

Laporan dokumentasi pengerjaan tugas 1 px4 Bayucaraka 2024

Farrel Ganendra | 5024231036

Tugas diawali dengan membuat folder workspace dan nabigasi kedalam folder src dengan command

```
mkdir -p tugasMagang-px4/src  
cd tugasMagang-px4/src
```

Kemudian fork repositori dan lalu clone repositori yang sudah di fork ke local. Di file manager, akan terlihat folder baru bernama “MagangBayu24-PX4Simulation”, saya masuk ke dalamnya, menampilkan file hidden, dan lalu meng-cut semua isinya ke luar folder dan lalu menghapus folder tersebut. Hal ini saya lakukan supaya hierarki folder dari workspace saya lebih rapi. Lalu kembali ke terminal dan membuat package ros2 dengan command berikut.

```
ros2 pkg create --build-type ament_cmake lintasan
```

Lalu pada file CMakeLists.txt, dibawah “find_package(ament_cmake REQUIRED),” tambahkan file file dependency yang diperlukan dengan command berikut

```
find_package(builtin_interfaces REQUIRED)  
find_package(eigen3_cmake_module REQUIRED)  
find_package(Eigen3 REQUIRED)  
find_package(geometry_msgs REQUIRED)  
find_package(px4_msgs REQUIRED)  
find_package(rclcpp REQUIRED)  
find_package(sensor_msgs REQUIRED)
```

Pada folder src didalam package, saya buat file baru dengan nama “lingkaran.cpp”. File ini untuk menaruh code publisher command ke drone kita. Kembali ke CMakeLists.txt lagi, berikan perintah untuk menambahkan executable, ament dependency, dan juga install supaya kita dapat menjalankan node yang kita buat dengan ros2 run.

```
add_executable(jalan_lingkaran src/lingkaran.cpp)  
ament_target_dependencies(jalan_lingkaran rclcpp px4_msgs)  
install(TARGETS jalan_lingkaran DESTINATION lib/${PROJECT_NAME})
```

Pada package.xml, lengkapi deskripsi, maintainer, dan juga license. Lalu tambahkan dependency dependency berikut.

```

<buildtool_depend>ament_cmake</buildtool_depend>
<buildtool_depend>eigen3_cmake_module</buildtool_depend>

<build_depend>eigen</build_depend>
<build_depend>ros_environment</build_depend>

<depend>builtin_interfaces</depend>
<depend>rclcpp</depend>

<depend>px4_msgs</depend>
<depend>geometry_msgs</depend>
<depend>sensor_msgs</depend>

<depend>launch</depend>
<depend>launch_testing</depend>
<depend>launch_testing_ros</depend>

<exec_depend>roslaunch</exec_depend>
<exec_depend>roslaunch</exec_depend>

```

Copy contoh publisher dari /TugasMagang-px4/src/px4_ros_com/src/examples/offboard/offboard_control.cpp ke lingkaran.cpp. Kemudian, hapus komen komen yang tidak diperlukan. Buat global variabel tipe float bernama “angle” dan assign 0.0. Variabel ini yang nantinya akan menjadi penentu posisi dan arah dari drone. Pada metode publish_trajectory_setpoint(), ubah posisi x menjadi 3 kali cos dari angle, posisi y menjadi 3 kali sin dari angle, dan posisi z menjadi -5.0. Kemudian untuk yaw ganti menjadi angle ditambah pi untuk mengantisipasi angle default dari drone (90 derajat).

```

void OffboardControl::publish_trajectory_setpoint()
{
    TrajectorySetpoint msg{};
    msg.position = {3*(float)cos(angle), 3*(float)sin(angle), -5.0};
    msg.yaw = angle + 3.14;
    msg.timestamp = this->get_clock()->now().nanoseconds() / 1000;
    trajectory_setpoint_publisher->publish(msg);
}

```

Kemudian pada bagian publik dari kelas OffboardControl. Ubah supaya drone di “arm” pada detik ke 0, dan lalu tambahkan kondisi untuk menambah angle sebesar 18 derajat setiap 1 detik. Lalu ketika angle sudah mencapai 360 derajat atau drone sudah melakukan 1 putaran.

Maka hentikan perubahan angle dan buat posisi z drone menjadi 0. Ternyata hal ini menimbulkan masalah, kita tidak dapat melakukan landing dengan hanya sekedar menurunkan drone ke posisi 0. Setelah saya membaca dokumentasi, ternyata saya perlu mengirimkan command untuk landing sendiri bernama VEHICLE_CMD_NAV_LAND yang merupakan enumerate dan anggota dari VehicleCommand di px4_msgs/msgs. Kita dapat mengirimkan pesan ini dengan fungsi publish_vehicle_command(uint16_t command, float param1 = 0.0, float param2 = 0.0). Berikut implementasinya

```
OffboardControl() : Node("offboard_control")
{
    offboard_control_mode_publisher_ = this->create_publisher<OffboardControlMode>("/fmu/in/offboard_control_mode", 10);
    trajectory_setpoint_publisher_ = this->create_publisher<TrajectorySetpoint>("/fmu/in/trajectory_setpoint", 10);
    vehicle_command_publisher_ = this->create_publisher<VehicleCommand>("/fmu/in/vehicle_command", 10);

    offboard_setpoint_counter_ = 0;


    auto timer_callback = [this]() -> void {
        if (offboard_setpoint_counter_ == 0) {
            this->publish_vehicle_command(VehicleCommand::VEHICLE_CMD_DO_SET_MODE, 1, 6);
            this->arm();
        }

        publish_offboard_control_mode();
        publish_trajectory_setpoint();

        if (offboard_setpoint_counter_ % 10 == 0 && offboard_setpoint_counter_ > 40) {angle += 0.314;}
        if (angle >= 6.28)
        {
            if (offboard_setpoint_counter_ < 400)
            {
                this->publish_vehicle_command(VehicleCommand::VEHICLE_CMD_NAV_LAND, 1, 0);
            }
            else if (offboard_setpoint_counter_ == 470){this->disarm();}
        }

        if (offboard_setpoint_counter_ <= 400){offboard_setpoint_counter_++;}
    };
    timer_ = this->create_wall_timer(100ms, timer_callback);
}
```

Kemudian tinggal source directory ros dan juga setup_local di semua 3 terminal, build package px4_msgs, lalu build package lintasan, jalankan agen dan simulator, dan jalankan node jalan_lingkar

Video hasil:  px4-lingkaran.mkv

<https://drive.google.com/file/d/1gcf195b7NBKaLaOLKkbJO2afYoaXpEN0/view?usp=sharing>