**창의적 시스템 구현 Final Report**

**20150721 조인준 (adv prof. 김형수)**

|  |  |  |
| --- | --- | --- |
| **Progress Report #1** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 3/19~3/25 |
| **What I have done:** 먼저 PC에 우분투를 깔아야 했습니다. 우분투는 리눅스 개발환경을 만들어줄 수 있는 OS 중 하나인데, 가장 보편적으로 알려진 리눅스 중 하나입니다. 저 같은 경우에는 조교님들께서 추천하지 않은 방법인 윈도우 환경에 가상머신을 이용해서 우분투를 깔게 되었습니다. 제 PC가 파티션을 따로 만들어서 우분투를 까는 것이 불가능해서 부득이하게 블루스크린이 뜰 수 있는 리스크를 안고 개발환경을 만들었습니다.  C:\Users\oprij\Desktop\스크린샷, 2018-03-28 07-45-50.png개발환경을 만든 후에는 주어진 절차에 따라서 ROS를 다운로드 했습니다. 조교님들이 만들어주신 교재를 사용할 수도 있었는데, 저 같은 경우에는 조교님들의 교재를 보다가 표윤석 박사님이 쓰신 책에서 많이 참고하신 것 같아서 보다 더 많은 설명이 나와있는 책을 따라가면서 ROS를 다운로드했습니다. 조교님들이 교재 페이지 하단에 남겨주신 출처를 타고 들어가서 ROS wiki등의 사이트에서도 참고를 많이 했습니다. 그렇게 ROS를 다운로드하고, 책에 나와있는 대로 계속 하다 보니 Publisher와 Subscriber 노드, Service server와 Client 노드를 만들 수 있었고, ROS launcher도 사용해볼 수 있었습니다. 추가 과제로 나온 publish와 subscribe를 동시에 할 수 있는 노드도 만들었습니다.  금요일에는 조교님들께서 code review 세션을 열어주셔서 참석했습니다. Code review 세션에서는 xbox controller와 lidar, 카메라 등을 연결해서 사용하는 법과 그에 대한 코드들이 어떤 의미를 가지는지에 대해서 가르쳐주셨습니다. Code review 세션 이후에 저희 조원들 중 ROS 파트는 따로 모여서 code review 세션에서 했던 내용을 다시금 짚어봤습니다. 직접 해 보면서 우선 lidar가 어느 정도로 잘 detecting을 하는지 시험해보았는데, 토의 후 lidar는 공을 직접 찾는 것보다 mapping에 사용하는 것이 적절하다는 결론이 나왔습니다. Lidar의 원리가 기본적으로 점 레이저를 계속적으로 쏘아서 반사되어 나오는 빛을 받아들여서 거리를 측정하는데, 점의 간격이 생각보다 좁지 않고 높이 차이가 조금만 나도 물체를 인지할 수 없었기 때문입니다. 카메라를 사용하면서 ball detection node가 계속 실행이 안되어 조교님께 여쭤보았고, 조교님이 원격조종으로 코드를 봐주시면서 distributed code에 x좌표 설정만 있고 y, z좌표 설정은 없는 것을 발견해 주셨습니다. | | |
| **What to do next week:** ROS에 대한 기본적인 이해가 끝났고, 구조적인 이해도 어느 정도 되고 있는 단계입니다. 다음주까지는 전체적인 프로젝트를 위해서 실제로 해볼 수 있는 부분을 찾아서 프로그래밍을 해 보면 좋을 것 같습니다. 그 밖에도 ROS는 모든 부분을 통합해야하는 역할이기 때문에 다른 부분들에 대한 이해 또한 필요합니다. 그렇기 때문에 다른 파트를 맡고 있는 조원들의 progress 또한 지켜보면서 이해하고, 각 파트를 어떻게 하면 잘 합칠 수 있을지 계속 생각해 볼 예정입니다. 마지막으로 저는 발표 준비를 맡게 되어 발표 준비에 치중할 계획입니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #2** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 3/26~4/1 |
| **What I have done:** 첫 번째 발표를 하기로 했기 때문에 발표 준비를 했습니다. 발표 준비를 하기 위해서 우선 criteria로 올라온 각 파트에 대한 조사들을 다른 조원들이 해 주었습니다. 저는 그 조사를 바탕으로 발표 자료를 담당하는 조원과 함께 발표를 준비했습니다. 내용 취합 후 발표 자료 순서에 따라서 대본을 만들었습니다.  발표는 기본적으로 여태까지 “어느 정도의 progress가 있나” 에 초점이 맞춰졌는데, 로봇을 이루는 각 파트(예를 들면 열전달, 서스펜션, 공을 가져올 방법 등)에 대해 팀원들이 조사 및 분석한 것을 바탕으로 만들게 되었습니다. 아직 brainstorming 단계에 있고 정해지지 않은 아이디어가 많았기 때문에 decision matrix와 수 시간의 토의를 통해서 ball retrieving process나 열전달 방법에 대한 결정을 했습니다.  발표 준비를 하면서 ROS 파트에서 어떤 진전을 보여야 할지도 생각했습니다. 현재 상황에서 프로젝트 진행을 위해서 어떤 일을 할 수 있을까 고민을 했는데, 처음에는 “한번 이정도로 알고리즘을 짜면 어떨까” 라는 식으로 써보다가 pseudo code를 짜 놓으면 도움이 되겠다는 생각에 pseudo code를 짜기 시작했습니다. 아직 작업 중에 있지만, pseudo code를 미리 짜놓고 계속 optimizing이나 simplifying에 대한 고민을 한다면 후에 ROS로 직접 소스 코드를 짤 때 도움이 많이 될 것이라고 생각합니다.  Pseudo code는 ball detecting과 ball picking up 방법에 대한 부분인데, 전체적인 틀만 짜여 있고 세부 사항은 조원들과 상의 후에 결정해야 하기 때문에 다음주나 그 다음주에 확실해질 것 같습니다. | | |
| **What to do next week:** ROS에서 사용할 package나 node를 조금 더 명시하고, 어떤 구조로 전체적으로 알고리즘을 짜 나갈지 조원들과 상의하는 것이 좋을 것 같습니다. 큰 그림이 먼저 확고하게 잡혀 있어야 나중에 코딩 작업을 할 때 짜임새 있게 일을 할 수 있을 것 같습니다. ROS 파트 조원들과 상의를 해봐야겠지만, lidar를 통해서 어떻게 하면 로봇 자체의 좌표값을 저장할 수 있을지, lidar를 통해서 받은 값을 자신의 2D 위치 좌표로 변환하여 subscriber node로 만드는 방법에 대해서 더 고민해보면 좋을 것 같습니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #3** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 4/2~4/8 |
| C:\Users\oprij\OneDrive\Current courses\ME400_창의적 시스템 구현\Progress Reports\tacking_ball_detection.jpgC:\Users\oprij\OneDrive\Current courses\ME400_창의적 시스템 구현\Progress Reports\tracking_ball_detection_rviz.jpg가장 큰 성과는 단연 OpenCV 파트의 tracking 코드와 ROS 파트의 ball\_detect node를 합친 것이었습니다. 서주환 조교님이 주신 힌트 대로, 각 코드에서 카메라로 인식하는 부분과 인식한 데이터를 변환하는 부분, 데이터를 전달하는 부분 등을 찾아서 나누었고, 주로 Tracking 코드의 공 인식 부분을 ball\_detect node에 가져와서 공을 인식한 것을 메시지로 전달할 수 있는 형태로 만들었습니다. 터미널을 열고 roscore, rosrun webcam webcam\_node, rosrun ball\_detection ball\_detection\_node를 차례로 실행한 결과는 다음과 같습니다.  C:\Users\oprij\OneDrive\Current courses\ME400_창의적 시스템 구현\Progress Reports\tracking_ball_detection.jpg오른쪽의 rviz 실행 결과는 빨간 공 하나만 잡아본 상태입니다. 이렇게 공을 영상으로 인식하기 위한 기반을 마련한 상황이고, 현재는 공을 가지러 가는 방법에 대한 논의가 진행중입니다. 공을 가지러 가는 방법에 따라서 코드를 변형할 수도 있습니다. 그 자리에서 사진을 찍게 하는 방법, 영상을 계속해서 찍으면서 tracking하는 방법 등을 생각 중이며, 더 창의적이면서도 효율적인 방법을 구상하려고 하고 있습니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #4** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 4/9~4/15 |
| ROS파트와 LabView 파트가 ‘합동으로 Xbox를 이용한 모터 구동’을 시도해보려고 하였습니다. ROS파트 학생들의 전공 시험 일정에 맞춰서 목요일 오후에 만나게 되었고, 만나서 우선 LabView 파트에서 현재까지 진행한 코드를 살펴보았습니다. 현재 있는 코드는 myRio를 통해서 모터를 구동할 수 있는 코드였습니다. Xbox 컨트롤러의 신호를 대신해서 PC와 연결된 키보드의 입력을 받았을 때는 바퀴의 방향을 원하는 방향으로 돌릴 수 있게 프로그래밍이 된 상태였습니다. 저희의 이번 주 목표는 Xbox 컨트롤러를 NUC에 연결, NUC와 myRIO를 연결하여 Xbox에서 입력한 신호를 NUC에서 받아서 myRIO로 보내주면 myRIO가 모터를 제어하도록 하는 것이었습니다. 하지만 계속 신호 송/수신에 문제가 있는 것 같아서 LabView 조교님인 이범준 조교님께 연락을 드렸습니다. 조교님이 오셔서 같이 오류를 고쳐보려고 하는 과정에서, 우선 ROS파트에서의 데이터 송신 문제와 LabView의 수신 문제는 없다는 결론이 나왔습니다. 문제는 크게 두 가지가 있었는데, 첫째, 학생들에게 배포되었던 LabView 코드에서 데이터 변환이 잘못되고 있었다는 점, 그리고 둘째, Xbox 버튼들에 대한 데이터와 LabView 출력값의 mapping이 잘못되었다는 점이었습니다. ROS에서는 Xbox 버튼에 대한 데이터를 24개의 float값을 가진 array로 보내는데, LabView에서 이를 변환해줄 때 문제가 있는 것 같다고 조교님께서 확인 후 말씀해주셨습니다. 또한, Xbox에서 특정 버튼을 눌렀을 때 LabView에서는 다른 버튼으로 잘못 인식하는 오류가 있었는데, 이는 학생들이 직접 해결하면 될 것 같다고 하셨습니다. 결론적으로 목표한 바는 이룰 수 없었지만, 코드의 구성과 작동 메커니즘에 대한 이해를 도울 수 있었던 시간이었고, 고쳐진 코드가 다시 배포되면 빠른 시일 내로 이 일을 다시 진행할 수 있을 것이라고 생각합니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #5** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 4/16~4/22 |
| C:\Users\oprij\OneDrive\Current courses\ME400_창의적 시스템 구현\Progress Reports\IMG_0285.jpg지난주에 문제가 있었던 LabView 코드가 고쳐졌습니다. Xbox 컨트롤러를 사용해볼 수 있게 된 상황이라서 급하게 모터 구동과 진행 방향 테스트를 하기 위해서 창시구실에 비치 되어있던 알루미늄 프로파일로 임시 차체를 제작했습니다. 그 후 TCP/IP 통신을 이용해 NUC와 myRIO를 연결하고, xbox 컨트롤러로 임시 차체를 조작해 보았습니다. 앞, 뒤, 양 옆으로 이동할 수 있는 상황이 되었고, 향후 LabView 에서의 mapping을 통해서 대각선 방향의 이동과 제자리 회전까지 구현할 예정입니다.  ROS파트에서 할 수 있는 부분 이외에도 회로를 연결하기 위한 납땜을 시작했습니다. 배터리 전압을 myRIO, NUC, 다이나믹셀, dc 모터가 나눠 받아야 하는데, 현재는 NUC, myRIO, 그리고 다이나믹셀에 연결되는 부분을 임베디드 보드에 납땜으로 연결한 상황입니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #6** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 4/23~4/29 |
| ROS 파트 일원 중 류준일 친구의 개인 PC로 마스터/슬레이브 연결에 성공했고, 배포 코드 중 data\_integrate 코드를 수정해서 ball\_detect 노드에서 공을 인지했을 때 그 정보를 이용하여 공을 따라가는 코드를 만들었습니다. ball\_detect 노드 실행 후 data\_integrate 노드를 실행하면, 우선 공이 없을 때는 반시계 회전, 공이 카메라 프레임 안에 들어오면 공이 프레임 중앙에 들어오도록 좌/우로 이동 후 전진하도록 만들었습니다. LabView 담당 학생이 대각선 방향 진행과 제자리 회전을 할 수 있도록 만들어 뒀기 때문에 가능했습니다. 현 시점에서 문제점은, myRIO에서 신호를 받아올 때 딜레이가 너무 심해서 공을 프레임의 중앙에 오게 하려고 좌/우로 이동을 하면 지나치게 많이 이동한다는 점입니다.  모터 구동 및 tracking test를 한 후, 이제 전체적으로 ROS의 알고리즘을 짜 보았습니다. 조교님께서 복잡할 것 같다고 귀띔을 해 주셨으나 현재로서는 어느 부분이 얼마나 복잡할지 잘 이해가 되지 않고, 일단 진행해 보고 나중에 수정해도 괜찮다고 하셔서 조교님 말씀대로 일단 이러한 구조로 프로그래밍을 할 예정입니다. Mapping이 부정확하고, 어렵기만 하다는 말이 많아서 lidar를 이용한 mapping은 포기하고 camera만 이용해서 tracking을 할 예정입니다. 위는 tracking만 이용해서 모든 알고리즘을 짠 것인데, 만약 mapping이 필요하다고 생각되고, 해야 할 때는 다른 계획이 있습니다. Mapping을 하게 된다면 path planning을 하는 편이 공을 찾아서 가져오는 데 있어서 매우 효율적일 것이라고 생각했고, research를 통해서 저희가 코딩을 통해서 구현할 수 있을 법한 path-generation 방법을 찾아냈습니다. 왼쪽 그림과 같이 전체 필드를 격자로 나눈 후에, 각 격자에 목표물, 장애물, 시작점 등을 구분할 수 있도록 숫자를 부여합니다. 그 후 목표물과 시작지점 사이에 숫자를 순차적으로 부여하면, 숫자가 1씩 줄어드는 칸으로 계속 이동했을 시에 최단 거리로 장애물을 피해서 목표물로 이동할 수 있습니다.  다음주에는 openCV 파트에서 calibration이 완료되면 tracking test를 한 번 더 해볼 예정이고, 코드 수정을 거쳐서 보다 정확한 tracking을 구현해보려고 합니다 다른 코드들도 조금씩 보면서 저희가 알고리즘 상에 어떤 문제가 있는지 파악하고, 발표 준비도 조금씩 도울 예정입니다. | | |
| **Progress Report #7** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 4/30~5/6 |
| 발표 준비를 우선적으로 진행했습니다. 첫 번째 발표 이후로 여태까지 진행해왔던 ROS 파트에서의 progress를 정리해서 ppt로 만들었고, 발표 자료를 만들고 발표를 하기로 했던 조원 Ashar에게 보냈습니다.  서스펜션을 제작하는 데에 시간을 꽤 많이 사용했습니다. 서스펜션을 연결하기 위해서 기존의 차체의 틀을 이루는 알루미늄 프로파일에 추가적으로 프로파일을 덧붙이고, 그 구조에 서스펜션과 메카넘 휠을 연결했습니다. 하지만, 기존에 H자형 프로파일로 xbox 컨트롤을 해 보았을 때와는 다르게, 바퀴가 계속 헛도는 현상이 일어났습니다. 발표 전날까지 계속해서 같은 구조 안에서 나사를 조이고 풀고, 조금 더 정확하게 바퀴를 연결해보는 작업을 계속 했었는데, 결국 최종적으로는 실패하였고 현재 가지고 있는 서스펜션을 개선하거나 서스펜션을 사용하는 것을 포기할 필요가 있다는 결론을 내렸습니다. 결론을 내린 이유 세 가지는 1. 서스펜션을 연결하기 위한 시간 소비가 너무 많다, 2. 서스펜션을 연결했을 때 얻을 수 있는 효과가 생각보다 크지 않다, 3. 현재 가지고 있는 서스펜션은 제대로 된 계산을 통해서 만든 것이 아닌 용수철 상수와 damping coefficient가 적절하지 않은 서스펜션이다 였습니다.  그래서 우선은 서스펜션을 제외하고 차체의 구조 자체도 사각형에서 H자형을 기반으로 한 모양으로 바꾼 후 다시 한 번 모터 구동을 해 볼 생각입니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #8** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 5/7~5/13 |
| 발표가 있은 후로부터 저희 조의 진행 상황이 많이 뒤쳐져 있다고 판단하였고, 남은 기간을 잘 활용하기 위해서 조원 전체가 교수님과 함께 모여서 각 파트의 각 task에 대한 구체적인 데드라인을 만들었습니다.  차체의 기본형을 이루는 프레임을 사각형 모양보다 안정적인 H자형 모양으로 바꿨고, 거의 다 만들어져 있던 회로를 마무리했습니다.  소프트웨어 쪽에서는 이제 ball\_detect 코드가 빨간 공과 파란 공 둘 다 인식하도록 다시 바꿔 놓았습니다. (원래 가능했으나, 오픈씨비 파트와 상의 후 코드를 변경하는 과정에서 편의상 한 가지 색깔의 공만 인식할 수 있도록 바뀌어 있었고, 여태까지 tracking 시행은 변경된 코드로 해 왔었습니다. ) 점점 알고리즘을 구현하고 적용해보는 단계에 다다르면서, actuator를 담당하고 있는 랩뷰 파트와 상의를 많이 하게 되었습니다. 랩뷰 파트에서는 현재 있던 delay 문제가 해결되었습니다. 저희의 알고리즘상 현재 ROS에서 필요한 것은 시간을 관리하는 방법을 터득하는 것입니다. 특정 함수가 특정 시간동안 실행되도록 하는 방법을 모르기 때문에 우선 공을 획득하는 알고리즘을 랩뷰 파트에서 만들어 놓았고, 실제로 로봇에서 구현되는 것도 확인하였습니다.  현재 ROS파트에서 가지고 있는 문제는 1. 메소드가 실행되는 시간을 코드로 제어하는 방법 2. Path finding(lidar를 이용한)이 필요한가 3. Path finding을 사용하지 않았을 때, 파란 공까지 가는 경로에 빨간 공이 있을 경우 어떻게 알맞은 공을 찾아 갈 것인가 입니다. 다음주에는 위 문제들을 해결하고, 차체가 완성되면 계속해서 시행해보면서 코드를 보완하는 과정까지 할 계획입니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #9** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 5/14~5/20 |
| 함수가 실행되는 시간을 코드로 제어하는 방법을 생각해내게 되었습니다. 지금 시점에서는 actuator를 다루는 Labview와 ROS가 같이 생각을 해야 했었는데, ROS 코드만으로는 해결하기 어려운 문제인 것 같아서 Labview로 시간 제어를 돌리기로 결정했습니다. ROS 코드에서는 특정 명령과 그 명령을 실행해야 하는 시간을 Labview로 보내도록 하였고, Labview에서 그 시간 동안 명령을 시행하도록 했습니다.  Lidar를 이용하기에는 사용 방법을 익히고 코드에 적용시키기에 시간이 너무 부족한 것 같아서 우선 가지고 있는 웹캠으로 모든 navigation을 해결해 보기로 결정했습니다. 지난주에 가지고 있던 문제 중 하나가 파란 공을 집으러 가는 경로상에 빨간색 공이 있으면 어떻게 해야하는가 였는데, 파란색 공 감지 후 빨간색 공을 확인, 만약 빨간색 공이 프레임 상으로 파란색 공과 가까이 있는데 파란색 공보다 더 가까이 있으면 파란색 공과 비교한 위치에 따라서 특정 방향의 대각선으로 움직이도록 하였습니다. 일정 시간(아직 정하지 못하였습니다.)동안 대각선으로 이동하여 빨간 색 공을 피한 후에 다시 파란색 공으로 가는 방법을 선택했습니다.  하드웨어가 계속 문제가 생겨서 아직까지 코드 테스트를 못하고 있습니다. 서스펜션이 완료가 되어서 바퀴가 땅에 잘 붙는 것이 확인되었고, 움직임도 훨씬 좋아졌습니다. Ball picking part에 대한 수정이 계속 이루어지고 있고, 차체도 아직까지 확정되지 않았기 때문에 직접 알고리즘이 시행되는 것을 보려면 시간이 더 필요할 것 같습니다. | | |

|  |  |  |
| --- | --- | --- |
| **Progress Report #10** | | |
| **20150721 조인준 (adv prof. 김형수)** | **ROS** | **Date**: 5/21~5/27 |
| Lidar 없이 어떻게 하면 공을 가지고 바구니로 돌아갈 수 있을까 계속 고민을 했습니다. 고민의 결과로 나온 방법들은 여러 가지가 있는데, 모두 부족한 부분이 있어서 계속 고민 중입니다. 지금 ROS 코드의 문제점은 한 번의 루프에 한 가지 명령만 내릴 수 있다는 점인데, 현재 생기는 대부분의 한계점이나 문제는 이것 때문인 것 같습니다. 그래서 월요일에 조교님께 조언을 얻고 코드에 대한 부분을 다시 생각해야 할 것 같습니다. 또한 코드를 실행하는 데에 있어서 루프는 계속 0.025초 간격으로 돌고 있는데, 실행해야 하는 명령은 1초가 넘는 시간이기 때문에 가끔 데이터가 쌓이는 듯한 문제도 보입니다. 움직임 상 가장 큰 문제점은 1. 가끔 정해놓은 시간보다 더 긴 시간 동안 명령을 실행한다 2. 명령을 두 번 실행한다 3. 위치를 갑자기 잘못 잡아서 무한 루프를 돈다 4. 파란색 공이 50cm 정도로 일직선상에 있을 때 인식을 잘 하지 못하는 네 가지 문제점이 있습니다.  Blade 용으로 주문했던 ax 모터가 고장이 나서 배송이 온 mx 모터로 바꿔야 합니다. 하드웨어는 계속 바뀌고 있고, 카메라 위치도 로봇의 중앙에서 blade 옆으로 이동했습니다. | | |

**마지막 주:**

ROS 코드를 대대적으로 바꿨습니다. 매번 루프를 0.025초마다 돈다는 특성을 이용해서 시간을 조절하는 방법을 생각해냈고, 그 때문에 코드를 대대적으로 바꿨습니다. 루프마다 ROS에서 LabView로 보낼 수 있는 커맨드는 하나밖에 없는데, 그 커맨드를 보내는 시간을 조절하기 위해서 조건 별로 커맨드를 보내는 루프 수를 조절했습니다. 그 덕분에 시간을 데이터가 밀리는 딜레이 없이 완전하게 컨트롤 할 수 있게 되었고, 각 움직임에 필요한 시간을 원하는 대로 완벽히 조정했습니다. 이전주에 있었던 네 가지 문제 중 첫 세 가지 문제가 모두 해결이 되었습니다. 네 번째 문제는 앞선 세 문제가 야기했던 문제이기 때문에, 시야 확보를 위해서 하나의 공을 줍게 한 후 왼쪽 뒤 대각선으로 이동하게 하는 것으로 해결했습니다. 다만 데모 시간을 적절히 활용하기에 LabView 연결 문제와 다른 장비 문제가 많아서 주차에 신경을 충분히 쓰지 못했습니다.

실제 데모 장소에서 많이 연습해보지 못한 상태로 최종 데모 당일이 되었습니다. 당일 저희 조는 두 번의 시행 모두 파란색 공을 빨간색 공으로부터 완벽히 분리해내서 세 개 모두 줍는 데에 성공했으나, 두 번 모두 주차에서 실패했습니다. 공을 바구니 쪽으로 가지고 와서 바구니에 넣는 과정에서, 방향과 거리에 약간의 착오가 있어서 바구니에 정확히 공을 넣지 못했습니다. 원인을 생각해본 결과, 1. 실제 데모 장소에서 시행해볼 기회가 별로 없었기 때문에 당연하게도 실제 평가에 사용되었던 바구니에 공을 넣는 연습을 많이 하지 못했던 점, 2. 그렇기 때문에 바구니에 놓여있는 공 사이의 거리를 잘 몰랐고, 거리 계산이 잘못되었던 점을 꼽을 수 있었습니다. 굉장히 아쉬웠지만, 다른 조에 비해서 진도가 많이 밀렸음에도 마지막까지 포기하지 않고 공을 다 획득할 수 있었다는 점은 만족할 만한 점이라고 생각합니다. 결국 점수에 들어가는 것은 공을 넣는 개수였기 때문에, 창시구 2에서는 결과에 조금 더 집중해서 회수 후 마지막에 목표물을 목표 장소에 놓는 것에 조금 더 집중을 해야겠다는 생각도 들었습니다.

발표 준비: ROS 에서뿐만 아니라, 저희 조는 마지막 주에 하드웨어가 완성되었기 때문에 열에 대한 분석도 마지막 주가 되어서야 할 수 있었습니다. 데모 장소에서 연습을 하면서 열화상 카메라를 찍어서 온도를 분석해 보았고, 그 결과 평가 기준이 되는 섭씨 70도 한참 아래의 온도가 나왔기 때문에 열을 낮추기 위해서 엄청난 노력을 할 필요는 없다는 결론이 나왔습니다. 그래서 passive 핀을 다는 것으로 마무리를 하였습니다.

개인적으로 창시구 1은 저에게 대학 생활을 하면서 가장 많은 것을 느끼게 해 준 과목입니다. 저는 이론을 배우는 것도 좋아하지만, 배운 이론을 이렇게 직접 활용해볼 수 있는 기회가 있는 것이 굉장히 좋다고 생각합니다. 직접 해 보았을 때 비로소 그 이론을 보다 더 완전하게 이해할 수 있고, 그렇기 때문에 창의적 시스템 구현이라는 과목이 정말 힘들지만 또 기계공학과 학생들에게 필수적인 과목이라고 느꼈습니다. 다만, 시간이 조금 더 많았더라면 더 많은 것을 적용시켜보고 그 과정이 더 좋은 복습이 될 수 있지 않았을까 하는 아쉬움이 남습니다. 창의적 시스템 구현 2에서 그 아쉬웠던 부분을 스스로 채울 수 있고 기계공학에 대한 이해도도 더 높일 수 있는 기회가 있었으면 좋겠습니다.