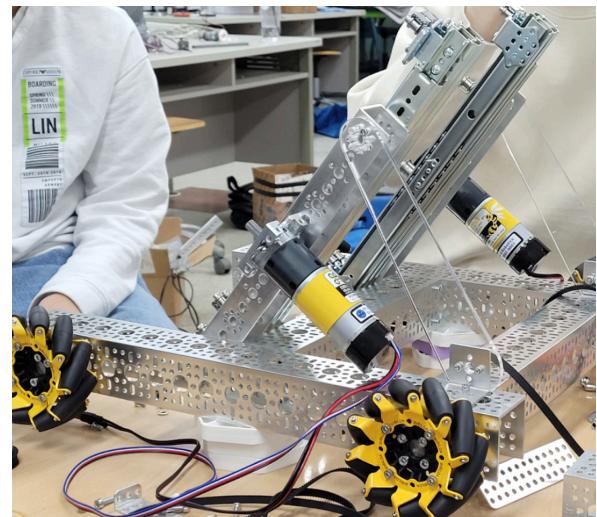
ㄷ자 프레임 두개를 사용하여 리니어가 빠지는 문제를 해결했을 뿐 아니라 독립적 도르래 설치가 가능한 공간을 생성함으로써 모터에 타이밍 벨트가 잘 걸릴 수 있도록 하였다. 하지만 근본적으로 리니어 슬라이드가 본체와



연결된 면적이 작다는 점, 구조적으로 불안정할 수 있다는 점은 해결하지 못하였다. 이에 아크릴과 레이저 커터를 활용하여 리니어와 본체를 연결할 수 있는 구조물을 제작, L자 브라켓을 활용하여 고정하였다.

2) 모터 장착 및 테스트

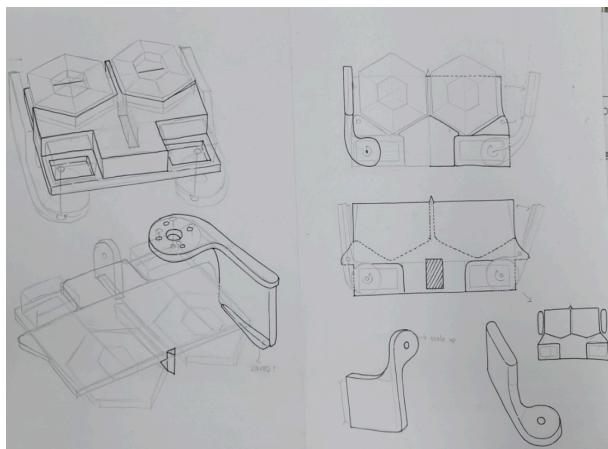
리니어 슬라이드에 사용될 모터의 위치는 컨트롤 허브나 배터리가 적재될 공간에 영향을 미치지 않으면서 타이밍 벨트, 도르래와 평행하게 연결될 수 있는 적절한 공간을 찾아 장착해야 한다. 가장 적절한 공간은 리니어 슬라이드에 평행하게 부착하는 것이라고 판단하여 모터를 장착하였다.



2. 이터 계획 및 팀 프로젝트 진행

픽셀을 로봇 내부로 넣어 바구니에 탑재하는 이터에 대한 아이디어는 크게 3가지로 나뉘었다. 각 방법에 대해 서로 장단점이 있다고 판단, 팀으로 나누어 정해진 시간동안 이터를 만들고, 가장 성공적인 이터를 로봇에 장착하기로 결정하였다.

1) 집게 이터



38cm 알루미늄 프레임을 2개 붙여 약 70cm 되는 프레임을 Servo 모터를 이용하여 돌려보았는데, 상당한 토크에도 원활하게 잘 돌아가는 것을 확인하였다. 팔이 어느정도 길더라도 집게를 간소화 시킨다면 충분히 원활하게 돌아갈 수 있겠다는 가능성을 확인하였다.

팔이 충분히 길어도 되기에, 모자이크 만들기에 더 효율적인 가로로 로딩 방법을 채택 후 3D 프린터로 픽셀 틀을 제작하며 설계 진행.

2) 로터 이터 - 컨베이어 벨트 사용

[벨트 종류의 선택]

타이밍 벨트 :: 홈의 간격이 적절하고 마찰이 좋으나, 루프 형태가 아니기에 적절한 길이의 루프로 만들 수 없음 (x)

고무밴드 (길이: 12cm, 폭: 0.5cm) :: 고무밴드가 마찰과 루프라는 면에서 적절함을 인식. 다만 기존에 가지고 있던 12cm짜리는 너무 짧아 장력이 커 알맞지 않음.

=> 길이 23cm, 폭 0.5cm 고무밴드 여러 매 구입

[구조 설계]

모터 선택 :: 서보 모터를 달 경우에도 체인으로 쓰기에 충분한 파워를 가지고 있으나, 차체의 구조와 여유있게 강한 힘을 확보하기 위해 육각 DC 모터(REV)를 선택

축 설정 :: 위 모터에 알맞는 육각 축을 선택했고, 차체에 알맞게 부착하기

위해 금속절단기로 40cm의 축 중 2.3cm를 절단

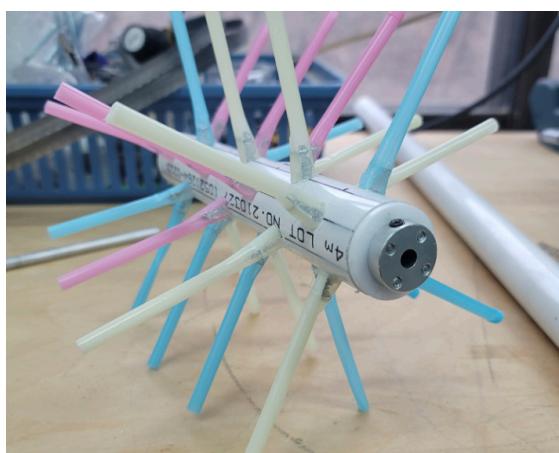
구조물 부착 :: 컨베이어 벨트를 최대한 바닥과 가깝게 하도록 달기 위해 구조물을 추가로 닮. 차체의 뒷부분 첫번째 칸에 3*5 판을 베어링과 함께 연결함. 판의 맨 밑부분에 축을 달아 고무 밴드를 연결하고자 함.

[고무줄 실험]

보유 중이던 12cm 고무밴드를 모터에 달릴 육각축과 구조물에 달린 반대 편 축에 연결 후, 픽셀이 잘 위까지 올라가는지 실험을 진행. 이 결과, 컨베이어 벨트 방식을 통해 픽셀을 운반 가능함을 확인.

3) 로터 이터 - 컨베이어 벨트 미사용

파이프에 구멍을 여럿 뚫고 못을 박은 뒤 빨대의 끝부분을 조금 잘라 못 부분과 빨대를 연결하여 이터를 제작하였다. 모터와 연결한 결과 픽셀을 잘 빨아들인다는 것을 알 수 있었다.

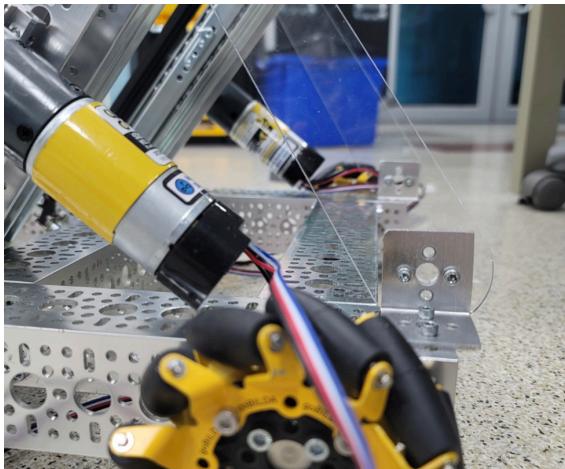


컨베이어 벨트를 사용하지 않는 만큼, 이터의 힘만으로 픽셀이 경사로를 올라가야 하기 때문에 이터의 힘을 강하게 하는 것에 중점을 두고 설계하였다.

왼쪽의 사진과 같이 직접 이터를 제작하는 등의 테스트를 하였다.

1월 16 (화)

1. 본체 뒤쪽 공간 확보



집게 이터, 로터 이터 둘 다 로봇의 뒤쪽 부분에 위치해 있기 위해서는 로봇의 뒤쪽에 많은 공간이 필요하였지만 리니어와 본체 프레임 자체의 설계에 의해 어쩔 수 없이 공간이 부족하였다. 이를 해결하기 위하여 본체의 가운데에 위치해 있는 L자 프레임과 리니어의 위치를 한 단위

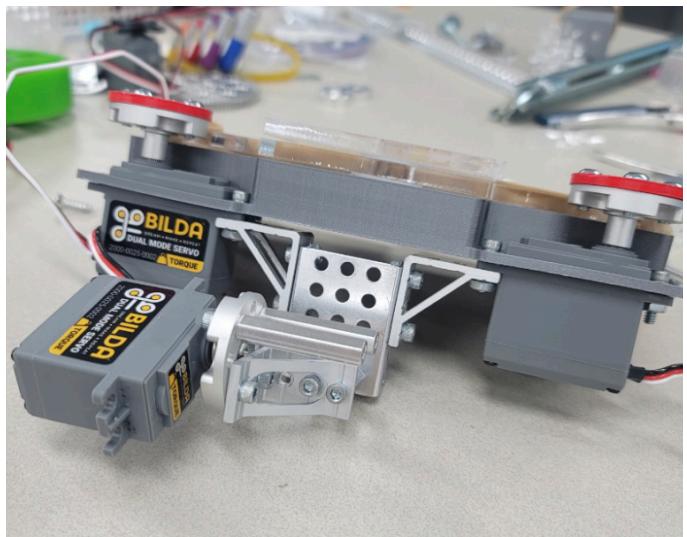
공간 만큼 앞 쪽으로 당겨 본체 뒷부분의 공간을 확보하였다.

하지만 리니어 슬라이드와 L자 프레임이 한 칸 당겨짐에 의하여 안정성 확보를 위해서 부착했던 아크릴과 L자 브래킷이 본체보다 튀어나오게 되어 위치를 다시 조정하여 아크릴을 다시 제작해야 한다.

2. 이터 팀 프로젝트 진행

1) 집게 이터 팀

3D 프린터를 이용하여 가로로 로딩할 수 있는 틀을 제작 후 아크릴로 픽셀을 안정적으로 들어갈 수 있도록 하였다. 픽셀을 하나씩 떨어뜨릴 수 있도록 양 쪽 끝에 집게를 제작하여 서보와 연결하였다. 그리고 앵글 브래킷



과 L자 프레임으로 서보를 고정하고, 손목 역할을 할 서보 모터까지 연결하였다.

그러나 생각보다 무게가 무거워서, 3D 프린트한 틀도 밀도 조정을 하고, 아크릴도 구멍을 뚫어 무게를 줄일 계획이다.

2) 로터 이터 - 컨베이어 벨트 사용 팁

[컨베이어 벨트 디자인]

한개의 육각 디씨 모터를 사용하여 축의 한쪽 부분에 모터를 설치하고 나머지 한 쪽 부분에는 축을 베어링을 사용하여 고정하고자 하였다. 이때 동력이 전달되는 축은 리니어와 본체가 예각을 이루고 있는 협소한 공간에 위치시켜야 하기 때문에 모터와 베어링의 위치를 고정하기 위해서는 3D 프린터를 사용하여 자체제작을 해야 했다. 이를 위해 모터와 베어링의 고정대 프린트하고 드릴을 사용해 본체와 연결시킬 구멍을 뚫고자 하였다.

[이터 로터 재료 선정]

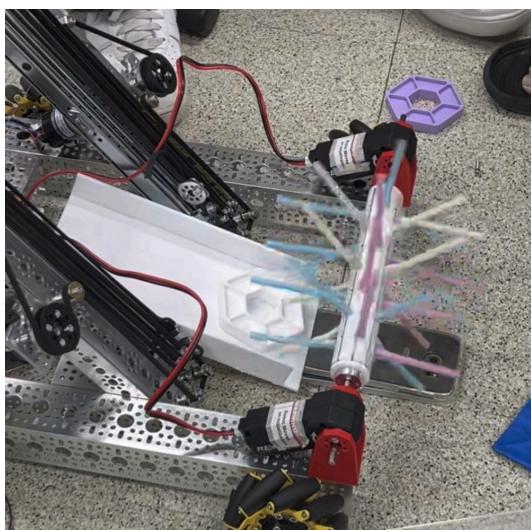
다음 재료를 로터로 사용하게 되면 로터의 축을 본체에 바로 연결하였을 때 적당한 높이에 위치해 있게 된다. 거기에 실리콘으로 제작되어 신축성이 좋아 이터 자체로도 굉장히 이점으로 가지고 있으며 직경 또한 크지 않아 컨베이어 벨트에 최소한으로 걸리게 될 것으로 예상 되었다. 위 로터를 만들기 위하여 디씨 모터를 본체 프레임에 수직으로 연결하고 이를 기어를 사용하여 90도 동력을 전환시켜 이터를 돌릴 수 있게 하고자 하였다. 기어의 개수가 부족하여 아크릴로 제작했다. 축의 연결은 본체 프레임과 육각 축을 연결해주는 베어링이 부족한 관계로 아크릴을 사용하여 육각축과 같은 직경의 구멍을 만들어 주었다. 디씨 모터는 마운트를 사용하여 연결하고자 하였으나 구멍의 위치가 여의치 않아 이 또한 아크릴로 만들어 연결하고자 하였다.



[고무줄 고정]

앞서 고무줄을 이용해 컨베이어 벨트를 제작했을 때 리니어쪽 축에서 고무줄이 탈선하는 일이 빈번하게 발생하여, 이를 방지하기 위해 고무줄을 양쪽에서 고정시켜주는 아크릴 판을 제작했다. 이때 이 판이 고무줄보다 위로 튀어나와 있으면 픽셀이 올라와서 미끄러질 수 있기 때문에, 판이 고무줄 두께만큼만 튀어나오도록 제작했다. 또한, 베어링이 새롭게 구입한 고무줄(폭 1cm, 길이 23cm)의 폭에 맞도록 하기 위해 기존의 베어링과 같은 크기의 아크릴 링을 만들어 베어링에 고정시켰다. 이렇게 제작한 것들을 활용하여 완성한 축의 모습은 상단의 우측 사진과 같다.

3) 로터 이터 - 컨베이어 벨트 미사용 팁



경사로 끝면에 픽셀이 계속 퉁겨나가는 현상이 발생하였다. 이를 보완하고자 쓰레받기 끝 부분, 자를 경사로 끝에 붙여서 픽셀이 올라가기 쉽게끔 수정하였다. 그러나 경사로가 지면과 정확하게 붙어있는 경우가 아니라면 픽셀이 경사로를 원활히 움직이지 못하는 것을 확인하였고, 경사로의 높이를 조정할 수 있는 방법이 필요하다.

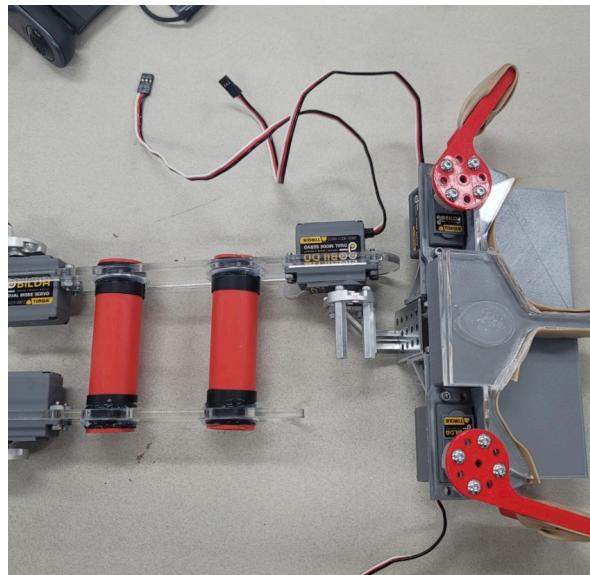
1월 17일 (수)

1. 이터 팀 프로젝트 진행

1) 집게 이터 팀

i) 도르래 추가 및 위치 수정

집게에 아크릴 두 개를 파이프 두 개로 연결하여 팔을 제작하였다. 또한 팔을 움직이기 위한 서보 두 개를 양쪽에 부착하였다.

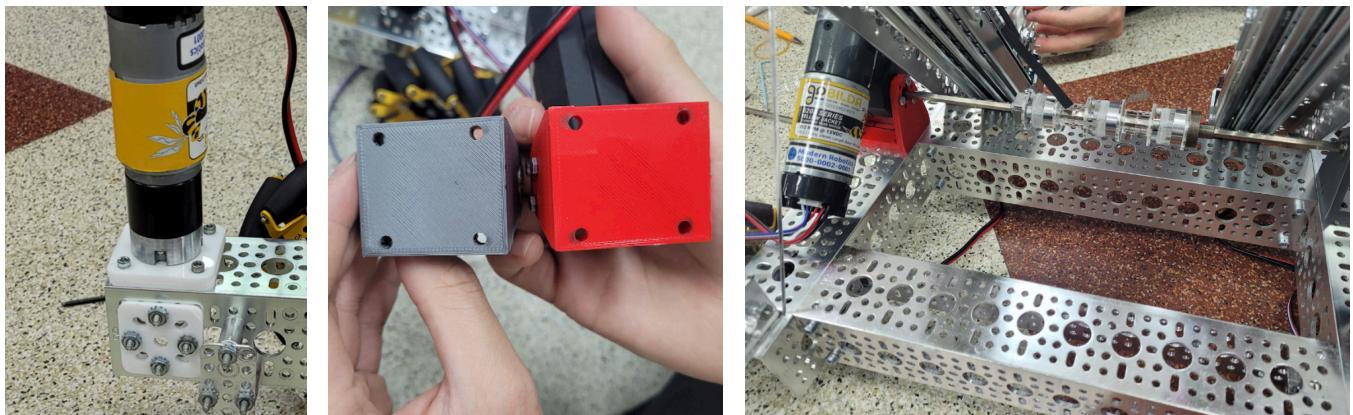


ii) 집게 이터 본체 부착

리니어 슬라이드의 가운데 위치한 엔드스탑 플레이트에 나사 두 개로 집게 팔과 연결하였다. 연결하는 과정에서 한 리니어 슬라이드 내부에 나사가 부서졌다. 나사를 빼기 어렵다고 판단하여 리니어를 4단에서 3단으로 줄임으로써 생긴 리니어 한 슬라이드와 교체하였다.



2) 로터 이터 - 컨베이어 벨트 사용 팁



[모터 고정, 고정대 제작]

아크릴로 제작한 기어, 마운트, 육각 축이 들어갈 판을 이용하여 로터를 돌릴 모터를 고정 완료했다.

컨베이어 벨트를 돌릴 축 중 리니어쪽 축을 돌릴 육각 DC 모터를 고정시키 위한 고정대를 3D 프린터로 만들었다. (1/16에 디자인한 것) 만든 후 이를 나사로 고정시키기 위해 드릴로 구멍을 뚫었다.

[바구니 디자인]

컨베이어 벨트와 함께 사용하고자 디자인한 바구니는 리니어 슬라이더에 고정되어 있는 것으로, 컨베이어 벨트를 통해 픽셀이 바구니에 수평한 방



향으로 들어가서 바구니의 안쪽 표면을 따라 기울어진 각도로 떨어진 후, 리니어에 고정되어 올라갈 때 바구니의 바닥을 서보모터를 사용하여 열면서 픽셀이 판에 떨어지도록 하는 구조였다. 어떤 모양이 가장 효율적인지 알아보기 위해 박스를 이용하여 prototype을 만들었고, 표면이 둥근 경우에 픽셀이 잘 떨어짐을 확인하여 같은 모양을 3D 프린터로 뽑았다.

[고무밴드 컨베이어 제작]

새로 주문한 고무밴드(폭 1cm, 길이 23cm)가 도착하여 그동안 만들었던 축에 직접 달아보았다. 각 베어링마다 고무밴드가 2개가 들어가서 총 6개의 고무밴드를 사용했다.

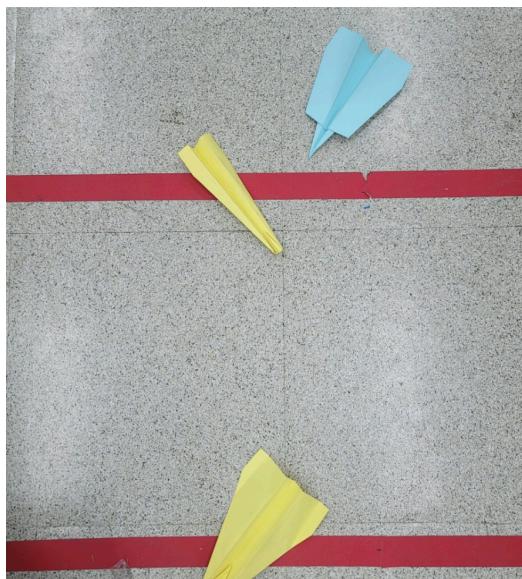
예상했던 것보다 장력이 조금 부족하여 컨베이어 벨트가 잘 작동하지 않았다. 우선 마찰을 늘리기 위해 고무밴드의 표면에 있는 하얀색 분을 씻어냈다. 이후 시도해보니 전보다는 나아졌지만, 그래도 장력은 여전히 부족했다. 이를 해결할 수 있는 선택지는 2가지가 있었다. 고무줄을 길이가 더 짧은 것으로 새로 구입하거나, 축의 위치를 조정하여 장력을 늘리는 것이었다. 축의 위치는 조정하면 다른 구조물의 배치에 영향이 갈 수 있었기 때문에 고무줄을 약 20cm의 더 짧은 것으로 구매해야 했다. 우선 장력이 좀 더 크게 작용할 때 컨베이어 벨트가 잘 작동하는지를 확인하기 위해 축의 위치를 잠시 옮기고 23cm 고무밴드를 팽팽하게 하여 시도해보았다. 그럼에도 여전히 컨베이어가 매끄럽게 작동하지는 않았다.

컨베이어 벨트의 방식을 사용하려면 고무밴드를 활용하지 않고 실제로 컨베이어 벨트를 구입하는 편이 나을 것 같다고 판단했다. 하지만 물품을 지금 새로 시키면 배송을 기다리는 동안 시간이 너무 지체되기 때문에 컨베이어 벨트를 이용한 이터는 사용하지 않는 방향으로 가기로 했다.

1월 18일 (목)

1. 종이비행기 미션

1) 비행기 제작



이번 종이비행기 미션에서의 종이비행기는 경기장 바깥 바로 앞에 바로 떨어져야 가장 높은 점수를 얻을 수 있기 때문에 양력을 많이 받아 멀리 날아가면 안된다. 적합한 종이비행기를 찾기 위해 몇 개 제작해보았고, 그 중 날아가다가 아래로 꺾여 떨어지는 동선의 비행기와 날아가다가 다시 날린 쪽으로 돌아오는 종이 비행기가 고득점하기에 적합해 보였다.

2) 비행기 발사대 계획

종이 비행기 발사대 구상을 시작하였다. 우선 약 30cm 가량 되는 \square 자 프레임 끝 양쪽에 용수철을 달았다. 그 뒤 비행기를 끼워넣을 틀을 제작한 뒤 용수철의 탄성력을 통해 비행기를 날릴 계획이다. 그리고 틀 뒷면을 \wedge 자 갈고리 형태로 만들고, 서보 부분에는 Γ 자 갈고리를 달아 용수철에 걸려 있는 틀을 당긴 뒤, 틀 뒤에 달려있는 갈고리를 서보 갈고리에 걸어서 비행기를 장전하고, 서보가 움직여 갈고리가 풀리면 비행기가 발사하는 형태로 만들 계획이다.

그러나 프레임 길이가 제한되어있기에, 용수철을 최대로 당길 수 있는 길이가 제한되어 탄성력이 부족할 수 있는 상황이다.

2. 스택 이터 제작

경기장에 픽셀이 쌓여있는 픽셀 탑에서 두 개만 빼오기 위해서 일명 ‘딸깍 집게’를 만들기 시작하였다. 딸깍 집게는 용수철의 탄성력을 이용하여 빠른 속도로 집게를 움직여 픽셀 탑의 한 층의 픽셀만 가져올 계획이다. 식탁 보 빼기와 비슷한 느낌이라고 할 수 있다.

3. 집게 구조 수정



기존의 집게 구조는 팔의 한쪽 아크릴 판에만 집게가 달려있어 불안정했다. 이에 서보 모터를 하나만 사용하면서 두 아크릴 판에 모두 고정할 수 있도록 하는 방안을 탐색해야 했다. 먼저 기존의 L자 브라켓으로 서보 모터와 연결되어 있던 부분을 U자 브라켓으로 바꾸고, 모터가 달리지 않은 부분도 잘 회전할 수 있도록 서보 혼을 달고 베어링으로 고정하였다. 그리고 서보 혼을 고정하기 위한 볼트를 박기 위해 한쪽 아크릴 판을 다시 뽑아야 했으며, 아래와 같이 디자인하여 제작하였다. 또한 아크릴 판 사이를 잇는 연결부 역시 기존의 원통형에서 트러스 구조의 안정적인 연결부로 바꾸었다.

4. 이터 최종 결정

[로터 이터 - 컨베이어 벨트 미사용]

컨베이어 벨트를 사용하지 않고 로터 이터를 사용할 때는 이터를 통해 픽셀을 받은 뒤 픽셀이 경사로를 통해 바구니로 로딩되어야 하는 점인데, 경

사로 맨 앞 부분에 픽셀이 튕기는 현상이 지속적으로 발생하여 이 현상을 없애기 위하여 경사로 맨 앞부분의 재질을 자, 아크릴, 케이블 타이 등 여러 시도를 해보았지만 픽셀을 원활하게 경사로로 옮길 방법을 찾지 못했다.

그리고 경사로에 원활이 통과한다고 하더라도 로터 이터 방법을 사용할 경우 로터 이터에 픽셀이 안 들어간 경우, 경사로에서 바구니까지 못가는 경우, 바구니로 로딩이 안되는 경우 등 다양한 단계로 인해 일어날 수 있는 변수가 너무 많다고 판단하여 컨베이어 벨트를 사용하지 않는 로터 이터 방법을 선택하지 않기로 하였다.

[로터 이터 - 컨베이어 벨트 사용]

이 또한 고무줄 장력과 축 휘어짐 현상으로 인해 여러 문제점이 발생하였기 때문에 이 방법도 선택하지 않기로 하였다.

[집게 이터]

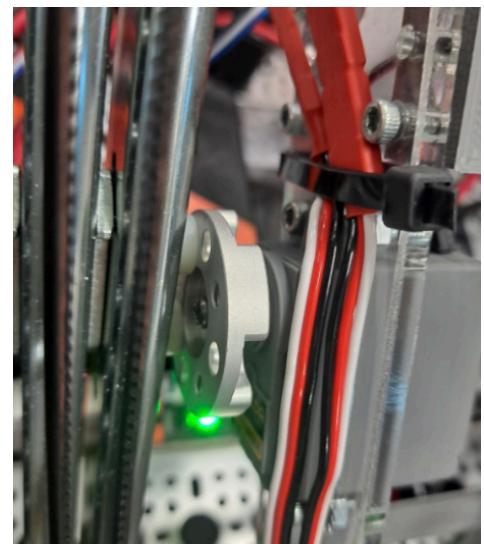
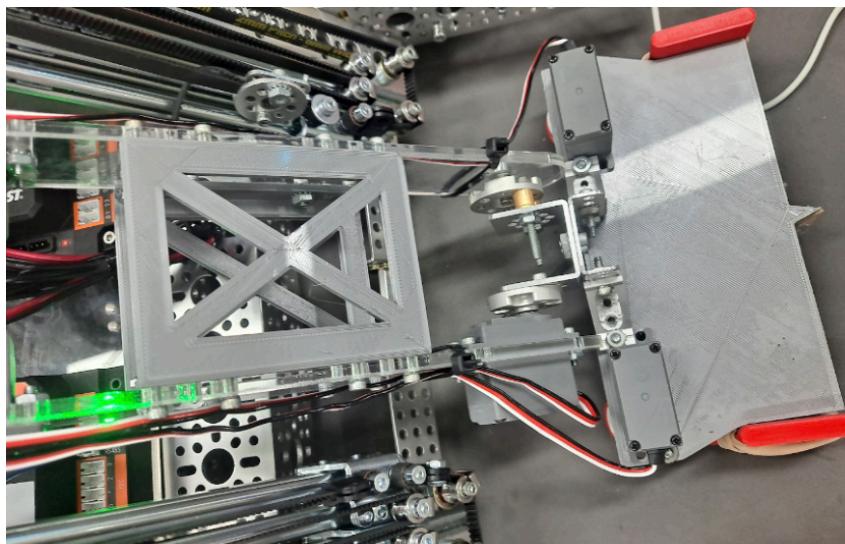
반면에 집게 이터의 경우에는 픽셀을 잡고 옮기는 것까지 수행이 되었기 때문에 다른 팀 프로젝트보다 가능성이 많아보였고, 현재 팀 프로젝트 중 가장 완성도가 높은 이터가 되어 집게 이터를 최종적으로 선택하기로 하였다.

[리니어 슬라이드 재이동]

집게 이터를 사용할 때는 백스테이지와 가까운것 보다 무게 중심이 우선이라고 생각하여 뒤로 한 칸 옮겼던 리니어 슬라이드를 다시 앞으로 당겼고, 이로 인하여 컨트롤 허브와 배터리를 넣을 공간을 확보하였다.

1월 19일 (금)

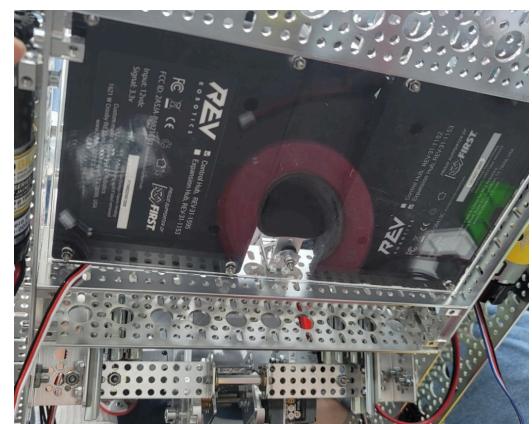
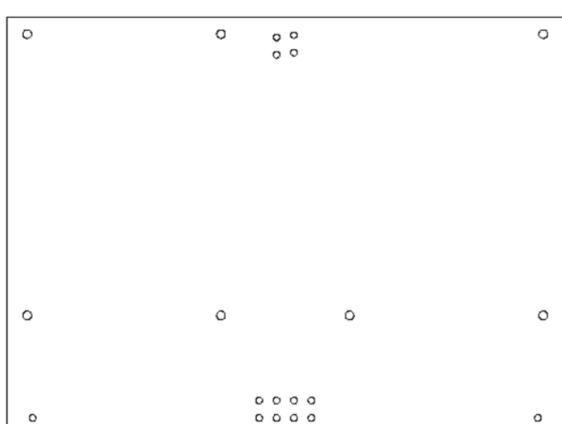
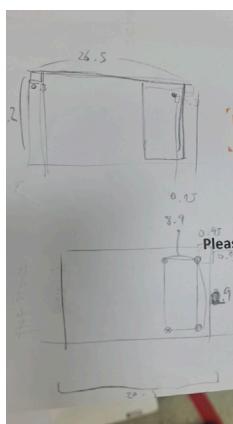
1. 집게 재조립



최종적으로 결정된 방식인 집게를 다시 본체에 달았다.

2. 허브 밟침 제작

본격적인 드라이빙 구현을 위해, 컨트롤 허브와 익스펜션 허브를 본체에 부착하고자 하였다. 이터를 사용하지 않기로 결정남에 따라 픽셀을 로딩할 박스를 위한 공간이 필요없어지면서, 본체의 앞쪽 빈 공간에 들어갈 아크릴 판을 제작하고 그 위에 허브들을 넣기로 하였다. 규격 측정 후 fusion 360으로 아크릴 판을 아래와 같이 디자인하였고, 0.5 cm 두께로 커팅하였다.



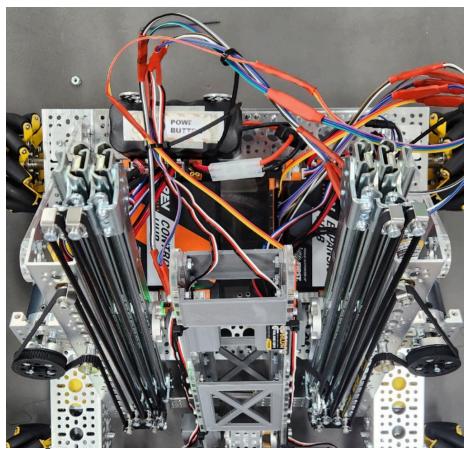
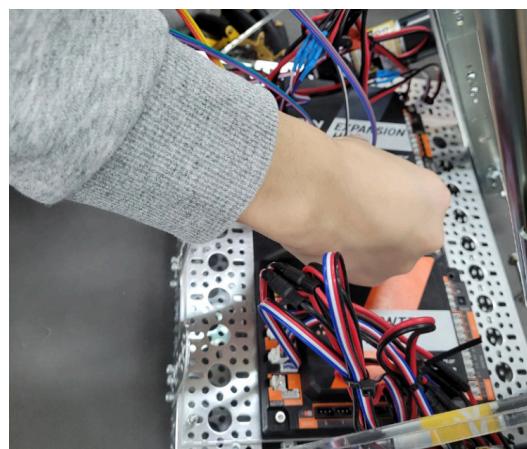
1. 본체 정비

1) 허브 위치 수정 및 모터 배치

선 정리 시도 중 기존의 위치(본체 프레임 아래)에 허브들을 배치할 경우 선 정리에 불편함이 있고, 모터의 위치 역시 허브들 사이에 배치 시 공간은 딱 들어 맞았으나 웹 캠 연결이 어려워진다는 단점이 있었다. 이에, 기존의 아크릴 판을 재활용하면서 위 문제를 해결하기 위해, 판을 본체 프레임 위에 고정하고, 모터를 차체의 맨 앞 프레임 위에 놓아 긴 볼트와 작은 판들로 움직이지 않도록 고정하였다.

2) 선 정리

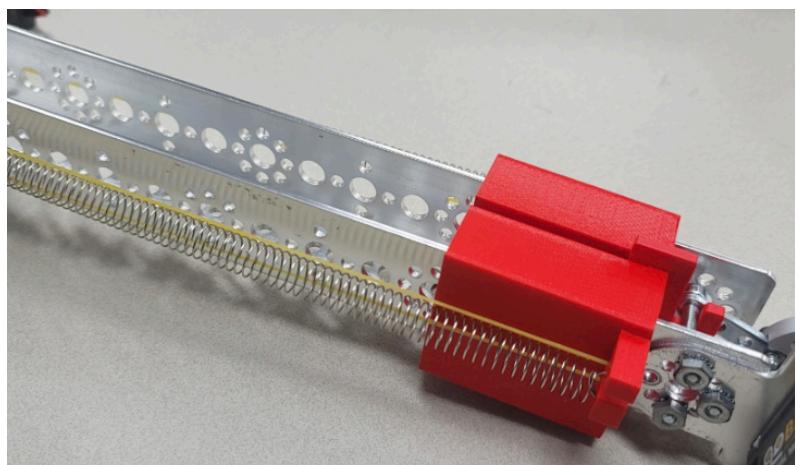
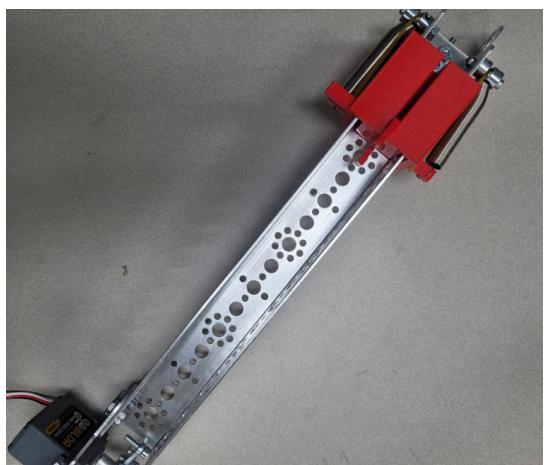
튜브로 고정이 불안정한 모터 선들을 고정하고, 선 정리를 완료했다.



2. 비행기 발사대 제작

종이비행기를 발사하기 위한 비행기 발사대를 3D프린터를 통해 제작하였다. 계획대로 틀 양쪽에 용수철을 달았지만 비행기를 날릴 만큼 장력이 강하지 않아 양쪽에 고무줄을 추가로 달았다.

발사대를 통해 비행기를 발사해보니 원하는 세기로 날아가였다. 이후 각도 조정을 하여 본체에 부착할 것이다. 그리고 서보와 연결되는 갈고리 부분은 부서질 위험이 존재하여 여분 갈고리를 미리 제작해둘 계획이다.



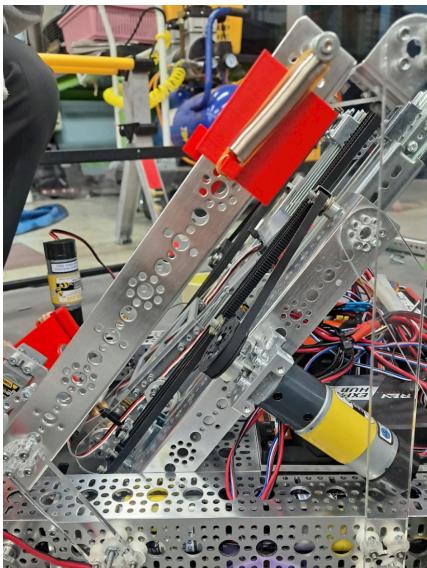
1월 21일 (일)

1. 1차 시연 성공

대회 상의 출발 위치에서 집게로 픽셀을 집고, 판 앞으로 이동한 후, 리니어를 통해 적절한 위치에서 픽셀을 떨어뜨리고 다시 복귀하는 일련의 과정을 드라이빙하는 시연에 성공했다. 이후 다양한 위치에서의 픽셀 강하, 종이비행기 날리기, 허들 아래 지나기, 허들 봉에 매달리기 등을 2차 시연 때 해보고자 한다.

연습 과정에서 집게의 움직이는 ‘손가락’ 부분이 상대적으로 낮아 픽셀이 집게 안에 쏙 들어가지 못하는 문제를 발견했다.

1. 종이비행기 받침대 부착

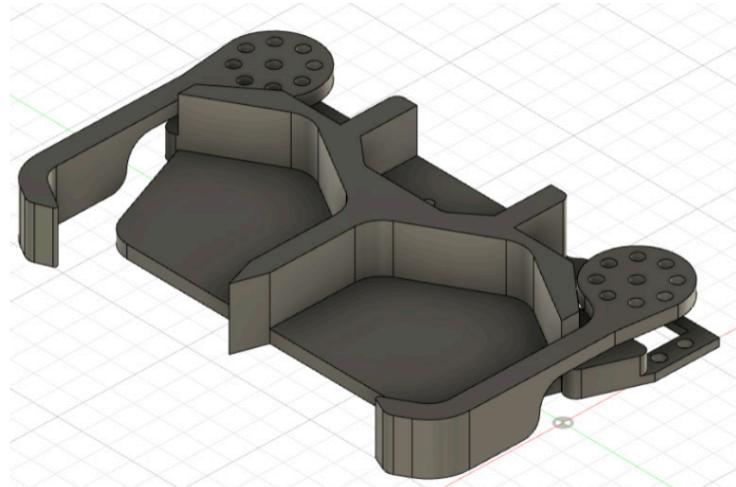


종이비행기 받침대를 본체에 부착하기 위해 서는 종이비행기가 놓여있는 3D 프린팅된 칸이 지나지 않는 곳과 본체를 연결해야 했다. 이때 볼트 구멍의 각도를 고려하여 아크릴 판을 디자인하였으며, 두 개의 서로 다른 길이의 아크릴 판을 뽑아 본체와 연결하였다. 이때 서보 모터와 연결된 갈고리가 3D 프린팅된 칸까지 도달하지 못한다는 사실이 발견되었으며, 아크릴 판 디자인의 일부를 절단하는 방안과 모터의 위치를 옮겨다는 방안을 떠올렸다. 이중 서보 모터 위치 변경을 택하여, 종이비행기를 성공적으로 날릴 수 있게 되었다.

2. 집게 디자인 변경

1) 집게 판 변경

집게 손가락으로 집게를 잡는 과정에서 집게 판이 걸려서 손가락이 제대로 픽셀을 잡지 못했다. 그렇기에 집게 판 양쪽 끝을 경사지게 만들어 집게가 픽셀을 직접적으로 잡도록 변경하였다.



2) 집게 손가락 변경

집게 손가락이 픽셀을 잘 잡지 못하여 집게 손가락의 길이를 더 늘리고 잘 잡기 쉽게 끝부분을 120도 꺾인 형태로 제작하였다.

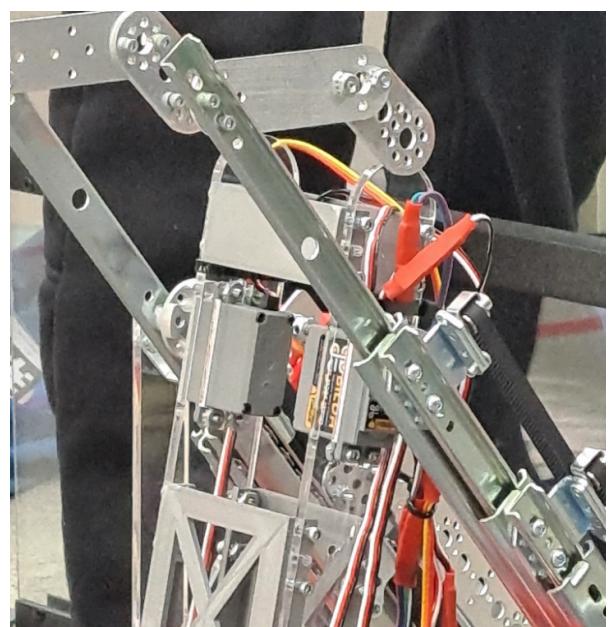
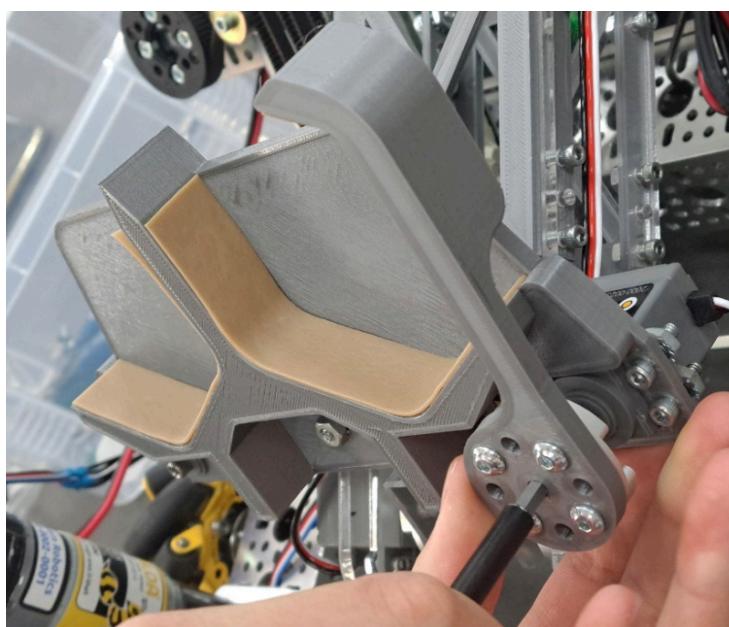
1월 23일 (화)

1. 집게 재부착

어제 바꾼 설계의 집게에 대한 3D 프린팅이 완료되어, 집게를 바꾸어 달았다. 처음에는 바꾼 집게의 손가락과 모터를 연결하는 서보 혼으로써 나사 연결 부분이 긴 부품을 사용하였는데, 이때 집게 손에 비해 손가락이 너무 아래로 내려가 테스트 시 픽셀을 제대로 잡지 못하였다. 이에 설계 담당 빌더가 원래 설계 시 나사 연결 부분이 짧은 서보 혼 사용을 고려했음을 확인했고, 이후 짧은 부품으로 교체하여 손과 손가락의 높이가 정확히 맞음을 확인하였다.

2. 갈고리 부착 (리니어 슬라이드)

엔드게임 때 허들에 매달리는 미션에서 리니어 슬라이드의 힘만으로 로봇의 무게를 감당 가능하다고 판단하여, 매달리기를 위한 갈고리를 설치했다. 얇은 판 두 개를 나사로 고정하는 단순한 방식으로 실행하였으며, 아래와 같이 완성 및 로봇이 성공적으로 매달려짐을 확인했다.



#5073

TEAM TALOS

ENGINEERING NOTE

] PROGRAMMER

2023-24 KOREA ROBOT CHAMPIONSHIP
KOREA SCIENCE ACADEMY OF KAIST

1. 사전 준비

우리 팀은 프로그래밍의 효율성을 증대시키고, 좀 더 높은 완성도의 로봇 컨트롤러를 제작하기 위하여 JAVA를 사용하기로 결정하였다. Hardware Client를 활용하여 로봇의 Control Hub에 직접 소스코드를 삽입하고 빌드하는 On Bot JAVA 개발 환경을 파악하고, 이와 관련한 공식 매뉴얼 및 자체 SDK의 메소드와 패키지를 확인하였다.

FTC에서 제공하는 SDK와 Project를 Pull하고, 공동 작업 및 후대 코드 계승을 편리하게 하기 위하여 이번 대회를 위한 로봇 소스코드를 저장할 Repository를 생성하였다.

2. 하드웨어 테스트

작년 대회 준비 당시 DC 모터의 인코더가 제대로 작동하지 않아 오토 코드를 작성할 때 정확한 위치로 움직이는 것이 힘들었다. 이 점을 보완하기 위하여 미리 제작된 구동부를 가지고 인코더 테스트를 수행하였다. 특별한 힘이 가해지지 않을 때에는 인코더를 활용하였을 때 같은 각속도로 모터가 회전함을 확인하였으며, 인코더 값도 정확하게 작용하여, 정확한 순간에 모든 모터가 동시에 정지하였다. 또한, SDK Document를 정독하는 과정에서 DC 모터를 더욱 효과적으로 활용할 수 있는 몇가지 함수를 찾아내었고, 이를 테스트 해보았다.

1) IMU (관성 측정 장치)

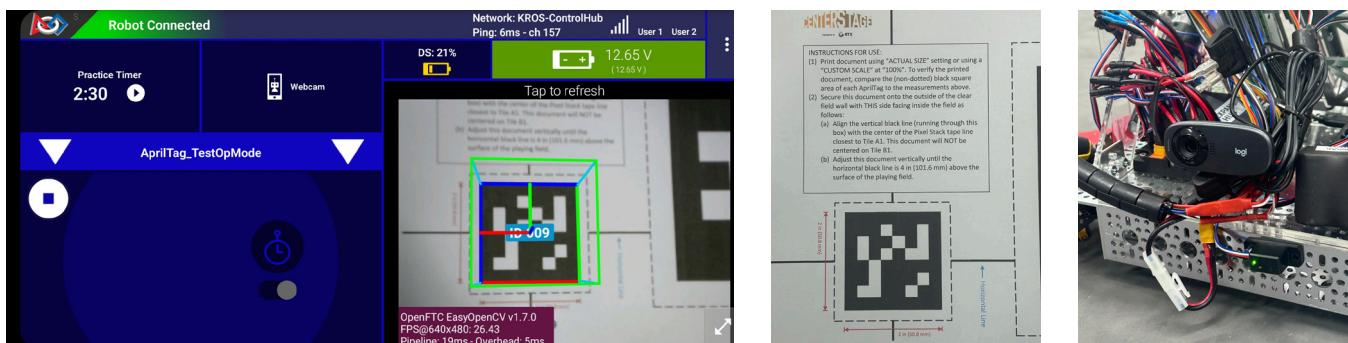
IMU (Inertial Measurement Unit)는 가속도 센서와 각속도 센서로 구성되며, Control Hub와 Expansion Hub에 내장되어 있다. 작년에는 IMU의 존재를 뒤늦게 알아서 실제 대회에서 적절히 활용하지 못하였다. 따라서 이번에는 초기에 미리 IMU를 테스트하고, 정확도를 확인하였

다. IMU 센서는 가속도를 이중 적분 하여 움직인 거리와 각도를 계산하기에 오차가 발생하기 쉽고, 실제로 테스트를 해본 결과 오차가 누적됨을 확인하였다. 흔들림이 존재하는 로봇 위에 탑재하면 더욱 오차율이 증가할 것이고, 이 문제를 해결하기 위하여 몇 가지 보정법을 고민하였다.

2) AprilTag

AprilTag는 미시건대학교에서 개발한 인식 태그로, 정확하면서도 효율적으로 인식할 수 있도록 설계되었다. FTC에서는 로봇의 현재 위치와 방향 등을 파악할 수 있도록 AprilTag를 활용한다.

경기장의 Backdrop에 부착되어 있는 AprilTag를 인식해 로봇의 현재 위치를 측정하고자 하여, 웹캠을 이용해 AprilTag가 정상적으로 인식되는지 테스트하였다. Autonomous Period에서 Team Prop의 위치를 인식한 후, 백드롭에서 이에 해당하는 위치에 픽셀을 떨어뜨려야 하므로 각 위치에 부착되어 있는 AprilTag를 인식하여 위치를 측정하면 정확도를 높일 수 있을 것이라고 예상하였다. 또한, 태그의 위치와 각도 등을 계산하여 로봇의 현재 위치를 보정할 수 있을 것이라고 기대하였다.

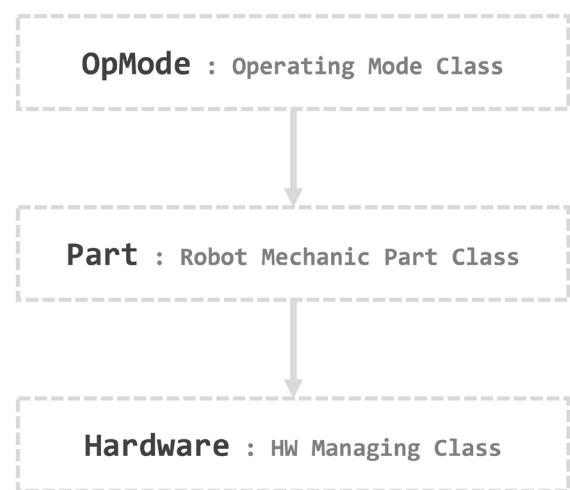


내장되어 있는 라이브러리를 사용하여 여러 종류의 태그의 ID와 이름을 불러오는 데 성공하였으며, 웹캠을 실제 로봇에 장착하여 테스트해보았다. 그러나 태그의 현재 위치와 각도 등을 계산하는 부분에서 어려움을 겪었으며, 백드롭의 위치를 측정하는 것도 광학 거리 센서를 이용했을 때의 결과와 큰 차이가 없어 태그는 사용하지 않는 것으로 결론지었다.

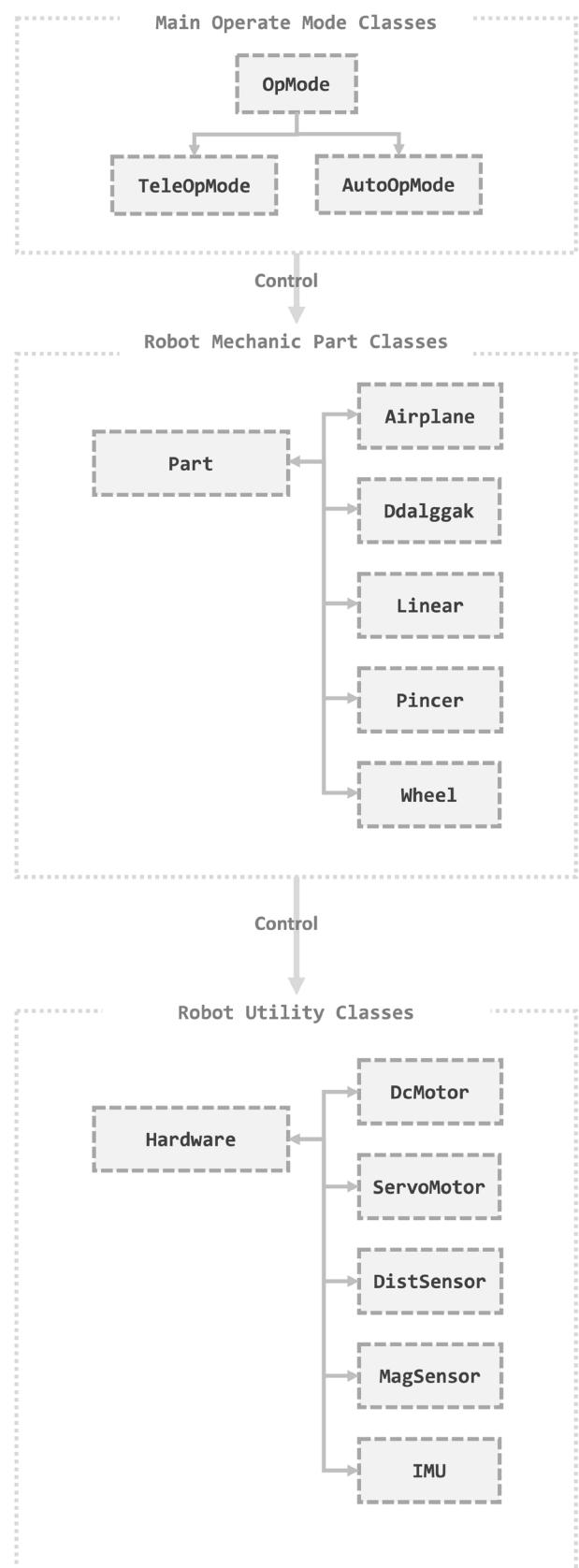
KROS 2024

Robot Code Class Diagram

Programming Team : 유태우 & 김준성



절차에 따라 상위 클래스가
하위 클래스에 동작 명령을 전달

**Github**

https://github.com/KSA-KROS/FTC2023-2024_RobotSourceCode
Copyright by 2023 KROS of KSA

3. 코드 구조 기획

코드를 효율적으로 작성하고, 유지 보수 및 가독성을 높이기 위하여 코드 구조를 기획하였다. 우리는 프로그램을 그 역할에 따라 Hardware, Part, OpMode로 세분화하여 객체 구조를 디자인 하였다.

Hardware는 FTC SDK를 직접적으로 활용하여 하드웨어 입출력 처리하는 역할을 수행한다. Part는 로봇을 기능에 따라 부분적으로 나누어놓은 객체로, 그 부분의 움직임을 관리한다. Part는 Hardware를 멤버 변수로 가지며, 자신의 Part에 해당하는 Hardware에 명령을 내리고, 관리하는 역할을 수행한다. 마지막으로 OpMode는 FTC SDK에서 제공하는 OpMode의 자식 클래스로, Part에 사용자의 입력 혹은 오토 수행 절차에 따라 명령을 내린다. Part는 OpMode의 명령을 받으며, OpMode에서는 주기적으로 Part를 업데이트하여 명령을 이행하도록 한다.

이를 통해 OpMode에서 하드웨어를 직접적으로 건드리지 않고, 추상화된 명령을 내려 움직임을 수행하도록 함으로써, 코드의 가독성을 높이고, 복잡성을 낮추었다. 또한, 명령을 내리면 곧바로 수행하는 것이 아니라, 특정 행동을 예약해두고, 업데이트를 통해 이를 이행함으로써 여러가지의 동작을 동시에 수행할 수 있도록 하였다.

4. 하드웨어 객체 개발

1) 하드웨어 추상 클래스 및 하드웨어 객체의 구동 방식 개요

Hardware는 FTC SDK를 통해 하드웨어의 입출력을 직접적으로 처리하는 객체들의 집합이다. 이 객체들은 모두 Hardware라는 이름의 추상 클래스를 상속받는다.

Hardware라는 공통된 부모 클래스를 상속받게 함으로써, 객체지향형 프로그래밍의 주요한 특성인 다형성이 가능하게 하였고, update()나 isFinished()와 같이, 필수적으로 Hardware 클래스들이 가져야 할 함수들을 추상 함수로 선언하여 오버라이딩 하게 함으로써, 프로그래머의 실수를 객체지향적 논리에 기반하여 방지하였다.

2) DcMotorHW

DcMotorHW는 DC 모터를 제어하는 클래스이다. setDirection, setUsingEncoder, setUsingBrake, setUsingFixation 등 DC 모터의 설정값을 함수를 통해 쉽게 변경할 수 있도록 하였으며, 이 함수들로 하여금 자기 자신을 반환하도록 하여, 설정 함수들을 연달아 사용 가능하도록 하였다. 이를 통해 프로그래밍의 편리성 및 가독성을 향상시켰다.

또한, 모터를 제어하는 함수인 move 함수를 오버로딩하여 멈춤 신호 없이는 특정 파워로 계속 움직이도록 하는 함수와 인코더 틱값을 입력하여 특정 틱 값에 도달할 때 까지 움직이도록 하는 함수를 제작하였다. 또한, 인코더 값의 변화량을 측정하여 모터가 움직이지 않을 때 까지 모터를 움직이게 하는 함수를 제작하여, 물리적으로 DC 모터의 범위를 정확하게 제어하면서도, 모터에 무리가 가지 않도록 하였다.

이 클래스의 또 다른 주요한 점은, DC 모터에 고정 기능을 구현하였다는 점이다. 일반적으로 DC 모터는 특정 위치로 고정되는 기능이 존재하지 않기에, 외력에 의해 위치가 변하는 경우가 존재한다. 이를 방지하기 위하여 Fixation 알고리즘을 구상하여 고정 기능을 구현하였다. 목표 틱 값에서 특정 범위 이상 벗어나면, 벗어난 양에 비례한 파워로 역방향으로 회전하여 목표 틱 값에 도달하게 하는 알고리즘이다. 강한 외력도 버틸 수 있도록, 현재 모터의 파워값과 계산된 파워값을 병합하여 역방향 회전을 가하였다.

3) ServoHW

ServoHW는 서보 모터를 제어하는 클래스이다. 서보 모터는 목표 각도를 설정하면, 그 위치로 움직여 고정된다는 장점을 가지고 있다. 하지만 서보 모터는 그 속도를 설정할 수 없고, 목표값만 알 뿐, 실제로 이에 도달하였는지는 알 방도가 없다. 따라서 속도를 조절하고, 실제 위치를 확인하기 위하여 moveWithInterval이라는 함수를 구현하였다.

모터가 동작할 시간과 최종 목표 각도를 입력하면, ServoHW가 업데이트 될 때마다 목표 각도를 변경하여, 특정 시간 동안 목표 각도가 서서히 최종 목표 각도에 도달하도록 하였다. 이 기능은 서보 모터가 무거운 물체를 들어올릴 때 유용하며, 더 정밀하고, 높은 확률로 무거운 물체의 회전을 성공하게 해준다는 이점을 가진다.

4) DistSensorHW

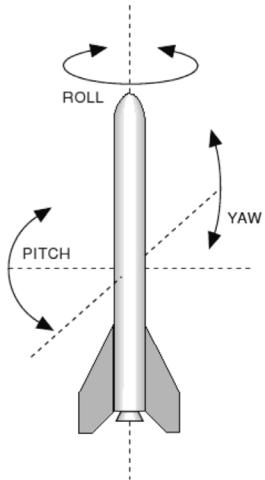
DistSensorHW는 광학 거리 센서를 제어한다. 센서는 REV Robotics 2m Distance Sensor를 사용하였으며, 이는 최대 2m까지의 거리를 탐지할 수 있다. 광학 센서는 차체의 정면과 우측에 장착되어 있으며, 아래의 함수들을 이용해 Team Prop과 Backdrop을 탐지하는 역할을 한다. isObjectDetected 함수는 10cm 혹은 주어진 거리 이내에 물체가 감지되었는지를 true / false로 리턴한다.

5) MagSensorHW

MagSensorHW는 자기장 센서를 제어한다. Magnetic Limit Switch를 사용하였으며, 이는 충분히 강한 자기장이 감지되었을 때 스위치가 켜진다. 그러나 자기장의 세기에 따라 입력값이 달라지지 않고, 충분히 강한 자기장의 존재 여부만을 확인할 수 있다. DC 모터의 인코더 값으

로 얼마나 회전했는지를 정확하게 확인할 수 없어, 마그네틱 센서를 이용해 모터가 충분히 회전했는지를 확인할 수 있도록 리니어 슬라이드에 적용하였다. `isActivated` 함수는 마그네틱 스위치 센서가 trigger되었는지 감지해 `true` / `false`로 리턴한다.

6) IMUHW



IMU는 내장된 `getAngularVelocity`라는 함수를 통해 Yaw, Pitch, Roll 방향의 각속도를 `zRotationRate`, `xRotationRate`, `yRotationRate`로 불러올 수 있다. 정지시켜 상태에서도 0이 아닌 각속도가 측정되었으며, 이를 해결하기 위해 측정된 각속도가 -2.0 초과 2.0 미만이라면 0으로 보정하여 출력하는 `getVelocity` 함수를 정의했다.

또한, 누적되는 오차를 보정하기 위해 위에서 새로 정의한 `getVelocity` 함수를 이용해 현재 향하고 있는 각도를 측정할 수 있도록 `getAngle` 함수를 정의했다. OpMode에서 코드가 한 번의 loop를 돌 때마다 실행되는 데 걸린 시간 (Δt , ms)을 측정해

$$(\text{현재 각도}) = (\text{현재 각도}) + (\text{각속도}) \times \Delta t$$

로 계산하였으며, 현재 각도가 [-180, 180]의 범위를 벗어난 경우 ± 360 으로 설정해주었다.

테스트 결과, 실제 각도를 측정한 값과 소수점 첫째 자리 이하에서만 오차가 있었으며 매우 천천히 움직일 경우 2.0 이하의 값이 0으로 치환되어 오차가 커지기도 하였다. 로봇의 이동 / 회전 속도가 빠른 편이기에 실제 오차는 거의 없는 것으로 확인되었다.

처음에는 Control Hub의 IMU만을 사용하여 계산하였으나, 이후 Ex-

pansion Hub의 IMU를 imu_sub로 불러와 각각 0.5만큼씩의 가중치를 두어 평균값으로 계산하였다. 그 결과, AutoOpMode에서 보다 정확하게 방향이 측정되었으며 안정적으로 미션을 수행할 수 있었다.

5. 파트 객체 개발

1) 파트 추상 클래스 및 파트 객체의 구동 방식 개요

Part는 Hardware를 제어하고, OpMode로 부터 받은 명령을 수행하는 클래스의 집합이다. 로봇을 기능별로 나누어 각각을 제어하는 역할을 담당한다. 이러한 클래스들은 모두 Part라는 이름의 추상 클래스를 상속 받는다.

Part를 추상 클래스를 상속받게 한 이유는 위 Hardware와는 약간 다르다. Hardware는 다양성을 위한 반면, Part 추상 클래스는 캡슐화와 추상화에 초점이 맞추어져 있다.

우선 Part의 작동 방식을 이해할 필요가 있다. Part는 startStep이라는 함수를 통해서 OpMode로부터 명령을 할당받는다. startStep은 nextStep이라는 추상 함수를 호출하고, 각각의 Part는 이 nextStep을 오버라이딩 하여, 각 명령에 대해 단계적으로 어떤 처리를 할 지 입력할 수 있다. Part에는 step이라는 변수가 존재하는데, 이 변수는 1부터 시작하여, 특정 명령의 수행 단계를 의미한다. nextStep에서는 Hardware 객체들에 명령을 할당하는데, 각각의 Hardware들이 명령을 할당받아 수행을 모두 완료하면, step이 1씩 자동으로 증가하고, nextStep 함수가 호출되어, 특정 명령의 다음 단계가 순차적으로 수행된다. Hardware들을 업데이트하고, 끝났는지 확인하여 step을 자동으로 증가시키는 함수들은 Part 추상 클래스에서 미리 정의되어있기에, 각각의 Part는 그들만의 생성자를 정의하여, Part별 Hardware들을 가져오고, nextStep

함수를 정의하여 명령에 따른 단계적 명령 수행 절차만 switch-case문으로 작성하면 끝난다. 이러한 객체 추상화를 통하여 코드 작성의 효율을 극대화하였으며, 가독성을 증가시키고 동일 기능을 부모 클래스에서 정의 함으로써 유지 보수가 쉬워지도록 하였다.

Part가 할당받는 명령은 Command라는 이름의 열거형 타입으로 정의 하였으며, JAVA의 Interface를 활용하여 열거형 타입에 객체지향적 다양성을 부여하여, 명령 처리 함수를 부모 클래스인 Part 추상 클래스에서 정의할 수 있도록 하였다.

2) Hardware Manager

Part는 하드웨어들을 단체로 관리한다. 하지만 각각의 Part가 어떤 하드웨어를 사용하는지 알 수 없고, 모두 다를 것이기에, Part의 하드웨어를 모두 업데이트하거나, 끝났는지 여부를 확인하기 위해서는 불필요한 단순 작업을 통해 각각 파트별로 업데이트 및 종료 확인 기능을 제작해야한다. 이는 버그를 발생시키기 쉽고, 효율성을 저해하므로, 이를 해결할 수 있는 HardwareManager 클래스를 제작하였다.

HardwareManager 클래스는 Hardware 클래스들이 Hardware 추상 클래스를 상속받기에, 다양성을 가진다는 원리에 기반한다. 업데이트나 종료 확인 기능은 모든 Hardware 클래스의 공통 기능이고, Hardware 추상 클래스에 정의되어 있기에, 객체의 종류가 다를지라도, 다양성을 활용하여 모두 Hardware라는 이름의 객체 리스트에 저장하여 관리 할 수 있다. HardwareManager는 registerHardware 함수를 통해 특정 파트의 모든 hardware 클래스를 받아와 리스트에 저장한다. 다양성을 활용하면 반복문을 통해 쉽게 모든 hardware 클래스를 일괄적으로 업데이트하고, 종료 여부를 확인할 수 있는 이점을 가진다.

01 HardwareManager 클래스는 Part 추상 클래스가 가지고 있는 객체로, 모든 파트들이 이 객체를 가지고 있으며, 이 객체를 통해 하드웨어에 일괄적인 처리를 간편하게 수행한다.

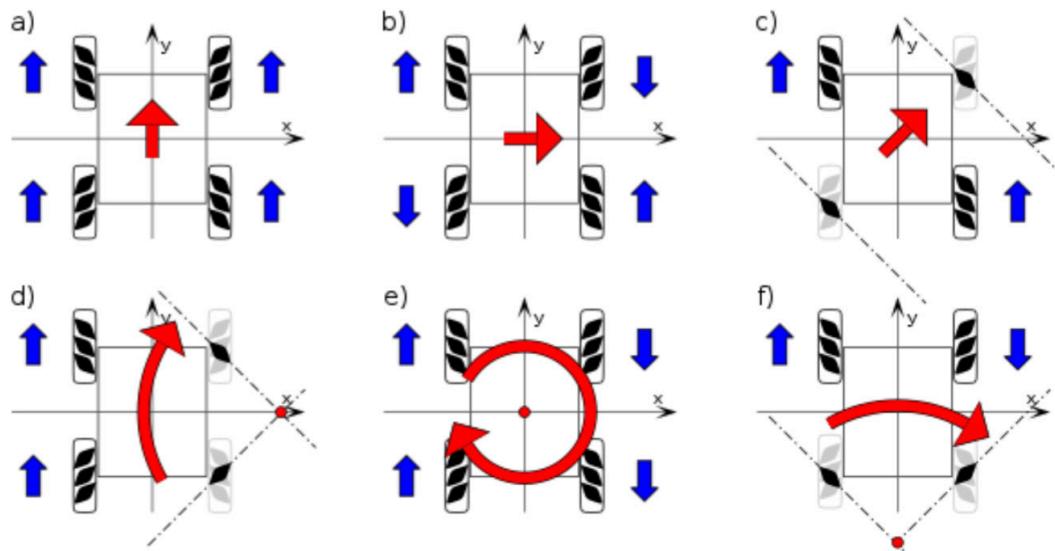
3) WheelPart

로봇의 구동부를 관리하는 클래스이다. 바퀴에 해당하는 4개의 DC 모터와, 오토 때 팀 소품 감지에 사용되는 2개의 거리 센서를 제어한다.

명령어	기능
MOVE_FORWARD MOVE_BACKWARD MOVE_RIGHT MOVE_LEFT	앞, 뒤, 오른쪽, 왼쪽으로 연속적으로 이동함.
VIEW_FORWARD VIEW_BACKWARD VIEW_RIGHT VIEW_LEFT	시작 시점의 방향을 기준으로 IMU를 활용하여 특정 방향으로 자동 회전함.
TURN_RIGHT TURN_LEFT	오른쪽, 혹은 왼쪽으로 연속적으로 회전함.
STOP	정지
AUTO_MOVE	정적 변수인 이동 거리와 방향으로 이동함. Auto Op 전용 함수.

또한, 이번 Wheel 파트의 가장 강력한 특징은 전방향 구동이다. 메카넘 휠은 4개의 바퀴의 회전 방향 및 회전 속력을 달리 함으로써 앞 뒤 뿐만 아니라, 그 외 방향으로도 구동이 가능하다. 이전에 출전한 대회에서는 앞, 뒤, 왼쪽, 오른쪽 4가지 방향 구동만 가능하였지만, 이번에는 수학적 계산을 통해 각도에 따라 바퀴 각각의 각속도를 달리하여 모든 방향으로 구동이 가능하도록 제작하였다.

삼각함수를 활용하면, 이동 방향을 가지고 이동 방향 벡터를 구할 수 있다. 여기서 벡터의 크기는 이동 속력이다. 좌측 상단과 우측 하단에 위치한 바



퀴는 이동 방향 벡터를 -45° 만큼 회전시킨 후, x축에 사영시킨 벡터의 크기에 비례하고, 우측 상단과 좌측 하단에 위치한 바퀴는 이동 방향 벡터를 $+45^\circ$ 만큼 회전시킨 후, x축에 사영시킨 벡터의 크기와 비례한다. 이를 통해 전방향 구동을 구현할 수 있다.

3) LinearPart

로봇의 리니어를 관리하는 클래스이다. 리니어를 구동시키는 2개의 DC 모터와 리니어의 높이 하한값 제한을 위한 마그네틱 스위치를 제어한다.

명령어	기능
MOVE_UP MOVE_DOWN MOVE_DOWN_POWERFUL	리니어 슬라이드를 위, 아래로 움직이며 각각 0.6, 0.4, 0.8의 속도로 이동함. (POWERFUL은 리깅 매달리기용 기능)
MOVE_DROP_POSITION MOVE_PSEUDO_DROP_POSITION MOVE_ORIGINAL_POSITION	리니어 슬라이드를 정해진 위치로 이동함.
STOP	리니어 이동을 멈춤.

리니어 슬라이드의 가장 큰 특징은 두 모터의 고정값 동기화이다. 우선 리니어 슬라이드는 중력이라는 강한 외력을 이겨낼 필요가 있다. 따라서 리니어 슬라이드를 구동시키는 DC 모터에는 고정 기능이 활성화 되어있다.

하지만, 두 모터의 고정 위치가 정확하게 일치하는 경우는 드물다. 따라서 두 모터가 엇나가고, 잘못 움직이는 경우가 관찰되었다. 이 문제를 해결하기 위하여 모터에 동기화 기능을 추가하였다.

오직 동기화한 모터만이 고정 위치를 판단하고, 위 아래 움직임을 관리한다. 동기화된 모터는 고정 위치를 판단하지 않고, 동기화한 모터의 움직임에 맞추어 같이 움직인다. 이를 통해 두 모터가 엇나가는 상황을 방지하고, 성공적으로 위치를 고정할 수 있게 하였다.

4) PincerPart

로봇의 집게 부분을 관리하는 클래스이다. 집게는 팔 (Arm), 손목 (Wrist), 손가락 (Finger)으로 구성된다. 각각은 서보 모터로 구동되며, 각 서보는 GrabPosition (픽셀을 집을 때의 위치)과 DropPosition (픽셀을 놓을 때의 위치) 값이 변수로 정해져있다.

명령어	기능
GRAB_PIXEL_RIGHT GRAB_PIXEL_LEFT DROP_PIXEL_RIGHT DROP_PIXEL_LEFT	각각 오른쪽, 왼쪽 손가락의 서보 모터를 구동해 픽셀을 놓거나 잡음.
GRAB_OR_DROP_PIXEL_RIGHT GRAB_OR_DROP_PIXEL_LEFT	현재 손가락의 위치를 고려해 각각 오른쪽, 왼쪽 손가락의 서보 모터를 구동해 픽셀을 놓거나 잡음. (현재 Grab이라면 → Drop)
MOVE_DROP_OR_GRAB_POSITION AUTO_MOVE_DROP_OR_GRAB_POSITION	현재 팔의 위치를 인식해 픽셀을 Backdrop에 놓을 수 있는 위치로 이동하거나 init 위치로 이동, 각각 TeleOp와 AutoOp에서 사용함.

팔의 서보가 회전하는 데 토크가 부족하다는 문제를 확인하였으며, 중간에 멈추는 문제를 해결하고자 구동 시간을 상수로 정하였다. TeleOp에서는 올리고 내릴 때 모두 3초, AutoOp에서는 각각 7초와 5초의 시간 동안 구동하도록 하여 정상적으로 작동함을 확인하였다.

5) Ddalggak Part

로봇의 스택 이터를 관리하는 클래스이다. 스택 이터는 두 개의 DC 모터로 구동되며, 픽셀을 집게로 밀어주는 역할을 한다. 스택 이터를 개발하던 중, 우리 팀은 이 파트를 비공식적으로 “딸깍 파트”라고 칭하였으며, 프로그래밍이 완료될 때까지 정식 명칭을 정하지 못하여 Ddalggak Part라고 코드를 작성하였다.

명령어	기능
OPEN_OR_CLOSE_DDALGGAK	현재 이터의 위치를 인식해 이터를 열거나 닫음. 파워를 0.8로 설정해 주어 강하게 움직이며, 스택에 쌓여 있는 픽셀을 가져오는 역할을 수행함.
OPEN_OR_CLOSE_DDALGGAK_GENTLY	현재 이터의 위치를 인식해 이터를 열거나 닫음. 파워를 0.2로 설정해 주어 약하게 움직이며, 휴먼 플레이어가 놓아 준 픽셀을 정확하게 집게 안으로 가져오는 역할을 수행함.
RESET_DDALGGAK	이터를 최대한 열어 상태를 초기화함.
CLOSE_PERFECTLY	이터를 최대한 닫음. (이터를 로봇 내부로 이동시켜 외력에 의한 물리적 손상을 방지하기 위한 기능)

처음에는 마그네틱 리미트 스위치를 이용해 이터의 구동 범위를 제한 및 초기화 하려고 하였으나, 센서에 의존한 구동 범위 제한은 경우에 따라 인식을 실패하는 등의 문제가 있어 활용하기 어려웠다. 따라서 우리는 물리적으로 회전할 수 있는 각도를 제한하고, 모터가 물리적으로 멈추게 되는 상황을 감지하여 멈추도록 함으로써 이 문제를 해결하였다.

6) Airplane Part

Airplane Part는 종이비행기 거치대에 고정되어 있으며, 서보 모터로 구동된다. airplane이라는 이름으로 등록된 서보 모터를 돌려 고무줄과 용수철이 연결된 거치대를 놓아주고, 이로써 종이비행기를 발사한다.

명령어	기능
FLY	서보 모터를 구동해 종이비행기를 발사함.

6. OpMode 개발

1) OpMode의 구동 방식 개요

OpMode는 로봇을 작동시키는 main 클래스로, 로봇의 구동을 총괄하는 역할을 수행한다. OpMode는 주로 Part 객체를 정의하고, Part에 명령을 할당하며, 업데이트함으로써, 명령을 수행하도록 한다. 또한, 경우에 따라 OpMode에서 바로 Hardware를 제어하는 경우도 존재한다. TeleOpMode는 컨트롤러 입력값을 통해 드라이버의 명령을 받고, 이에 따라 Part에 명령을 할당하는 클래스이다. 따라서 이 클래스는 주로 조건문을 활용하여 컨트롤러의 입력값을 확인하고, 이에 따라 특정 동작을 시행한다.

AutoOpMode는 이미 짜여진 구동 절차에 따라 Part에 명령을 할당하고, 오토 미션을 수행하는 클래스이다. 따라서 이 클래스는 Part와 비슷하게 Command와 Step을 기반으로 프로그램이 작동한다. 정해진 바에 따라 Command를 순차적으로 실행하고, 각 Command가 실행될 때마다 startStep 함수를 통해 명령을 수행한다. 파트들의 명령 수행이 끝나면 step 변수를 1씩 증가시키면서 Command에 적혀있는 명령들을 순차적으로 수행한다.

2) TeleOpMode

TeleOpMode는 두 명의 드라이버가 Driver Controlled Period 안 컨트롤러 (게임패드)를 이용해 로봇을 조종할 수 있도록 한다. 드라이버

들은 로봇의 이동과 픽셀 관련 구동을 각각 담당하게 되며, 조이스틱과 버튼 등을 이용해 로봇을 조종한다. TeleOpMode의 개발 과정과 컨트롤러 사용법은 다음과 같다.

i) 개발 과정

개발하며 가장 중요했던 부분은 위에서 언급한 “로봇의 앞은 어디인가”라는 문제였다. 게임패드의 조이스틱과 방향키를 이용하여 조종해야 하기에, 로봇의 앞이 어디로 정해지느냐에 따라 컨트롤 방법이 바뀌었다. 기존에는 로봇 차체에서 “앞”을 정의하고 이를 기준으로 컨트롤러를 매핑하였으나, 드라이버의 요청으로 드라이버가 바라보고 있는 방향을 기준으로 “앞”을 결정하도록 하였다. 이에 따라 TeleOpRight와 TeleOpLeft를 따로 작성하여 드라이버가 바라보는 방향에 따라 조종 방향 또한 바뀌도록 하였다.

로봇을 움직이는 것은 방향키를 이용하여 앞, 뒤, 오른쪽, 왼쪽으로 이동할 수 있었으나 이후 조이스틱을 이용해 보다 직관적으로 조종할 수 있도록 하였다. 조이스틱을 미는 힘에 따라 로봇의 속도가 변하며, 대각선으로 움직일 수 있도록 함으로써 이동 속도를 단축할 수 있었다.

TeleOp를 개발하며 게임패드의 버튼과 관련해 드라이버들과 많은 토의를 거쳤다. 제작한 경기장에서 연습을 거듭하며 더 편리하게 조종할 수 있도록 다양한 컨트롤 방식을 테스트해 보았으며, 이를 통해 꾸준히 점수를 높일 수 있었다고 생각한다.

ii) 컨트롤러 사용

우측의 사진과 같이 컨트롤러 사용을 위한 가이드를 제작하였다.



Controller Guidebook

DRIVER #1 : 구동부 담당



TIP

- 먼 거리를 움직일 때는 조이스틱이 유용
- 트러스를 지날 때는 조이스틱으로 약하게 조종
- 픽셀을 집거나 떨어트릴 위치로 이동할 때는 미세조종



Controller Guidebook

DRIVER #2 : 리니어, 짐개, 팔짝이 담당



TIP

- 리니어 하강은 센서로 제한되어있기에 안심할 수 있음
- 팔짝이는 열 때 자동 초기화
- 리니어 당기기에는 턱걸이 외 사용 금지
- 트러스를 기준으로 짐개는 픽셀 저장소 쪽에서만, 리니어는 백드롭 쪽에서만 조작하는 것을 권장



Controller Guidebook

EMERGENCY MODE



Emergency Mode

위험 상황 발생 시, 긴급 정지용 기능
하지만 이번 대회에서는 그다지 긴급 정지할 상황은 발생하지 않을 것이라고 생각함



3) AutoOpMode

AutoOpMode는 30초동안의 Autonomous Period동안 로봇이 자율적으로 구동하도록 작성하였다. Team Prop을 거리 센서로 인식해 알맞은 위치에 보라색 픽셀을 내려놓고, 역시 거리 센서로 Backdrop의 위치를 인식해 노란색 픽셀을 떨어뜨린 다음 집게를 접은 후 주차한다.

연습 경기에서 AutoOpMode로 45점의 점수를 안정적으로 획득할 수 있었으나, 시작하는 위치에 따라 전략을 바꾸어야 할 수 있다고 판단하였다. Backdrop과 가까운 위치에서 시작할 경우 노란색 픽셀을 놓고 주차 까지 함으로써 45점을 득점하는 것을 목표로 하고, 먼 위치에서 시작할 경우 보라색 픽셀만을 놓는 것을 목표로 하였다.

AutoOpMode에서 가장 중요하게 다룬 부분은 거리 센서와 IMU이다.

i) 거리 센서

AutoOpMode는 거리 센서의 입력값에 의존한다. 개발 초기에는 정면의 거리 센서만을 이용해 Team Prop에 있는 위치에 도달한 후, 정면 / 좌측 / 우측을 순서대로 바라보며 거리 센서에 물체가 인식되는지를 확인하였다. 그러나 이는 시간상 비효율적이기에, 우측에 센서를 추가로 장착하여 회전 없이 Prop의 위치를 확인할 수 있도록 하였다. (정면과 우측 모두 물체가 감지되지 않는다면 Prop이 좌측에 있음)

거리 센서를 이용해 Backdrop의 위치 또한 확인할 수 있다. 보라색 픽셀을 떨어뜨린 후, (Blue Team 기준) Backdrop이 보이지 않는 위치까지 왼쪽으로 이동한다. 거리 센서에 Backdrop이 감지될 때 까지 오른쪽으로 이동하며, 감지된 이후에는 Prop의 위치에 따라 노란색 픽셀을 놓을

위치로 이동한다.

ii) IMU

IMU는 AutoOpMode동안 로봇이 정확한 방향으로 나아가도록 돋는다. TeleOpMode의 경우, 로봇이 올곧게 이동하지 않을 시 드라이버가 로봇을 회전시킴으로써 방향을 보정할 수 있다. 그러나 Auto의 경우 사람이 개입할 수 없기에 정확한 방향으로 이동할 수 있도록 해야 한다.

우리는 Control Hub와 Expansion Hub의 IMU를 활용하여 로봇이 바라보고 있는 각도를 확인할 수 있도록 하였다. 몇 번의 시행착오를 거치며 두 개의 IMU를 사용하여 평균값을 계산함으로써 오차를 보정할 수 있었다.

#5073

TEAM TALOS

CONCLUSION & PLANS

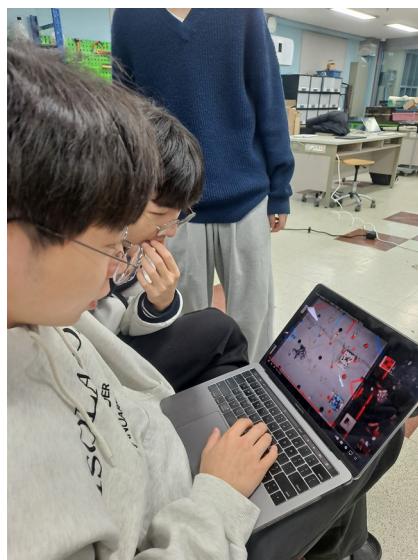
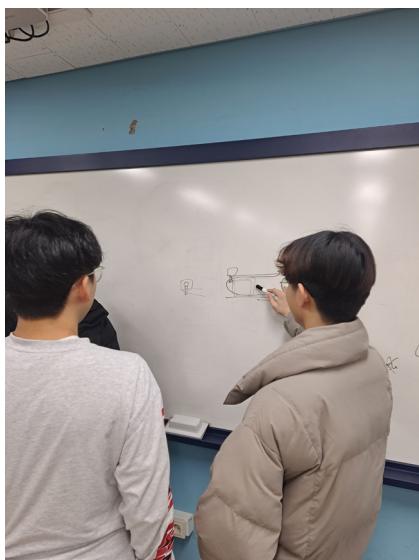
2023-24 KOREA ROBOT CHAMPIONSHIP
KOREA SCIENCE ACADEMY OF KAIST

CONCLUSION : 결론 및 제언

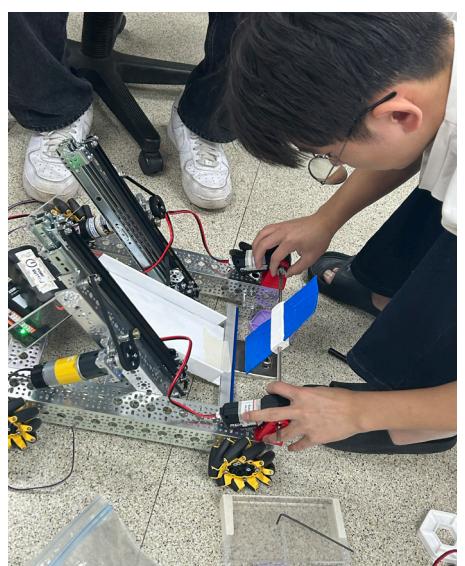
1월 12일부터 26일까지 학교에 잔류하며 최종적으로 로봇 “우이” 및 코드 작성을 완료하였으며, Autonomous Period와 Driver control Period를 충분히 테스트할 수 있었습니다. 드라이버와 휴먼 플레이어를 맡은 팀원들 또한 충분히 연습할 시간을 가지며 대회를 준비하였습니다.

처음 계획한 구상과 전략에 거의 알맞은 로봇을 만들고, 프로그래밍도 짧은 시간 내 많은 수정과 재확인을 거쳐 완성시켰다는 점에서 뿌듯함이 큽니다. 로봇 설계와 제작, 전략 구상에 있어 전 과정에 열심히 참여하고 많은 도움과 가르침을 준 TALOS의 팀원들 모두와 지도해주신 김호숙 선생님께 감사하다는 말을 전하고 싶습니다.

직접 만든 경기장에서 시범 경기를 진행하였을 때에는 높은 점수를 획득하였으나, 실전에서 오류나 실수가 없도록 하는 것이 가장 중요하다고 생각합니다. 로봇의 제작이 마무리된 지금도 로봇 조종과 자율 구동에 대한 테스트를 계속 진행하고 있으며, 마지막까지 열심히 노력해 실전 대회에서 좋은 결과를 내고자 합니다.



TALOS ENGINEERING NOTE



본 엔지니어링 노트의 경우, 각자 자신이 맡은 부분 혹은 제작한 부분에 대한 설명을 작성하고 매일 한 명이 이에 대한 총괄을 담당하여 관리하였습니다. 로봇의 바뀐 부분과 발생한 문제점, 팀원들과 논의한 사항 등을 EN에 모두 기록하였고, 추후 참가할 대회에서 참고할 수 있는 기록물을 남길 수 있었습니다. Google Docs에서 공유 문서에 이를 작성하여 취합하였으며, 작성이 모두 완료된 후 Adobe InDesign 프로그램을 이용하여 편집하였습니다.

잔류 기간이 길지 않다 보니 구상한 모든 아이디어를 실현할 수 없었다는 점이 작은 아쉬움으로 남습니다. 그럼에도 FTC 2023-2024 시즌을 준비하며 로봇 제작의 기술과 프로그래밍에 대해 더 배울 수 있었을 뿐만 아니라, 로봇 제작과 프로젝트 진행의 전반에 대해서도 많은 것을 배울 수 있었습니다. 연구회 KROS에서 여러 차례 FTC에 출전한 만큼, 선배들의 경험을 토대로 하여 직접 시행착오를 겪으며 팀 전체로써 더 발전할 수 있는 기회가 되었다고 생각합니다.



PLANS : 향후 계획

FTC 2023-2024를 준비하면서 연구회 KROS와 Team TALOS에 대한 앞으로의 계획에 대해서도 고민해볼 수 있었습니다. 연구회와 팀이 꾸준히 유지되며 매년 대회에 참가할 수 있도록 하기 위해, 앞으로 다음과 같은 활동을 더욱 진행해 보고자 합니다.

1) 신입 연구회원 선발 및 인수인계

새로 학교에 들어오는 24학번 신입생들 중, Team TALOS에 적합한 인재를 면접을 통하여 선발할 계획입니다. 또 새로 들어온 24학번에게 이번 2023-2024 FTC에 대한 경험을 바탕으로 앞으로의 FTC도 원활하게 진행할 수 있도록 보조해주고 전수해줄 계획입니다.

2) 대회 피드백

학교 개학 이전까지는 다같이 회의를 하기 힘들기 때문에 개학 이후에 대회 동안 느꼈던 감정들을 공유하고 더 나은 Team TALOS가 되기 위하여 지난 대회를 피드백할 계획입니다.

3) 연구회 홍보 및 다양한 활동

연구회 KROS를 홍보하는 데에 더 노력하고, FTC이외에도 연구회 KROS, TEAM TALOS가 할 수 있는게 무엇이 있을지 찾아보고, 더욱 다양한 활동을 추진할 계획입니다.

ENGINEERING NOTE

TALOS
#5073