**Capstone Design I**

Group 9 progress report

2018년 6월 16일

**한국과학기술원 기계공학과**

**20150804 한지혜**

감사의 글

초록을 대신하여, 보고서로 들어가기 앞서 감사의 글을 적습니다.

먼저 창의적 시스템 구현 수업의 책임을 맡아 학생 하나하나 찾아다니며 불만사항을 듣고 챙겨주신 박수경 교수님 외 여러 교수님들, 특히 9조를 맡아 함께 고민하고 조원들이 결속할 수 있도록 지원과 칭찬과 조언을 아끼지 않은 오일권 교수님, 9조 담당 조교이자 학교 선배로서 수업 뿐 아니라 심적으로 많은 보탬이 되어주신 이강규 조교님, 이강규 조교님이 잠시 자리를 비우셨을 때 대신 챙겨주신 오재성 조교님, 그리고 인터넷 결제 문제로 많은 도움을 주신 오일권 교수님 연구실의 임지영 선생님께 감사드립니다.

그리고 그 누구보다도 동고동락하며 같이 밥도 먹고 밤도 샌 팀원이자 친구들- Vivek, 예성, 민우, 우석, 태연, 태홍 오빠, 수현. 처음 만나서 어색할 때가 아직 어제 같은데 무사히 미션을 성공해서 즐거운 마음으로 마무리하고 벌써 종강이다. 비록 프로젝트는 어려웠고 한 학기가 어떻게 지나갔는지 모를 정도로 바쁘고 힘들었지만, 기계공학과 학부 마지막 학년에 너희 같은 좋은 사람을 만나 끈끈한 인연을 쌓을 수 있었던 것만으로도 충분히 가치 있는 고생이었다고 생각한다. 너희가 내게 그런 만큼, 나도 너희에게 좋은 친구로 남았기를 바라며.

1. **서론 및 개요[[1]](#endnote-1)**

창의적 시스템 구현 수업은 졸업을 앞둔 기계공학과 학생이 이전까지의 수업에서 배운 내용을 토대로 창의적으로 어떤 시스템을 제안하고 제작하는 수업이다. 이 과정에서 학생은 필요한 지식을 이미 알고 있는 내용에서 적용하거나, 새로 인터넷 혹은 관련 논문, 상위 과목 등에서 공부하고, 이를 실제 시스템에 맞게 설계 및 실험으로써 학습하게 된다. 또한 이 모든 과정을 혼자 진행해온 기존 수업들과 달리, 창의적 시스템 구현은 조별 활동으로 구성된다. 따라서 미션을 수행하기 위해서 각자 담당한 내용을 조원들과 공유하고, 논의를 통해 솔루션을 도출하고 이를 통합하며 함께 시스템의 기획, 설계, 구현을 진행하는 예행연습을 거치게 된다.

특히 18학년도 창의적 시스템 구현(이하 창시구) 수업에서는 최근 화두가 되고 있는 4차 산업혁명의 흐름에 맞추어 기존과는 색다른 지향성을 갖고 수업을 구성하였다. 기존의 기계공학이 힘의 운용과 에너지 전달에 기반한 기계의 작동 원리에 초점을 맞추었다면, 앞으로는 인공지능과 로봇 기술을 실제와 가상이 통합되고 사물을 자동제어할 수 있는 시스템이 구축되는 산업상의 변화가 있으리라고 예측된다. 특히 소프트웨어 기술을 활용한 기반 기술과 전통적인 의미의 기계공학 시스템이 서로 어떻게 연계되는지를 배우기 위해, 이번 창시구 수업에서는 비전 센서를 활용한 자율 주행의 기초 하드웨어와 소프트웨어를 공부할 수 있도록 미션이 주어졌다.

1. **프로젝트의 목표**

주어진 미션은, 간단하게 말해, 랜덤으로 놓인 세 개의 빨간 공과 세 개의 파란 공 중 파란 공 세 개만 가져오는 시스템을 구축하는 것이다. 이때 바퀴는 mechanum wheel 두 쌍을 사용하고, 바퀴를 구동하는 모터는 Dynamixel로 주어진다. 또한 모터 제어는 myRIO에서 LabVIEW 컨트롤로 진행되며, 웹캠 카메라와 OpenCV 비전 인식 프로그램을 활용하여 공을 인식한다. 이 모든 것은 인텔사의 소형 컴퓨터인 NUC에서 제어되며, 특히 로봇 운영 체제인 ROS를 활용하여 전체 시스템을 통합하고 제어할 수 있도록 한다.

구체적으로 고려해야 할 목표 사항은 다음과 같다:

* 빠르고 안정적으로 미션을 수행할 수 있도록 모터 제어 시스템 설계: 기구 설계, 시스템 모델링 및 제어의 응용
* 안정적인 비전 인식과 차체 구동을 위한 진동 저감: 진동 공학, 차량 모델링 및 제어의 응용
* 공을 수집하는 데에 가장 효율적인 알고리즘을 구축하고, 이를 실제로 응용할 수 있는 소프트웨어 기반 수립: 알고리즘 설계, 소프트웨어 및 컴퓨터 언어 응용
* 전체 시스템의 온도를 적정 수준으로 유지하기 위한 쿨링 시스템 설계(구체적으로, 섭씨 70도를 넘지 않도록 할 것): 열전달, 열시스템 설계 응용
* 그리고 무엇보다 이 모든 것을 포함한 ‘창의적인 시스템 구현’

*앞서 간략하게 언급한 바와 같이 본 창시구 프로젝트에는 한국과학기술원 기계공학과 학부과정에서 수학한 다양한 지식을 동원하게 된다. 그러나 구체적인 적용 과정은 각 조마다 목표하고 계획한 바에 따라 큰 편자를 보일 것으로 생각된다. 본 장 이하로는 특별히 우리 조에 국한된 설명을 기술하며… 어떤 학문적 성과보다는 현실적인 팁을 전수(?)하기 위해 일부 격식에서 벗어난 문장이 있음을 미리 양해 부탁드립니다. 과학적인 내용보다 다음 수강생에게 전하는 팁(?)에 가까운 내용은 기울임체로 기록합니다.*

1. **배경 지식**

9조(이하 IKOH)에서 활용한 배경 지식은 다음과 같다. 소프트웨어 파트에서 활용한 코드는 특별히 기술할 만한 배경 지식을 요하지 않으므로 필요한 부분은 아래 progress report에 기술한다. IKOH의 경우, 주로 진동 저감은 모터의 소프트웨어 제어를 통해, 열 전달 효율 극대화는 하드웨어를 보강하여 해결하였다.

1. **진동 저감**
   1. **모터 입력 제어**

일반적으로 모터의 입력 신호는 step function으로 주어진다. 그러나 진동공학과 기구설계학 두 수업 모두에서 배우지만, 이는 변위의 시간에 대한 미분항, 두 번 미분항, 세 번 미분항 모두가 불연속이기 때문에 급가속/급감속할 뿐 아니라 차체에 jerk가 걸려 무리가 간다. IKOH는 이를 해결하기 위해 reference filtering을 활용한 low-pass filter를 설치했다. 즉,

Z transform을 적용하면,

* 1. **Damper 설치**

진동측정기를 사용해 차체 전체의 진동수를 측정했다. *당연하지만 바퀴 흔들리면서 발생하는 진동이 제일 크다.* 측정해보면 전체 진동 , 바퀴의 진동 로 바퀴 진동수와 차체 진동수가 대략 비슷한 것을 알 수 있다. 바퀴의 진동수는 추정치임을 감안하면 거의 바퀴 움직임에 의한 진동이라고 추정할 수 있다.

비전 인식을 위해 웹캠을 설치하는데, 진동이 심하면 웹캠 화면이 흔들리면서 인식이 잘 되지 않는 문제가 발생한다..*고 추측했지만 사실 인식 잘 된다.* 그보다 큰 문제는 차체 진동 때문에 웹캠 고정이 조금씩 풀리면서 시야가 틀어지는 부분이었다. 자칫 목표물이 화면에 안 잡힐 수도 있고, 시야각이 바뀌면 설정한 거리(pixel값을 실제 거리로 변환하는 비율)가 미션 중에 틀어질 수 있다. 따라서 댐퍼는 웹캠 아래쪽에 설치하기로 했다. 세탁기, 자동차 등에 사용되는 소음차단기부터 청소용 스펀지, 화장용 퍼프 등 생각할 수 있는 스펀지 재질의 물건을 여러 가지 구해서 실험했고, 그 중 최종적으로 진동이 가장 줄어든 것을 선택하였다.

댐퍼의 효율을 구하기 위해, 댐퍼가 변형되는 면적으로 용수철 상수 k를 측정하여, 웹캠을 질량으로 하는 전체 용수철 시스템을 등가 질량과 등가 용수철 상수로 변환, 최종적으로 앞서 계산한 바퀴의 진동수와 비교하여 transmissibility를 구했다.

1. **열전달**

IKOH는 Nusselt 수를 계산하여 heat transfer coefficient를 계산하는, 열전달 수업에서 배운 방식의 해석 대신 하나의 열시스템으로 설계하고 이 전체의 효율과 비용을 맞추는 방식을 택했다. 즉, 파트 각각의 열전달계수를 계산하고, 팬 등을 달아 이를 개선하고 다시 열전달계수를 계산하는 대신 아예 전체 시스템을 냉각할 수 있도록 팬 여러 개로 구성한 duct를 이용했다.

효율은 좋았지만 여기서 한 가지 문제가 발생한 것이, duct 안에서는 어떤 모델을 적용해 열해석을 하기 힘들었다는 점이다. 굳이 우리가 배운 분류 내에서 나누면 duct는 internal flow기는 한데, entry region이니까 internal flow 공식을 적용할 수는 없고, 그렇다면 heat exchanger를 사용하자니 우리가 임의로 만들어놓은 시스템이라 heat exchanger 상수도 하나도 아는 바가 없고…최종 발표는 가까워지고….그나마 합의점(?)으로 우리가 그동안 추론해온 방식을 가능한 한 과학적인 용어와 공식을 통해 풀어내도록 노력했다.

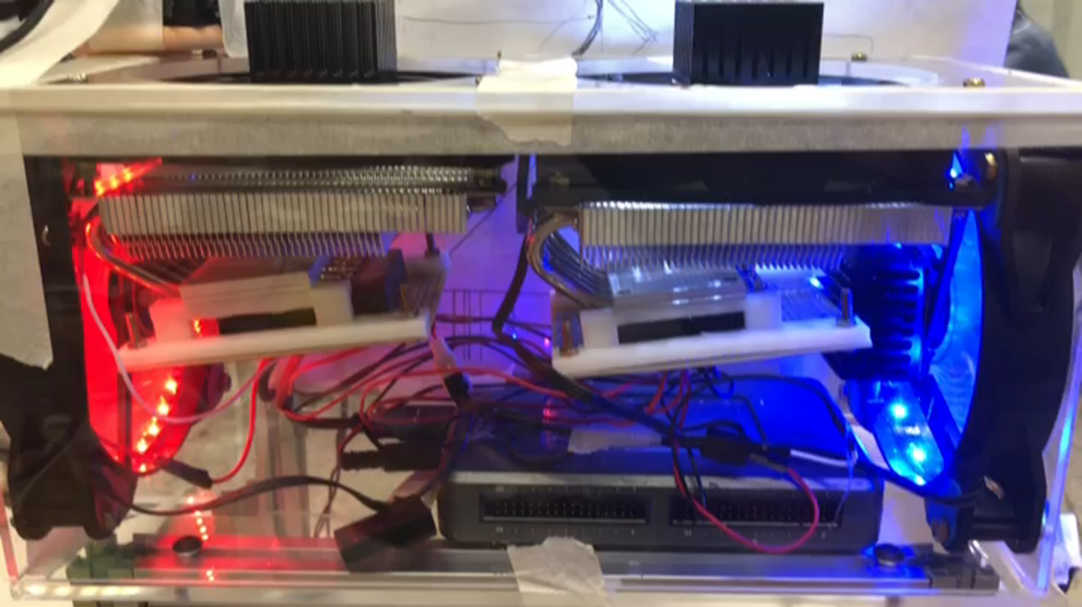


그림 1. Duct 시스템

구체적인 제작 과정은 아래 progress report에서 다시 다루게 될 것이므로 넘어간다. 간략하게 말하면 우선 컨버터가 가장 뜨겁기 때문에 CPU 쿨러를 이용해 강하게 냉각시켜줄 필요가 있었고, myRIO나 기타 부품도 함께 식혀주기 위해 duct를 제작하여 전부 duct 안으로 넣기로 했다. Duct가 밀폐공간이므로 CPU 쿨러의 유량만큼 duct 안으로 들어오는 유량이 있겠지만, 공기의 흐름을 단순히 압력차에 의존해 유도하기보다 다시 팬을 이용하여 보다 원활하게 유지하도록 하기 위해 팬을 다시 달았다. 위 그림 1에서 양 옆에 두 개, 위에 두 개 달려있다. 양 옆 두 개는 duct로 들어오는 방향, 위의 두 개는 CPU 쿨러와 연결된 팬인데 위로 나가는 방향의 팬이다. 차가운 공기는 아래로 들어오고 뜨거운 공기는 위로 나가도록, 그리고 제한된 공간 안에 전체 회로를 효율적으로 배치하기 위해 그렇게 배치하였다. 이왕 하는 김에 팬에 LED가 들어오는 것이 있길래 그걸로 했다. 예쁘다.

*그리고 온도 측정할 때 열화상측정기는 쓰지 말자. 열화상측정기마다 사양 차이는 있지만 온도 범위로 나오기 때문에 그리 정확하지 않고, 최고 온도만 찍어주기 때문에 다른 파트의 온도는 알 길이 없다. 특히 표면의 radiativity에 따라서 측정된 온도가 심하게 부정확할 수 있다. 아마 최선은 thermocouple을 DAQ보드에 연결하여 측정된 온도 데이터를 LabVIEW로 받는 방법인데, 일단 DAQ보드가 없기 때문에…. 그래도 적어도 thermocouple이나 thermometer를 이용하도록 하자.* IKOH는 한 번 실험에 여러 파트의 온도를 함께 측정해야 했기 때문에 thermometer를 이용할 수는 없었고, 레이저 온도계를 사용하였다(개인 비품입니다).

1. **하드웨어 모델**

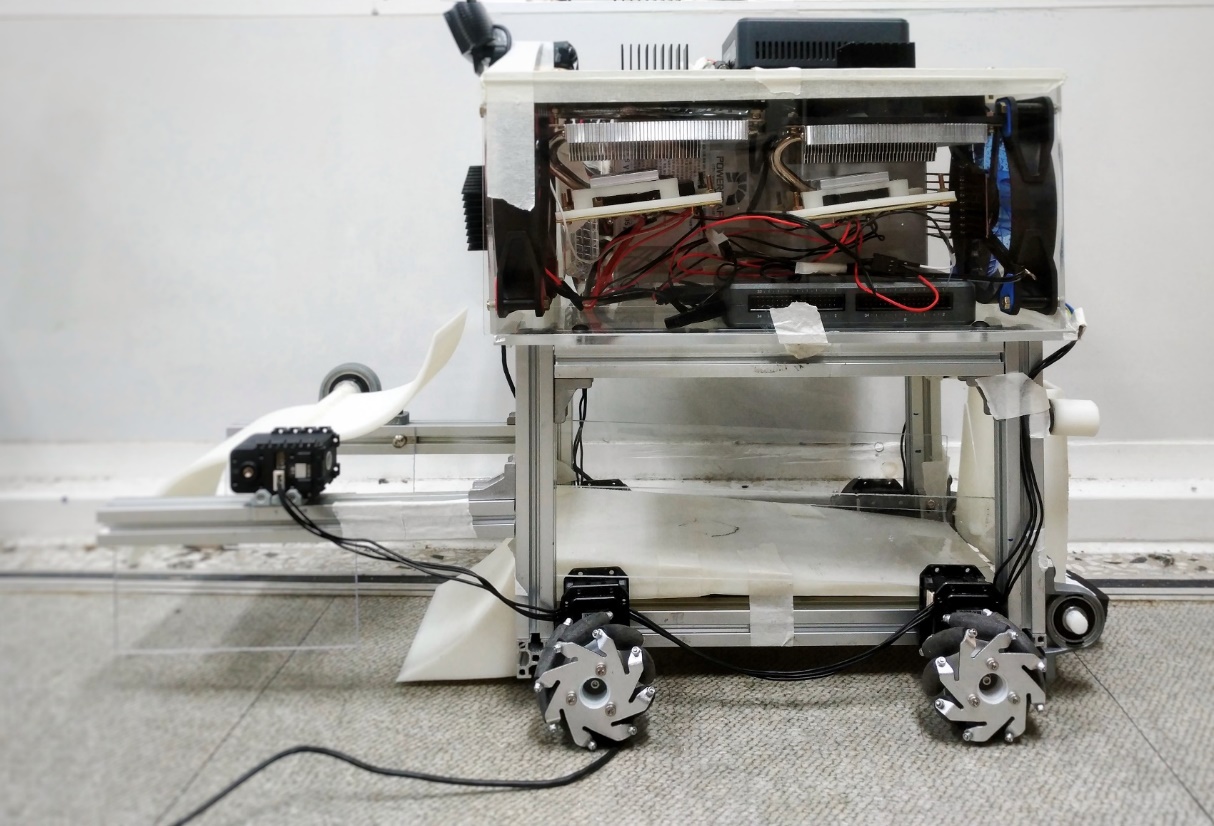


그림 2. 외관

IKOH는 앞쪽의 blade가 돌아가면서 공을 집어넣는 방식이다. 전체적인 외관은 그림 2와 같다. 특히 매 발표마다 자랑했지만, 공을 내려놓을 문을 여닫을 때 새 모터를 다는 대신 그림 3에서 보듯 문 아래쪽이 바구니에 닿으면 그 힘으로 자동으로 열리도록 했다. *모터가 은근 비싸더라…*.

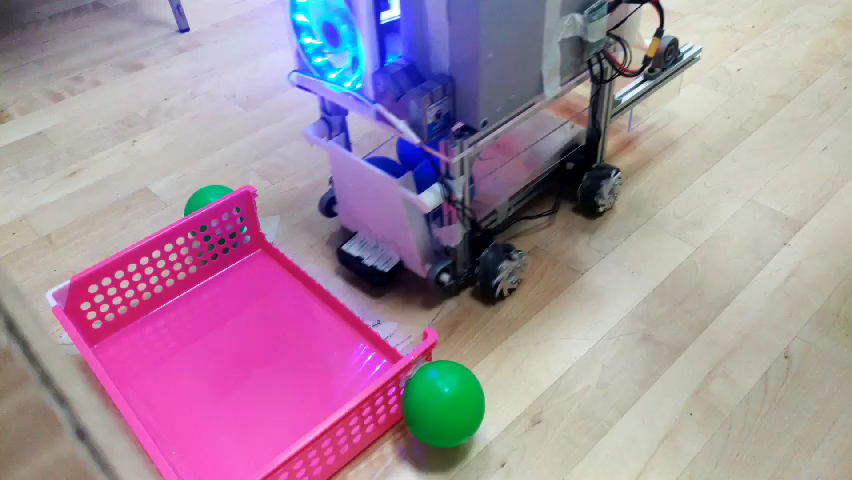


그림 3. 문의 열림

1. **과정(progress report)**

지난 과정 동안 제출한 progress report를 요약하고, 다음 수강생에게 전해주고 싶은 팁을 위주로 다시 정리하였다.

1. **전체 진행과정**
   1. 1분기(시작~1차 발표)

한 달 정도의 짧은 기간이고, 금년 창시구의 경우 이때까지 미정인 부분이 많았기 때문에 실질적으로 크게 진행된 사안은 없었다. 이런 디자인으로 만들겠다는 제안 정도. 각 파트별 강의/실습 시간이 주를 차지했다. 특히 solidworks 팀의 경우 제안된 디자인을 토대로 1차 초안을 완성하기까지가 과제였다고 들었다.

* 1. 2분기(1차 발표~2차 발표)

본격적으로 제작에 들어간 시기이다. 중간에 애매하게 중간고사가 끼어 있기 때문에 시간 배분하면서 애를 많이 먹었다. *특히, 물품이 주문한다고 바로 오질 않으니 며칠 내다보고 주문해야 한다. 주문 절차가 복잡하기 때문에, 교수님께서 담당하시는 연구실의 행정 선생님이나 조교님이 보통 도와 주시기는 하지만 조 내에서도 한 명이 구입을 전담해서 관리하는 게 혼선도 없고 기타 서류는 물론 누적 사용 예산을 관리하기도 편할 수 있다.*

* 1. 3분기(2차 발표~3차 발표)

제작은 2차 발표 즈음에 거의 완료되었고, 전체적인 시스템을 가다듬고, 불안정한 부분을 해결하는 데에 주력하는 시기였다. *2차 발표 이후에 시스템에 뭔가를 덧붙이고 싶다면, 재고해보자. 높은 확률로 생각지도 못한 문제가 터져 시간만 허비할 수 있다.* IKOH의 경우 8주차에 시간을 단축하고 싶은 욕심에 더 큰 지름의 메카넘 휠을 구매했다. 바퀴만 바꾸는데 괜찮겠지… 라고 생각했으나 바퀴 지름이 커지면서 차체가 기울고, 중심이 안 맞고, 진동이 커지고, 무엇보다 dynamixel의 회전축에 비해 새 mechanum wheel이 지나치게 커서 토크를 충분히 지탱하지 못해 온갖 문제만 발생해서 시간만 버리고 원래 바퀴로 돌아왔다.

1. **담당 파트 1 – OpenCV**

1주차. OpenCV의 기본적인 내용을 다뤘다. Matrix와 vector의 표현방식에 대한 비교가 주를 이루었다…. 그러나 실제로는 설치만 하다 끝난다. 우분투 재설치만 다섯 번 했다. 같은 조 임수현 학생은 우분투 설치하다 뭘 잘못 건드렸는지 메인보드가 멎었다. *백업은 미리미리.* *우분투에 디스크 용량은 20G 이상을 할당하도록 하자.* 행렬, 벡터, 포인트 벡터 등은 이때 배우기로는 그냥 수를 저장하는 방식의 차이이지만, OpenCV는 이미지 프로세서기 때문에 이미지 저장 방식의 차이가 된다.

*행렬은 이미지를 저장하는 방식이다. 단순하게 말하면, RGB 이미지를 행렬로 저장할 때 행렬의 한 요소가 해당 픽셀의 RGB값이 된다. 일반적으로 물리량에서 말하는, 원점을 기준으로 어떤 점의 위치를 나타내는 값은 벡터가 아니라 포인트 벡터이다. 특히 경계선을 따는 이미지 프로세싱에서, 경계선의 위치를 저장할 때 포인트 벡터를 사용한다 벡터는 위치보다 순서쌍이라고 생각하는 쪽이 이해하기 편했다. 처음 두 항은 위치이고, 그와 함께 반지름 혹은 hierarchy를 함께 저장한다. 예를 들어서, Hough Circles는 이미지 안에서 원을 찾아내 원의 위치와 반지름을 저장하는 이미지 프로세서이다. 이 이미지 프로세서는 Vec3f 형태로 데이터를 저장하는데, Vec3f라고 하면 세 항목 a, b, c의 순서쌍 (a, b, c)를 가리키며 이때 a와 b는 각각 원의 x, y좌표, c는 원의 반지름을 저장하게 된다.*

*수업만 들어서는 감이 잘 안 온다. 각 이미지 프로세서마다 저장 방식이 다르기 때문에, 저장 방식을 정확히 이해하고 있지 않으면 적재적소에 필요한 이미지 프로세서를 활용하는 데 어려움을 겪을 수 있다.*

2주차. 기본적으로 활용할 수 있는 이미지 프로세싱 방법을 배운다. Blur 처리하는 방법으로 bilatural, Gaussian, median, homogeneous blur 방식을, 경계선(edge)을 따는 방식으로는 equalize, Laplace, canny edge, Hough line transform, Hough circle transform, find contour 등을 배운다.

*원본 이미지는 경계선을 딸 때 애로사항이 많다. 경계선은 기본적으로 양 옆 픽셀 간의 색상 편차가 큰 부분을 인식하는데, 원본 이미지는 지나치게 선명하기 때문에 색상 편차가 커서 경계가 아닌 부분도 경계로 인지한다. 따라서 blur 처리를 한 번 하여 비슷한 색상 범위를 하나의 색상으로 만들고, 그 뒤에 경계선을 따는 것이 좋다. 이미지 프로세서마다 다르지만 edge를 따는 프로그램의 경우 애초에 프로세서 안에 blur가 들어가있는 경우도 있다.*

3주차. Ball detection에 사용할 수 있는 코드를 아예 배부 받고, 조별로 적용하도록 했다. 주어진 코드는 blur 프로세서로는 median blur와 Gaussian blur를 사용하고, 경계선 프로세서는 find contour와 canny edge detection을 사용한다.

4주차. ROS팀 코드와 OpenCV 코드를 합치는 과정을 진행했다. 앞서 언급한 것처럼, *이미지 저장 방식(행렬/벡터형)을 제대로 이해하고 있지 않으면 이때 OpenCV가 출력하는 저장형과 ROS가 받는 저장형이 달라서 애를 먹는다.* ROS팀의 원래 목표는 SLAM/NAVIGATION(전체 미로 지도를 식별하여 자동으로 길을 찾는 알고리즘)을 구현하는 것이었지만, *빠르게 포기했다.* 약 2주쯤 지난 뒤, 전체 수강생을 대상으로 SLAM은 2학기에 본격적으로 다루는 것으로 목표가 수정되었다. *때로는 빠른 포기가 정답일 때도 있는 것 같다.* OpenCV도 처음에는 바구니 색상을 판별하여 목표물을 찾으려 했지만, 공과 달리 바구니는 식별하기가 어려웠다. 다행히 이후 바구니 주위에 초록 공을 달아주는 것으로 수정되어 부담을 덜었다.

*IKOH가 SLAM을 미리 포기한 타당한(?) 이유는 네 가지가 있는데, 1. 기계공학동 로비의 구조를 감안할 때, 특별한 장애물이 없으므로 RPLidar를 이용한 전체 매핑만 완료하면 굳이 SLAM을 이용해 “미로찾기”를 할 이유가 없으며 2. ‘빨간 공’이 장애물로 기능은 하지만 이는 고정 장애물이 아니라는 점, 3. 알고리즘을 공부하는 데 필요한 현실적인 시간, 4. SLAM-Navigation이 그리 가벼운 코드가 아니므로 소모 전력도 고려할 때 굳이 필요하지 않다면 돌릴 이유가 없다는 점이다.*

5주차. 중간고사 기간이었다. 크게 작업이 된 것은 없고, HSV값(hue, saturation, (brightness) value – 공의 색상값)을 조절했다.

6주차. Morphological illusion 값을 조절했다. *간단히 말하면, 앞서 말했듯 이미지 프로세싱은 기본적으로 주변 픽셀과 비교하여 색상값이 크게 변하는 구간을 경계선으로 인식한다.* 그러나 이때 한 물체 안에서도 색상이 튀는 부분이 있고 이는 blur처리로도 보정이 어렵다. 예를 들어, 공이 직사광선을 받으면 그 부분만 명도가 너무 높아 다른 부분으로 인식된다. 일반적으로는 인식할 때 크게 상관이 없지만, 간혹 인식되지 못한 면적이 너무 많으면 공을 작은 공 두 개로 인식하는 경우가 생기고 그러면 공까지의 거리도 잘못 계산해버린다. *Morphological illusion은 이를 보정하기 위해서 분해능을 낮춘다. 즉, 인식된 부분의 면적을 넓히는 작업이다.* 예를 들어 파란 공을 인식할 때, 직광을 받은 부분만 인식이 안 되었더라도 인식된 면적을 전체적으로 넓히면 자연스럽게 공 전체를 인식한 것으로 처리되고 경계선 처리 알고리즘으로 넘어가게 된다. 이때 공 자체가 조금 크게 인식되지만, 이는 경계선 처리 후에 반지름을 보정해주면 되므로 거리 인식에는 크게 문제가 없다.

7주차. 기존에는 find contour 코드를 사용했지만, 공을 인식하는 정확도를 높이기 위해 hough circles 이미지 프로세싱을 이용하는 코드로 변경했다. Find contour와 hough circle이 서로 저장하는 저장형이 달라서(find contour는 포인트 벡터형으로 contour의 위치를 저장하고, hough circles 코드는 3차원 벡터형에 x, y, r값을 순서대로 저장한다) 저장형을 통일해주는 작업에 약 2주가 걸렸다. *아직 OpenCV 자체가 초기 단계라 그런지, 아카이브에 많은 자료가 없다. 이미지 프로세싱을 하다 보면 조절해야 하는 값들이 많다. Hough Circle의 예를 들면, hough circle 자체에 canny edge processing 처리가 들어가는데 이에 대한 보정값, 원형을 인식하므로 원과 원 사이의 거리, 원으로 인식하는 최소 반지름, 최대 반지름, 분해능 등. 그러나 무슨 숫자가 어디에 대응되는지는 아카이브와 위키에 적혀있지만 구체적으로 활용하는 방법은 아직 구체적인 설명이 없어서 직접 trial-and-error로 수고해야 한다.*

8주차. 중간고사 이후 역할 분담을 재배치하면서, 영상 분석 쪽은 어느 정도 완료가 되어 OpenCV 담당하는 다른 팀원에게 거의 전담하고 하드웨어 부분에 집중했다.

Find Contours 기반 코드를 이용해서는 ROS의 ball picking 알고리즘이 잘 돌아갔지만, hough circle은 프로세싱 변수를 조절하는 데에 난항을 겪었다. Hough circle은 갈 길이 멀고, 하드웨어 작업도 충분히 많아서, 잘 되는 find contours 코드로 선택했다.

10주차. Find contours 코드에서 심각한(?) 문제가 발견되어 소프트웨어 팀에 잠깐 돌아갔다. 공을 두 개로 인식했다. 우리 조 뿐만 아니라 전체 조가 동일한 문제를 겪고 있었으며 배부된 프로그램의 문제였다고 한다. *개인적으로 추측하는 부분은 다각형 근사에서 오류가 있지 않았을까 싶다. 즉, approxpolyDP와 mincircle이라고 하여, find contours로 찾은 경계선을 가장 근사한 polygon으로 한번 근사하고(approxpolyDP), 이를 다시 circle로 바꾸어 해당하는 반지름을 찾는(mincircle) 과정이 있다. 그런데 approxpolyDP 코드가 찾은 컨투어에서 상하좌우 네 점을 잡아서 이에 외접하는 polygon을 그리는 방식이다 보니, 특히 원에 대해서 불안정하다*. 이 때문에 문제가 발생했을 확률이 가장 큰 것 같다.

Hough circle 코드를 실행했을 때는 이 문제가 없었는데, Hough circle은 코드 자체에서 원으로 인식하고 반지름까지 출력하므로 approxpolyDP와 mincircle 코드를 사용하지 않아서 이런 문제가 없었던 것으로 보인다. 사실 매우 근접한 공 두 개로 인식하고, 어차피 공을 주운 다음에는 인식되지 않으므로 크게 문제될 것은 없다. *그러나 코드가 한 번 잘못되면 다시 예상하지 못한 문제가 우후죽순 생기곤 한다.* 이것이 걱정되어 크게 패닉했었지만 조교님이 원래 그렇다고 말씀해주셔서 한숨 돌렸다(?).

IKOH는 바구니 근처에서 180도 돌아 후진하여 공을 넣는 방식인데, 원래 후진 거리를 맞추는 데에 문제를 겪고 있었다. 이날 이것저것 하다 보니, *공이 화면 밖으로 조금 나가 컨투어가 잘리면 공 하나로 인식되는 것을 발견했다(개인적인 추측 - approxpolyDP에서 점을 세 개만 잡게 되면서, 세 점을 지나는 외접원 하나를 깔끔하게 그릴 수 있게 되어 그런 듯 하다).* 공이 화면 밖으로 나가면 하나로 인식되고, 화면 안에 들어오면 두 개로 인식되는 점에 착안하여 바구니를 표시하는 초록 공과의 거리를 맞출 수 있게 되면서 알맞은 후진 거리를 구할 수 있었다. *때로는 문제도 이용할 수 있다.*

1. **담당 파트 2 – 하드웨어(열전달)**

3주차. 차체의 주요한 파트를 3D 프린터로 뽑고, 알루미늄 바를 이용해 조립했다. 배터리팩이 도착하여 회로 납땜까지 진행했다. 납땜할 때 컨버터를 학생들이 직접 구매해오도록 하는데, *인터넷으로 주문하는 편이 좋다. 대흥동 공구상가에 없다. 처음에 컨버터 용량 때문에 많이 헤맸는데, 정격 전압은 범위만 맞춰주면(myRIO 6~16V, NUC 12~19V) 크게 상관 없다. 정격 전류를 맞춰주고, 특히 NUC는 정격 전력에 유의하자. 옆 조는 12V 컨버터에 NUC 끼우고 잘만 돌렸는데, 우리 조는 15V에 팬과 같이 연결했다가 전력 부족으로 NUC가 강제종료되었다.*

4주차. 컨버터와 기타 필요한 물품을 주문했다. *앞에서 언급한 대로, 행정선생님이 도와주시기는 하지만 후불 구매 신청하고 견적서 수령하고 다시 세금 결재 받는 과정이 처음 하면 복잡해서 차라리 한 사람이 담당 하면 혼선이 없어 편하다…고 본인이 주문 및 결재 담당으로 고생했으니 당당하게 말해본다.*

5주차. 팬과 핀을 달기로 했다. 처음에는 굳이 달지 말까…하기도 했지만 팬 하나 소비전력이 10W가 채 안 되지만 효율은 상당하다고 해서 이예성 학생이 원래 갖고 있던 여분 팬을 달아보았다. *효과 좋다. 컨버터 기준으로 팬을 달면 30~40도, 핀만 달아도 거의 20도 정도 온도가 떨어진다.*

7주차. 2차발표를 지나 프로젝트가 중간을 넘어가면서(대략 중간고사 이후) IKOH 팀원들은 Solidworks/LabVIEW/ROS/OpenCV로 나뉜 기존의 역할 분담을 재배치하여, 팀원 각각의 세부 전공(진동제어/열유체)에 따라 소프트웨어 및 진동저감/하드웨어 및 열효율 보강 두 팀으로 나누었다. 예를 들어 본인은 본래 OpenCV 담당으로 소프트웨어 쪽이었지만, 세부 전공이 열 전달이라 중간 이후로는 하드웨어 팀에 합류하여 열 전달 쪽을 담당했다. *2차 발표 이후의 목표에 맞춘 재배치이면서, 자연스럽게 하드웨어/소프트웨어 두 팀이 한 번 섞이면서 서로 입장을 이해하고 최종 통합하기가 한결 수월하게 진행되었다. 물론 팀원 전체가 한 자리에 모여 프로젝트를 진행하는 방법도 있었겠으나 8명이 시간을 맞추기가 현실적으로 그리 호락호락하진 않았다.*

8주차. Mechanum wheel과 관련 연결 부품을 새로 구입했다. 운행 속도를 높이기 위해서였지만, 이후 전체공지로 바퀴 지름을 바꾸면 안 된다고 말씀하셔서.. 이미 달아버린 조는 어쩔 수 없으니 지름이 큰 것을 감안하고 채점한다고 하셨다.

열전달 효율을 높이기 위해 CPU 쿨링 팬과 전체 duct에 사용할 팬을 추가로 구입하고 duct 구조를 재설계했다. *CPU 쿨링 팬은 확실히 효과가 있었다. 운행하는 내내 컨버터 온도는 거의 오르지 않았다(IKOH 팀 실제 운행 결과와는 차이가 있을 수 있습니다만, 데모 전날에 NUC의 전력 부족 문제로 급히 컨버터 하나를 새로 달면서 여기에는 팬을 못 달아서 그렇습니다).* 이론적인 설계를 병행하려 했으나, 앞서 배경 이론에 적은 것처럼 duct가 이상적인 상황(적어도 fully developed된 상황)이 아닌 관계로 불가능했다. 대신 ventilation system을 기본 골자로 하여 개략적인 유동을 고려, duct를 재설계했다.

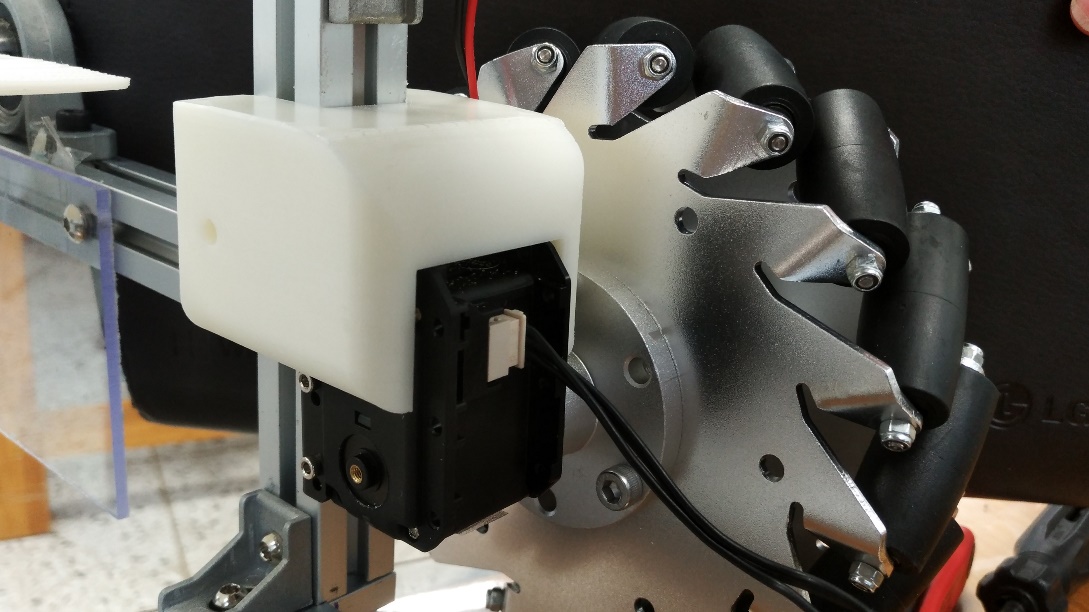
9주차. 앞의 blade(공을 모으는 회전판)를 원호 모양으로 변경하고, 152mm mechanum wheel로 바꾸어 달았다. 바퀴가 속을 많이 썩였다. 속도는 확실히 빨라졌지만 진동이 훨씬 커졌다. 더군다나 dynamixel에 비해 지나치게 크고 무거워서 dynamixel과 알루미늄 바의 연결부가 자꾸 느슨해지는 문제가 발생했다. 몇 번을 떼고 다시 달았는지 모르겠다. 최민우 학생이 3D 프린팅으로 다이나믹셀을 알루미늄 바에 꽉 붙여둘 수 있는 구조를 뽑아 해결했다(그림 4).

그림 4. 다이나믹셀과 알루미늄 바의 연결

그랬더니 이번에는 dynamixel 회전축이 바퀴에 비해 터무니없이 작아서 토크가 부족하고 바퀴 각도가 휘는 문제가 발생했다. 그러면서 차체가 구동할 때 회전하는 각도가 제대로 맞지 않아 후진하고 공을 바구니에 넣을 때 정확도가 많이 떨어졌다. 소프트웨어 팀의 경우 wheel kinematics를 이용하여 자동제어를 시도하고 있었는데, 큰 바퀴는 일정한 입력 신호를 받아도 매번 가는 방향이 틀어져서 wheel kinematics가 불가능했다. 결국 조원 전원(사실 민우만 포기 안 했지만)이 지쳐서 포기하고 원래의 작은 바퀴로 도로 바꾸어 달았다.

10주차. 팬을 추가로 납땜하여 현재 총 4개의 팬으로 duct를 운용한다. 팬을 달기 전/후, 예전 duct와 현재 duct 등을 비교하여 최종적으로 온도가 얼마나 올랐는지를 측정. NUC가 고장나는 바람에 크게 당황했다. 팬을 4개 장착하면서 컨버터를 공유했는데, 이때 전력이 부족하여 우리도 모르게 몇 번 강제종료가 되면서 컴퓨터에 부담이 되지 않았나 추측한다. 여분 NUC를 받아 새로 우분투와 ROS, OpenCV등을 급히 설치하고 기존 NUC는 수리를 맡겼다. *myRIO를 보다 수월하게 식히기 위해 핀을 달았었는데, 핀을 다니 myRIO가 와이파이를 잘 수신하지 못하는 것을 발견해 핀을 떼었다.*

1. **소감**

결과는, 두 번 트라이얼 중에 첫 번째는 파란 공을 주워온 뒤 후진해서 내려놓는 부분에서 실패했다. 원인은 정확히 알 수 없으나, 웹캠의 위치가 틀어졌거나… 그냥 이유 없이 초록 공의 위치를 잘못 인식했거나. 중간에 막간을 이용하여 웹캠의 위치를 조절하고 다시 calibration을 진행했다. 다행히 두 번째에는 깔끔하게 성공했다.

내 친구들이 대부분 내년에 창시구를 듣다 보니, 조금이라도 도움이 되지 않을까 하는 마음에 어떤 학술적(?) progress report보다는 실제로 겪고 고생했던 이야기를 더 많이 적은 것 같다. 솔직히, 창시구는 정말 바쁘고, 힘들고, 고생스러운 과목이다. 모든 사람이 입을 모아 그렇게 말하니 수강하기 전부터 막연하게 스트레스였다. 그리고 실제로 수강해보니 너무 바빠서 힘든 줄도 모르고 지나갔고, 3차 발표 끝나기 무섭게 쓰러져서 죽은 듯이 잠을 잤지만…. 그러나 그렇게 어찌어찌 한 학기를 무사히 마치고 나니 왜 이 과목이 기계공학과 전공 필수로 지정되었는지 막연하게 알 것 같다.

우리가 수업 시간에 배운 이론과 실제는 매우 다르다. 실제 시스템은 훨씬 복잡하고, 전혀 예상치 못한 결과를 가져오기도 한다. 최첨단 기술이 집약되어 만들어진 핸드폰이나 컴퓨터도 몇 달만 사용하면 앱이 안 열리고 배터리 수명이 닳기 시작하는데, 우리가 4개월 조금 안 되는 시간에 만드는 조잡한(?) 시스템은 말할 것도 없다. 어제 멀쩡히 구르던 모터가 서는가 하면 반대로 돌아갈 리가 없는 코드가 돌아간다. 사실 창시구 수업을 들으면서 가장 난관에 부딪친 부분은 그런 예상치 못한 곳에서 발생하는 어려움이었다.

수업의 가장 근본적인 의의는 우리가 기계공학과에서 배운 과목들- 4대역학과 9대 기반과목을 필두로 기타 여러 과목에서 배운 내용을 실제로 적용해보는 것이다. 그러나 그보다도 실제 시스템을 설계할 때 무엇을 각오해야 하는지, 예상치 못한 곳에서 어려움이 발생해 얼마나 당황스러울 수 있는지, 다른 사람들과 협업하며 함께 솔루션을 찾아가는 과정은 또 얼마나 복잡하고 그럼에도 뜻밖의 시너지 효과가 발휘될 때 그것이 얼마나 가치 있는지를 수업으로 배울 기회를 제공하는 학교는 그리 많지 않으리라 생각한다. 창의적 시스템 구현은 졸업을 앞두고 우리가 무엇을 위한 어떤 공부를 해왔으며 앞으로 어떤 길을 살아가게 될지를 배운 수업이었다.

그리고 몇 번을 적어도 미처 다 담지 못할 것 같지만, 보고서를 끝으로 한 학기의 창의적 시스템 구현 수업을 마무리하면서 한 학기 함께해준 조원들에게 진심으로 다시 감사를 보낸다. 지금 이렇게 긍정적으로 수업을 기억하는 것은 서로의 의견을 존중하고, 누군가 무너지면 다독여 일으켜주고, 자신이 힘들 때 도움을 요청할 줄 알고, 맡은 일에 책임을 지고 적절한 일을 타인에게 양보할 줄 아는 진실된 사람들을 만났기에 가능했다. 그런 사람들과 한 팀을 이루어 공부했기에 때로 난관에 부딪쳤지만 절망하기보다 서로 위로하며 다른 해결책을 찾는 법을 배웠고, 결국에 좋은 결과로 맺을 수 있었다.

1. References

   창시구 2018 교재, ver 1.9 [↑](#endnote-ref-1)