***Robotis-Turtlebot3***

***Professor : 박순용***

***Student Name : 이재황, 홍준성***

***Student Number: 2013105078, 2014097091***



**< Table of contents >**

1. **INTRODUCTION**
   * 1. Brief Introduction to the entire scenario
2. **RESEARCH CONTENTS**
   * 1. Deeper Look
3. **RESULTS OF RESEARCH**
   * 1. Summary
        1. **INTRODUCTION**

**< 1 *. Brief Introduction to the entire scenario>***

본 프로젝트에서 저희 조는 Follower 를 구현하였습니다. Follower 는 조원들 간의 브레인스토밍을 통해 선정하였습니다. 프로젝트 주제의 테마는 강의에서 배운 ros-turtlebot 프로그래밍을 기반으로 가장 창의적이고 흥미로운 주제를 찾도록 고심하였습니다. 하지만 로봇에 대한 접근이 처음이었던 저희는 아쉽게도 다른 특별한 아이디어를 떠올리지 못하였습니다. 그리하여 저희들은 교수님께서 추천해주신 예시들 중 하나인 Following human을 선정하여 진행하게 되었습니다. 본 프로그램의 전체적인 흐름은 아래와 같습니다.

1. 임의 개수의 점들의 평균 거리 구하기(최소 거리인 인덱스을 구하기 전, 노이즈를 제거하기 위함)

2. 최소 거리의 인덱스 구하기

3. 인덱스에 따라 터틀봇의 회전 방향 정하기

4. 목표 방향에 도달 시

5. 카메라를 통한 다리 인식

6. 다리의 너비에 따라 거리 인식, 다리와의 거리 유지

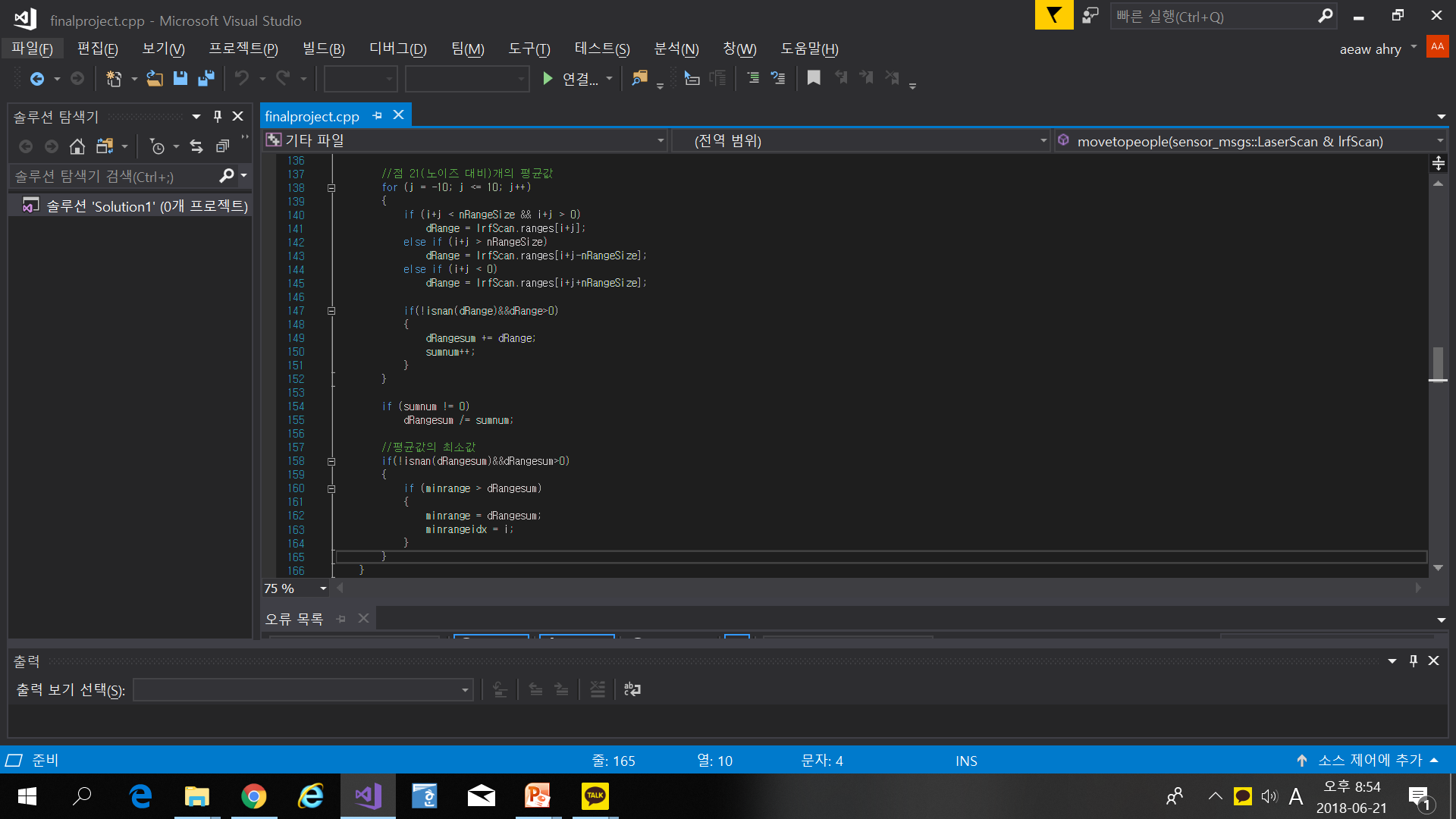
6-1. 다리가 인식되지 않는다면, 다음으로 가까운 인덱스를 향해 회전

먼저, Lidar 스캔을 통해 읽어오는 점들을 통하여 그 중 임의의 개수의 점들을 골라 거리의 평균값을 구하였습니다. (이 임의의 개수는 후에 실제환경에서의 구동을 통하여 오차를 최적화한 결과 저희는 21개를 가지고 하게 되었습니다.) 그 후 평균값들의 최소값을 구하여 그 값의 인덱스가 터틀봇 왼쪽이면 좌회전 오른쪽이면 우회전을 하도록 프로그래밍 하였습니다. (각도에 따라 회전속도를 다르게 해보았습니다.) 그 후 rpi 카메라를 통해 다리 인식 기능을 구현하려 했으나, rpi 카메라와 마스터 pc와의 통신 속도, 구현 난이도 등의 이유로 이는 미완성으로 남았습니다. 그래서 거리 유지는 카메라 대신 Lidar로 읽어온 거리를 그대로 이용하는 방향으로 진행 되었습니다. 터틀봇의 전방 60도 각도에 가장 가까운 물체가 있으면 거리가 25cm 이상이면 직진, 25cm 미만이면 후진하도록 하여 25cm를 유지하도록 하였으며, 전방에 가장 가까운 물체가 없을 경우에는 linear로 이동하지 않도록 하였습니다.

**2. RESEARCH CONTENTS**

**< 2 -ⅰ*. Deeper Look >***

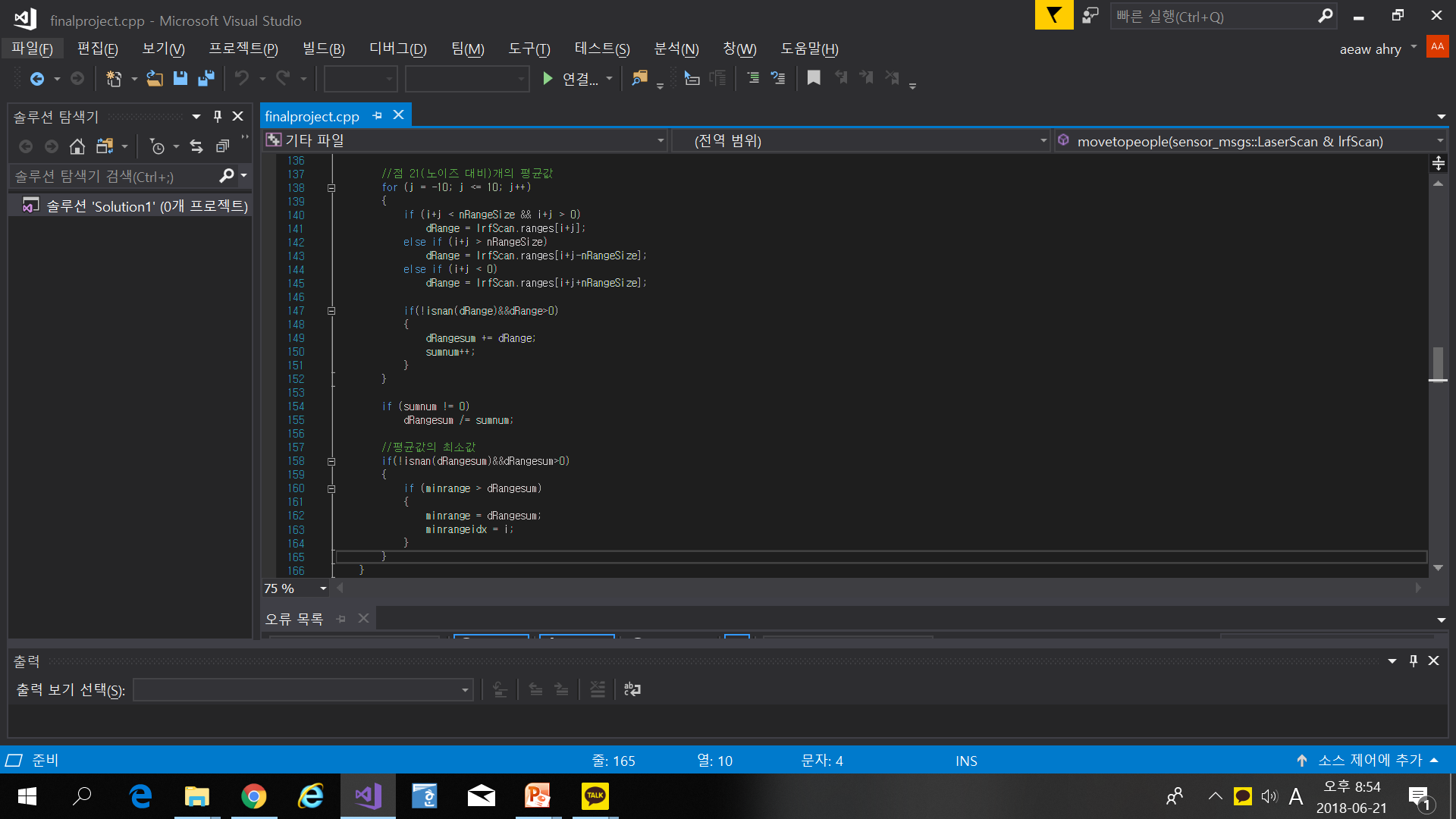
*(본 장에서는 프로그램 구동에 핵심적인 요소에 대해 기술합니다. 세부 사항은 소스코드 파일을 첨부합니다.)*

1. 임의의 점들을 통한 평균 거리 구하기 

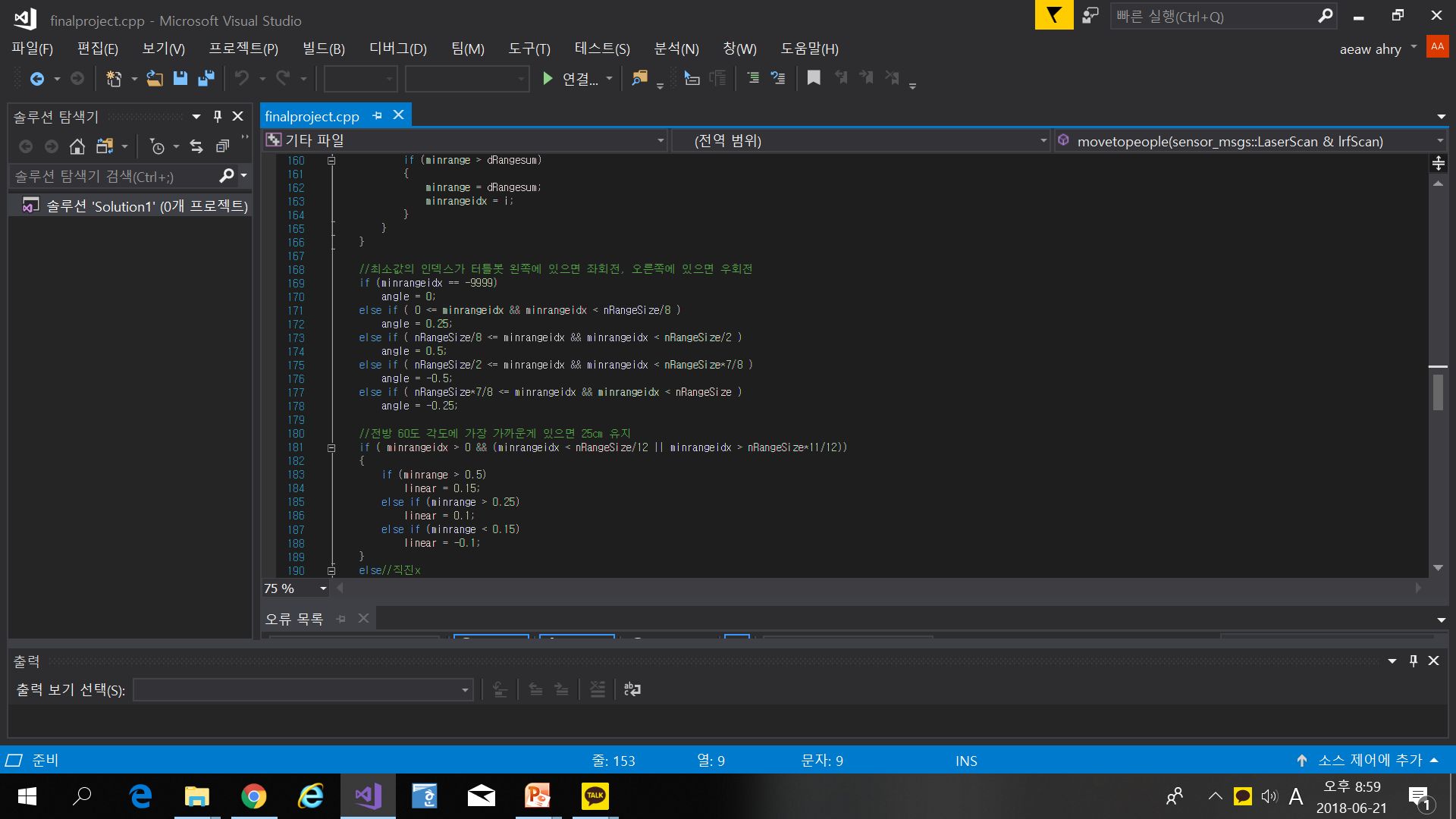
실제 환경에서의 구동을 통하여 21개가 저희에게 적절하다고 판단하여 21개로 구현하였습니다.

Lidar 스캔을 통해 읽어온 거리가 유효한 경우에만 합산, 이를 합산된 횟수만큼 나누어 21개 점의 평균 거리를 측정하였습니다.

1. 평균 거리의 최소값 구하기

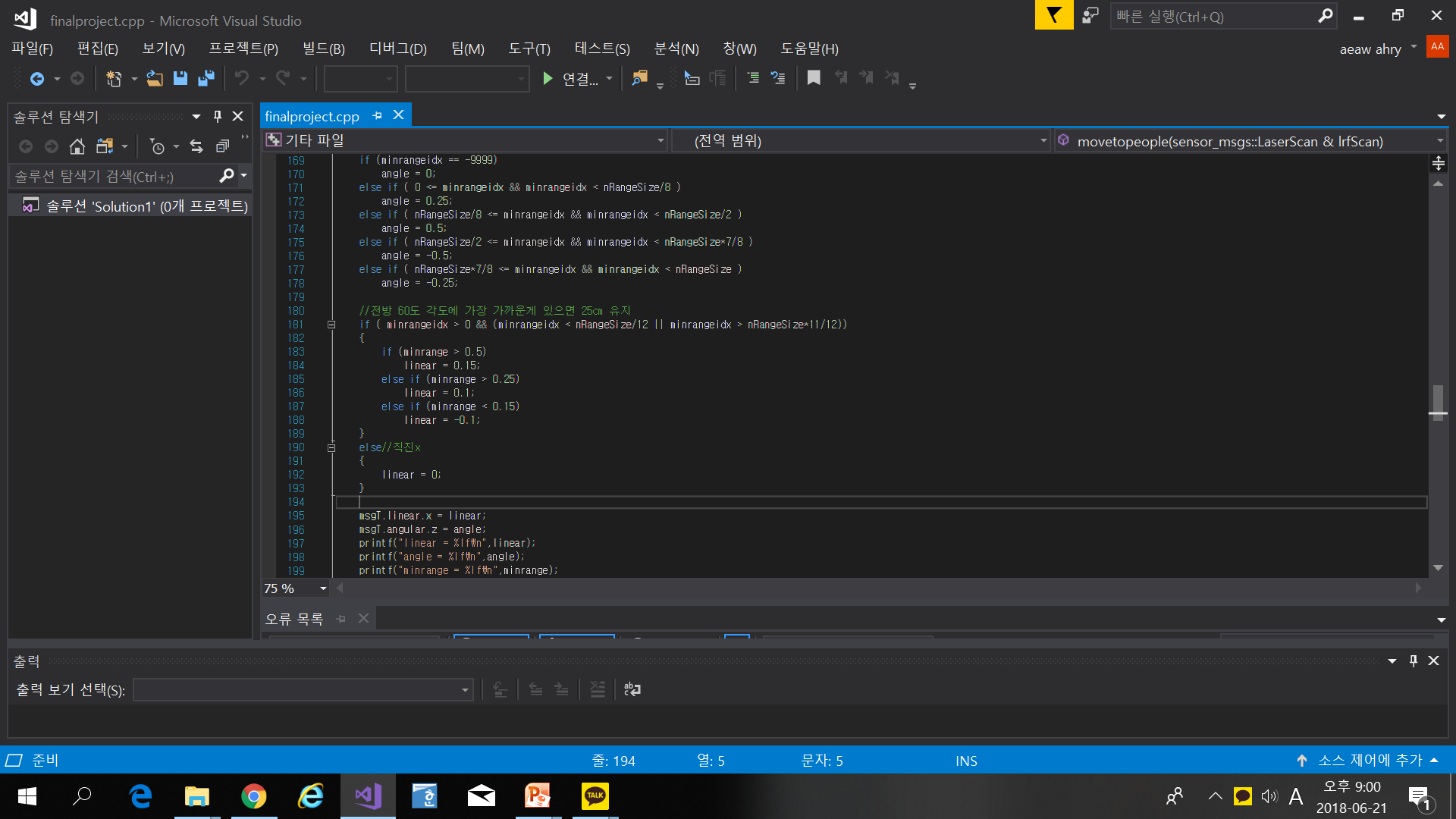


위의 점들 로부터 평균 거리의 최소값을 구하여 가장 가까운 인덱스를 구했습니다.

1. 최소값에 따른 터틀봇 회전 방향 정하기
2. 가까운 물체 쪽으로 회전하기 위해 최소값의 인덱스를 통해 대략적인 각도를 구하여 터틀봇의 왼쪽에 있으면 좌회전, 오른쪽에 있으면 우회전을 하도록 하였습니다.

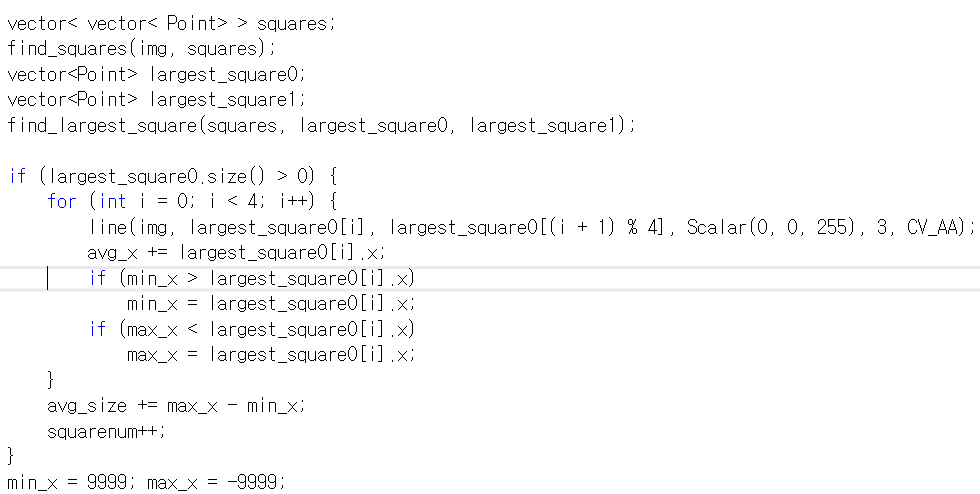
실제 구현을 통하여 값들을 수정하였습니다.

1. 목표 방향에 도달 시



목표 위치를 물체와 붙어버리면 안되기 때문에 전방 60도 각도에 가장 가까운 물체가 있으면 거리가 25cm 이상이면 직진, 25cm 미만이면 후진하도록 하여 25cm를 유지하도록 하였으며, 전방에 가장 가까운 물체가 없을 경우에는 linear로 이동하지 않도록 하였습니다.

1. 카메라를 통한 다리 인식



교수님의 예제를 토대로 카메라를 통하여 읽어오는 정보를 통해 사각형들을

인식할 수 있도록 구현하였으나 아쉽게도 이 기능은 실제 환경에서 제대로 동작하지 않았습니다.

1. **RESULT OF RESEARCH**

**<3 -ⅰ*. Summary >***

본 프로그램에서는 평소 결정론적 프로그래밍을 위주로 하였던 저희에게 수많은 독립변인들이 실제 상황에서 적용된다는 점을 일깨워 주었습니다. 그러한 어려움에도 불구하고 실제로 저희가 로봇을 만들고 작동하게 하는 과정에서 많은 배움이 있었으며, 이러한 로봇 관련 프로그래밍에 흥미를 얻어 관련 자료들 또한 저희 스스로 더욱더 찾아보며 더욱 심도 있게 배우게 되는 계기가 되었습니다. 더불어 직접 프로젝트를 진행하며 구현하는 과정에서 ros의 특성에 대해 더욱 깊이 이해할 수 있는 계기가 되었다고 생각합니다. 차후 로봇 관련 프로그래밍은 물론 앞으로의 모든 프로그래밍을 하는 과정에도 이러한 경험이 더욱 심도 있는 프로그램 구현 및 이론의 이해에 도움이 될 수 있을 것이라 생각합니다.

*< 끝 >*