시뮬레이션을 활용한 교내 드론 배송시스템 구현

김성근, 문주현, 서민석, 이태준, 진준형

A+

발표자 : 진준형
군사탐방 1조 "날틀"

목차

동기

- 국내 드론 배송시스템이 활성화 되어있지 않은 이유?

  1. 드론의 물리적 한계

  2. 법적 규제의 한계

- 시스템을 구축하고 있다면 빠르게 상용화 될 수 있음

- 향후 공군 기지내(비행단, 레이더기지, 방공포부대 등) 군수 물자 수송시스템 구축에도 도움이 될 수 있음

기존의 HUB 선정 알고리즘의 3차원 확장

Ⅰ. 장애물 회피 알고리즘

그림 1.1은 2차원에서 장애물을 회피하는 과정

→ x,y 좌표 만으로 장애물 회피 알고리즘 설계

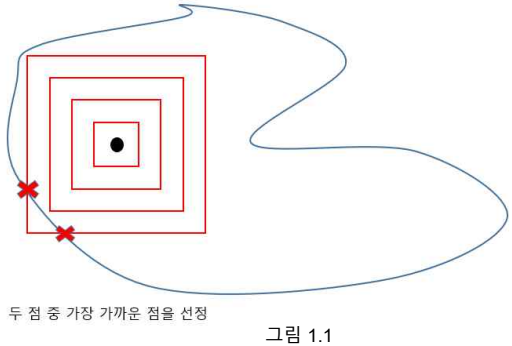


그림 1.1

그림 1.1은 3차원에서 장애물을 회피하는 과정

→ 그림 1.1에서 선정한 HUB의 Z좌표를 읽은 다음 해당 좌표를 최종 HUB로 선정

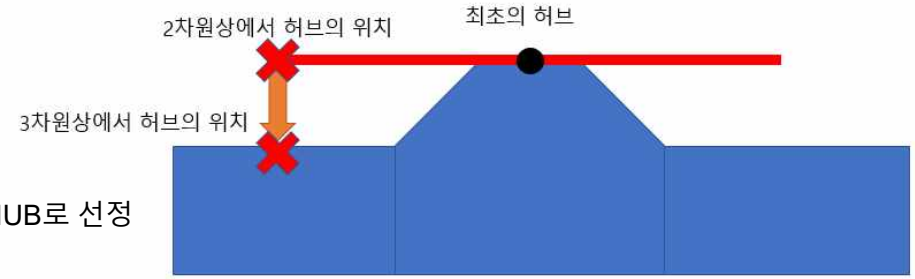


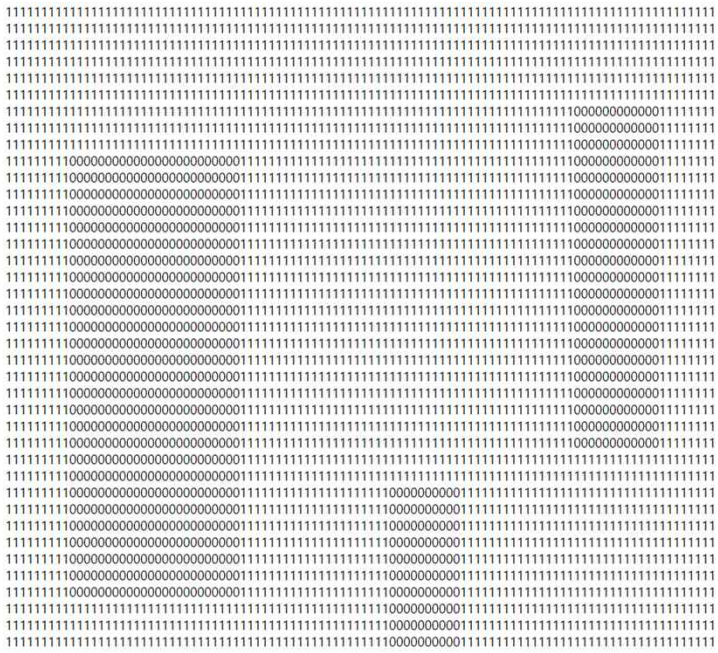
그림 1.2

기존의 HUB 선정 알고리즘의 3차원 확장

Ⅱ. 맵 데이터의 확장

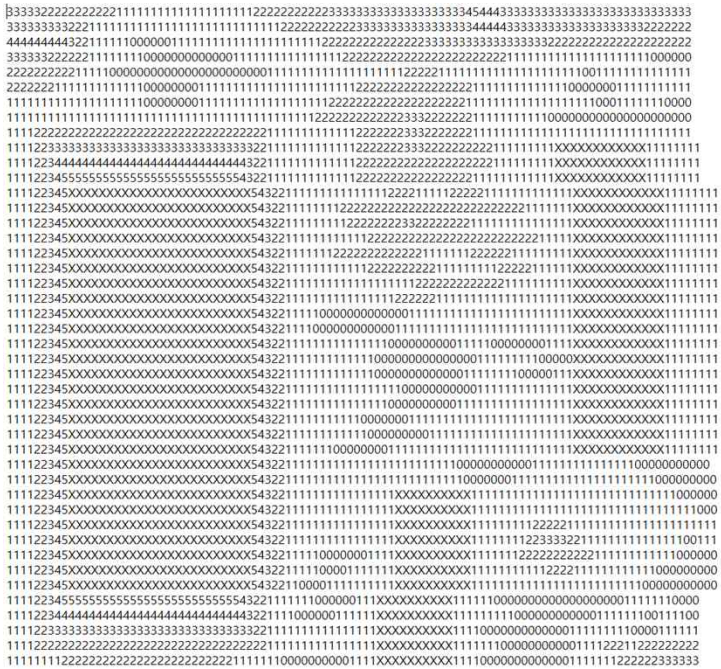
기존의 2차원 맵 데이터

드론이 착륙할 수 있는 지점을 1, 착륙할 수 없는 지점을 0으로 표기



3차원 맵 데이터

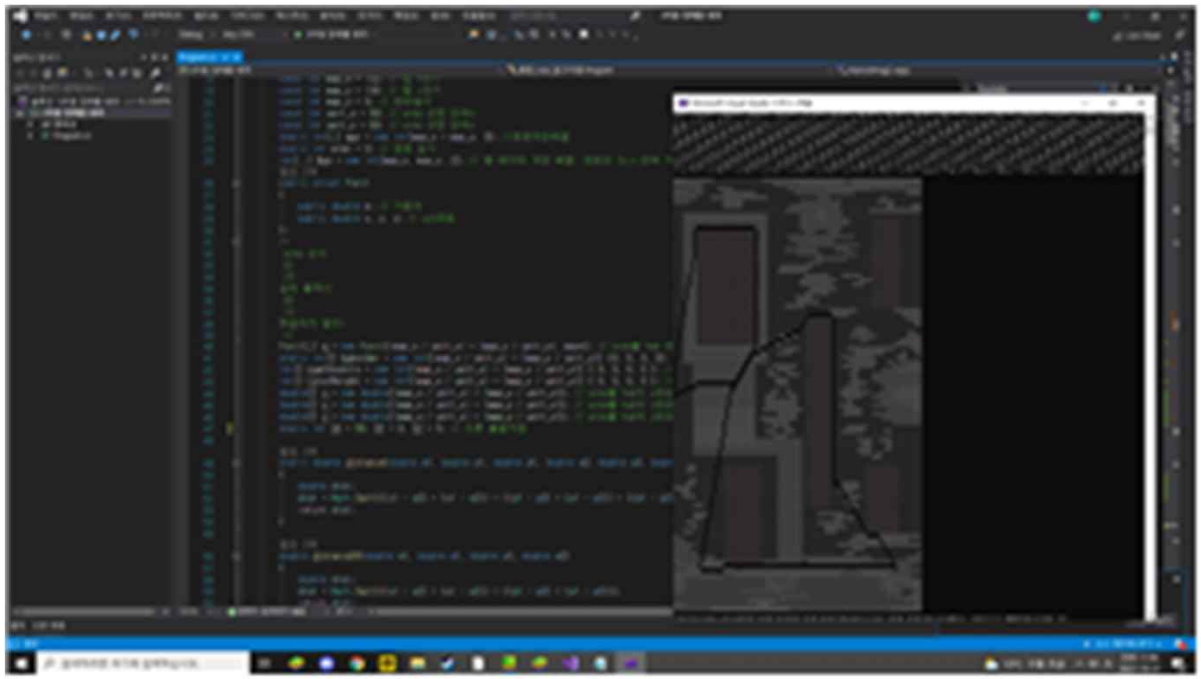
- 착륙 가능성을 문자(O,X)로 표기
- Z축에 관련된 높이를 0~5로 표기



기존의 HUB 선정 알고리즘의 3차원 확장

Ⅱ. 맵 데이터의 확장

최종 완성 된 맵 데이터 모습



A* 알고리즘

A* 알고리즘 이란?

주어진 출발 꼭짓점에서부터 목표 꼭짓점까지 가는 최단 경로를 찾아내는 그래프 탐색 알고리즘

→ **현 위치에서 드론이 이동할 매우 가까운 다음 지역(1m 이내)까지의 비용 + 다음 지점에서 목표지점까지의 비용을 합한 값을 구한 후 이 값들 중 비용이 가장 적은 경로를 선택함**

매우 가까운 지역이기에 비용을 구하기 쉬움

지형에 대한 정보가 거의 없어 정확한 값을 구할 수 없다

→ Huristic 함수를 이용하여 예측

A* 알고리즘

6 6 16 16 6 16 16 6 16 16 6 16 16 16 6 16 16 6 16 16 6 16 16 6 16 16 16 6 16 16 6 16 16 6 16 16 16 6 16 16 6 16 16
23 23 23 23 23 23 23 23 26 23 23 26 23 23 26 23 23 26 23 23 23 26 23 23 26 23 23 26 23 23 26 23 23 23 26 23 23 2
26 16 16 26 16 16 26 16 16 16 26 16 16 26 16 16 26 16 16 16 26 16 16 26 16 16 26 16 16 26 16 16 16 26 16 16 26
6 3 3 3 6 3 3 6 3 3 6 3 3 6 3 3 3 6 3 3 6 3 3 6 3 3 6 3 3 3 6 3 3 6 3 3 6 3 3 6 3 3 3 6 3 3 6 3 10 1
10 10 20 10 10 20 10 10 20 10 10 10 20 10 10 20 10 10 20 10 10 20 10 10 10 20 10 10 20 10 10 20 10 10 20 10 10
23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 2
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 23

경로 알고리즘을 통해 도출된 드론 배송의 최단 경로를 표현한 텍스트 파일

3진법을 활용해 데이터를 표현하고자 했음

EX. 6의 경우 3진법으로 나타내면 020 이고,
이는 x축방향으로 -1, Y축방향으로 1, Z축 방향으로 -1만큼 드론을 이동시키라는 것

유니티 내 아주대학교 모델링

지형 데이터를 삭제하고 모든 지형 고도를 Default값으로 설정

→ 건물 간 부자연스러움 해결

→ 맵 데이터 함수 작성에도 용이

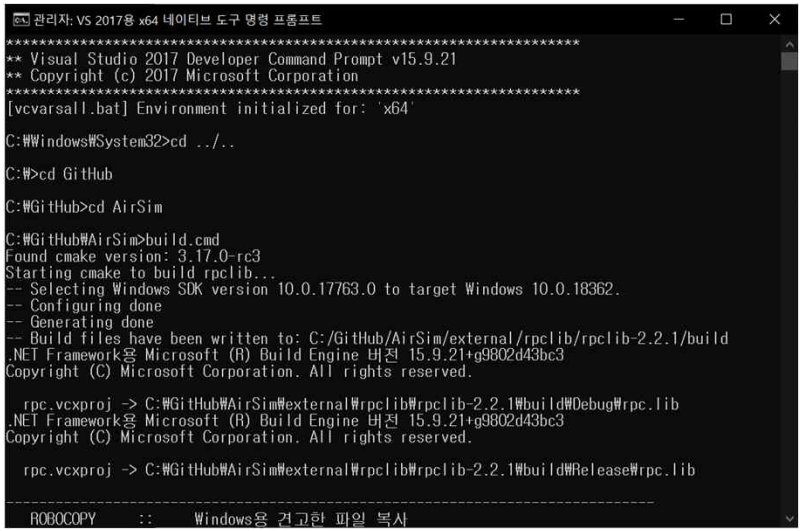
최종 완성된 아주대학교 맵 개관

Unity 내 Airsim Building

Airsim이란? - Aerial **I**nformatics and **R**obotics **Sim**ulation

→ 가상 환경을 제작하고 자율주행 시스템에 관한 딥러닝 및 인공지능을 테스트할 수 있는 마이크로소프트 사의 오픈소스 플랫폼

개발한 Hub 선정 알고리즘을 토대로 Unity 내 아주대학교 맵에서
드론을 구동하기 위해 Airsim을 활용할 예정



← **Unity내에 Airsim을 빌드하는 모습**

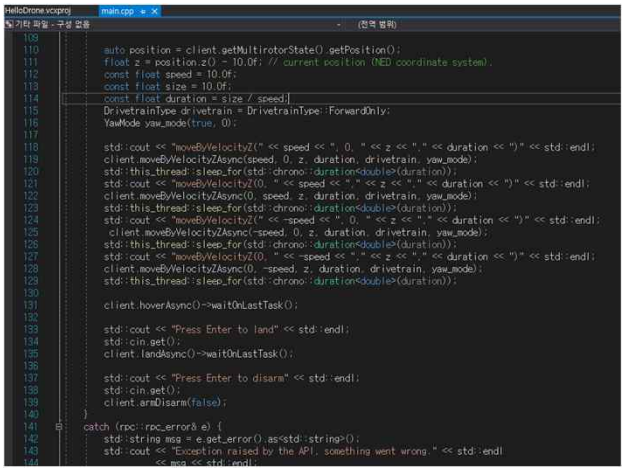
빌드 과정에서 오류가 상당 부분
발생하였지만, 결과적으로 빌드 성공

Unity 내 Airsim Building

API란? – Application Programming Interface

→ 응용 프로그램에서 사용할 수 있도록 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스

Unity내에 드론 제어 API가 정상적으로 실행되는 모습



```
        auto position = client.getMultirotorState().getPosition();
        float z = position.z() - 10.0f; // current position (NED coordinate system).
        const float speed = 10.0f;
        const float size = 10.0f;
        const float duration = size / speed;
        DrivetrainType drivetrain = DrivetrainType::ForwardOnly;
        YawMode yaw_mode(true, 0);

        std::cout << "moveByVelocityZ(" << speed << ", 0, " << z << "," << duration << ")" << std::endl;
        client.moveByVelocityZAsync(speed, 0, z, duration, drivetrain, yaw_mode);
        std::this_thread::sleep_for(std::chrono::duration<double>(duration));
        std::cout << "moveByVelocityZ(0, " << speed << "," << z << " " << duration << ")" << std::endl;
        client.moveByVelocityZAsync(0, speed, z, duration, drivetrain, yaw_mode);
        std::this_thread::sleep_for(std::chrono::duration<double>(duration));
        std::cout << "moveByVelocityZ(" << -speed << ", 0, " << z << "," << duration << ")" << std::endl;
        client.moveByVelocityZAsync(-speed, 0, z, duration, drivetrain, yaw_mode);
        std::this_thread::sleep_for(std::chrono::duration<double>(duration));
        std::cout << "moveByVelocityZ(0, " << -speed << "," << z << "," << duration << ")" << std::endl;
        client.moveByVelocityZAsync(0, -speed, z, duration, drivetrain, yaw_mode);
        std::this_thread::sleep_for(std::chrono::duration<double>(duration));

        client.hoverAsync()->waitOnLastTask();

        std::cout << "Press Enter to land" << std::endl;
        std::cin.get();
        client.landAsync()->waitOnLastTask();

        std::cout << "Press Enter to disarm" << std::endl;
        std::cin.get();
        client.armDisarm(false);
    }
    catch (rpc::rpc_error& e) {
        std::string msg = e.get_error().as<std::string>();
        std::cout << "Exception raised by the API, something went wrong." << std::endl
```

드론제어 API 코드

Unity 와 Airsim Building