#ifndef FLAPPING\_CONTROLLER\_HPP

#define FLAPPING\_CONTROLLER\_HPP

#include <gazebo/common/Plugin.hh>

#include <gazebo/physics/physics.hh>

#include <rclcpp/rclcpp.hpp>

#include <std\_msgs/msg/float32.hpp>

namespace gazebo

{

class \_\_attribute\_\_((visibility("default"))) FlappingController : public ModelPlugin

{

public:

FlappingController() = default;

~FlappingController() override = default;

void Load(physics::ModelPtr \_model, sdf::ElementPtr \_sdf) override;

void OnUpdate();

private:

physics::ModelPtr model\_;

physics::JointPtr joint\_left\_;

physics::JointPtr joint\_right\_;

rclcpp::Node::SharedPtr node\_;

rclcpp::Subscription<std\_msgs::msg::Float32>::SharedPtr sub\_;

event::ConnectionPtr update\_connection\_;

double freq\_ = 0.0;

};

} // namespace gazebo

#endif // FLAPPING\_CONTROLLER\_HPP

#ifndef FLIGHT\_CONTROLLER\_GUI\_H

#define FLIGHT\_CONTROLLER\_GUI\_H

#include <QMainWindow>

#include <QPushButton>

#include <QSlider>

#include <QTextBrowser>

#include <QTimer>

#include <rclcpp/rclcpp.hpp>

#include <std\_msgs/msg/bool.hpp>

#include <std\_msgs/msg/float32.hpp>

#include <geometry\_msgs/msg/vector3.hpp>

#include "ui\_test1.h" // 这是由 Qt Designer 生成的 UI 文件

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = nullptr);

~MainWindow();

std::shared\_ptr<rclcpp::Node> get\_node() const { return node\_; }

private:

// UI components

Ui ::MainWindow \*ui;

// ROS2 components

std::shared\_ptr<rclcpp::Node> node\_;

rclcpp::Publisher<std\_msgs::msg::Float32>::SharedPtr freq\_pub\_;

rclcpp::Publisher<std\_msgs::msg::Float32>::SharedPtr pos\_pub\_;

rclcpp::Publisher<std\_msgs::msg::Float32>::SharedPtr dir\_pub\_;

rclcpp::Publisher<std\_msgs::msg::Bool>::SharedPtr stat\_pub\_;

rclcpp::Subscription<geometry\_msgs::msg::Vector3>::SharedPtr l4\_sub\_;

rclcpp::Subscription<std\_msgs::msg::Float32>::SharedPtr height\_sub\_;

QTimer \*ros\_timer\_;

// Functions

void setupStyle();

void initROSComponents();

void connectUiComponents();

// Event handlers

void onStartClicked();

void onStopClicked();

void onFrequencyChanged(int value);

void onPositionChanged(int value);

void onDirectionChanged(int value);

};

#endif // FLIGHT\_CONTROLLER\_GUI\_H

#ifndef LIFT\_DRAG\_PLUGIN\_HPP

#define LIFT\_DRAG\_PLUGIN\_HPP

#include <gazebo/common/Plugin.hh>

#include <gazebo/physics/physics.hh>

#include <rclcpp/rclcpp.hpp>

#include <geometry\_msgs/msg/vector3.hpp>

#include <std\_msgs/msg/float32.hpp>

namespace gazebo

{

class LiftDragPlugin : public ModelPlugin

{

public:

void Load(physics::ModelPtr model, sdf::ElementPtr sdf) override;

private:

void OnUpdate();

physics::ModelPtr model\_;

physics::LinkPtr link\_;

event::ConnectionPtr update\_connection\_;

rclcpp::Node::SharedPtr node\_;

rclcpp::Publisher<geometry\_msgs::msg::Vector3>::SharedPtr pub\_force\_;

rclcpp::Publisher<std\_msgs::msg::Float32>::SharedPtr pub\_height\_;

};

}

#endif

#ifndef PITCHING\_CONTROLLER\_HPP\_

#define PITCHING\_CONTROLLER\_HPP\_

#include <gazebo/common/Plugin.hh>

#include <gazebo/physics/physics.hh>

#include <rclcpp/rclcpp.hpp>

#include <std\_msgs/msg/float32.hpp>

namespace gazebo

{

class \_\_attribute\_\_((visibility("default"))) PitchingController : public ModelPlugin

{

public:

void Load(physics::ModelPtr model, sdf::ElementPtr sdf) override;

void OnUpdate();

private:

physics::ModelPtr model\_;

physics::JointPtr joint\_;

rclcpp::Node::SharedPtr node\_;

rclcpp::Subscription<std\_msgs::msg::Float32>::SharedPtr sub\_;

event::ConnectionPtr update\_connection\_;

float pitch\_pos\_ = 0.0f;

};

} // namespace gazebo

#endif

#ifndef ROLLING\_CONTROLLER\_HPP\_

#define ROLLING\_CONTROLLER\_HPP\_

#include <gazebo/common/Plugin.hh>

#include <gazebo/physics/physics.hh>

#include <rclcpp/rclcpp.hpp>

#include <std\_msgs/msg/float32.hpp>

namespace gazebo

{

class \_\_attribute\_\_((visibility("default"))) RollingController : public ModelPlugin

{

public:

void Load(physics::ModelPtr model, sdf::ElementPtr sdf) override;

void OnUpdate();

private:

physics::ModelPtr model\_;

physics::JointPtr joint\_;

rclcpp::Node::SharedPtr node\_;

rclcpp::Subscription<std\_msgs::msg::Float32>::SharedPtr sub\_;

event::ConnectionPtr update\_connection\_;

float roll\_pos\_ = 0.0f;

};

} // namespace gazebo

#endif

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'test1gvMrVN.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.12.8

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef TEST1GVMRVN\_H

#define TEST1GVMRVN\_H

#include <QtCore/QVariant>

#include <QtWidgets/QApplication>

#include <QtWidgets/QCheckBox>

#include <QtWidgets/QFrame>

#include <QtWidgets/QGroupBox>

#include <QtWidgets/QHBoxLayout>

#include <QtWidgets/QLabel>

#include <QtWidgets/QMainWindow>

#include <QtWidgets/QPushButton>

#include <QtWidgets/QSlider>

#include <QtWidgets/QTabWidget>

#include <QtWidgets/QTextBrowser>

#include <QtWidgets/QToolButton>

#include <QtWidgets/QVBoxLayout>

#include <QtWidgets/QWidget>

QString darkStyleSheet = R"(

QWidget {

background-color: #2E2E2E;

color: #FFFFFF;

}

QMenuBar {

background-color: #2E2E2E;

color: #FFFFFF;

}

QMenuBar::item {

background-color: #2E2E2E;

color: #FFFFFF;

}

QMenuBar::item::selected {

background-color: #555555;

}

QMenu {

background-color: #2E2E2E;

color: #FFFFFF;

}

QMenu::item::selected {

background-color: #555555;

}

QPushButton {

background-color: #4A4A4A;

color: #FFFFFF;

border: none;

padding: 5px;

}

QPushButton::hover {

background-color: #555555;

}

QPushButton::pressed {

background-color: #666666;

}

QTextBrowser {

background-color: #2E2E2E;

color: #FFFFFF;

border: none;

}

QMainWindow {

background-color: #2E2E2E;

}

ColorButton {

background-color: #4A4A4A;

color: #000000;

border: none;

padding: 5px;

}

ColorButton:hover {

background-color: #555555;

}

ColorButton:pressed {

background-color: #666666;

}

)";

QT\_BEGIN\_NAMESPACE

class Ui\_MainWindow

{

public:

QWidget \*centralwidget;

QTabWidget \*tabWidget;

QWidget \*tab;

QLabel \*label\_4;

QFrame \*frame\_2;

QWidget \*widget;

QSlider \*verticalSlider\_2;

QSlider \*verticalSlider;

QSlider \*horizontalSlider;

QWidget \*layoutWidget;

QHBoxLayout \*horizontalLayout;

QLabel \*label;

QLabel \*label\_2;

QLabel \*label\_3;

QFrame \*frame;

QVBoxLayout \*verticalLayout;

QPushButton \*pushButton;

QPushButton \*pushButton\_2;

QToolButton \*toolButton;

QFrame \*frame\_4;

QWidget \*horizontalLayoutWidget;

QHBoxLayout \*horizontalLayout\_2;

QVBoxLayout \*verticalLayout\_2;

QLabel \*label\_6;

QLabel \*label\_5;

QLabel \*label\_7;

QLabel \*label\_13;

QVBoxLayout \*verticalLayout\_3;

QTextBrowser \*textBrowser\_2;

QTextBrowser \*textBrowser\_3;

QTextBrowser \*textBrowser;

QTextBrowser \*textBrowser\_16;

QWidget \*tab\_2;

QLabel \*label\_12;

QFrame \*frame\_3;

QWidget \*horizontalLayoutWidget\_2;

QHBoxLayout \*horizontalLayout\_3;

QVBoxLayout \*verticalLayout\_12;

QLabel \*label\_8;

QLabel \*label\_9;

QLabel \*label\_10;

QLabel \*label\_11;

QVBoxLayout \*verticalLayout\_9;

QTextBrowser \*textBrowser\_4;

QTextBrowser \*textBrowser\_7;

QTextBrowser \*textBrowser\_10;

QTextBrowser \*textBrowser\_13;

QVBoxLayout \*verticalLayout\_10;

QTextBrowser \*textBrowser\_5;

QTextBrowser \*textBrowser\_8;

QTextBrowser \*textBrowser\_11;

QTextBrowser \*textBrowser\_14;

QVBoxLayout \*verticalLayout\_6;

QTextBrowser \*textBrowser\_6;

QTextBrowser \*textBrowser\_9;

QTextBrowser \*textBrowser\_12;

QTextBrowser \*textBrowser\_15;

QWidget \*tab\_3;

QGroupBox \*groupBox;

QWidget \*verticalLayoutWidget;

QVBoxLayout \*verticalLayout\_4;

QPushButton \*pushButton\_3;

QPushButton \*pushButton\_4;

QWidget \*widget\_2;

QFrame \*frame\_5;

QWidget \*verticalLayoutWidget\_2;

QVBoxLayout \*verticalLayout\_5;

QCheckBox \*checkBox\_2;

QCheckBox \*checkBox;

QCheckBox \*checkBox\_3;

QCheckBox \*checkBox\_4;

QCheckBox \*checkBox\_5;

QCheckBox \*checkBox\_6;

void setupUi(QMainWindow \*MainWindow)

{

if (MainWindow->objectName().isEmpty())

MainWindow->setObjectName(QString::fromUtf8("MainWindow"));

MainWindow->resize(906, 547);

centralwidget = new QWidget(MainWindow);

centralwidget->setObjectName(QString::fromUtf8("centralwidget"));

tabWidget = new QTabWidget(centralwidget);

tabWidget->setObjectName(QString::fromUtf8("tabWidget"));

tabWidget->setGeometry(QRect(10, 10, 871, 531));

tab = new QWidget();

tab->setObjectName(QString::fromUtf8("tab"));

label\_4 = new QLabel(tab);

label\_4->setObjectName(QString::fromUtf8("label\_4"));

label\_4->setGeometry(QRect(180, 420, 421, 51));

QFont font;

font.setPointSize(21);

label\_4->setFont(font);

frame\_2 = new QFrame(tab);

frame\_2->setObjectName(QString::fromUtf8("frame\_2"));

frame\_2->setGeometry(QRect(10, 30, 491, 371));

frame\_2->setFrameShape(QFrame::StyledPanel);

frame\_2->setFrameShadow(QFrame::Raised);

widget = new QWidget(frame\_2);

widget->setObjectName(QString::fromUtf8("widget"));

widget->setGeometry(QRect(10, 10, 471, 351));

verticalSlider\_2 = new QSlider(widget);

verticalSlider\_2->setObjectName(QString::fromUtf8("verticalSlider\_2"));

verticalSlider\_2->setGeometry(QRect(230, 10, 31, 281));

verticalSlider\_2->setAutoFillBackground(false);

verticalSlider\_2->setMinimum(-50);

verticalSlider\_2->setMaximum(50);

verticalSlider\_2->setSliderPosition(0);

verticalSlider\_2->setTracking(true);

verticalSlider\_2->setOrientation(Qt::Vertical);

verticalSlider\_2->setInvertedAppearance(false);

verticalSlider\_2->setTickPosition(QSlider::NoTicks);

verticalSlider = new QSlider(widget);

verticalSlider->setObjectName(QString::fromUtf8("verticalSlider"));

verticalSlider->setGeometry(QRect(390, 10, 31, 291));

verticalSlider->setOrientation(Qt::Vertical);

verticalSlider->setTickPosition(QSlider::NoTicks);

horizontalSlider = new QSlider(widget);

horizontalSlider->setObjectName(QString::fromUtf8("horizontalSlider"));

horizontalSlider->setGeometry(QRect(30, 200, 161, 18));

horizontalSlider->setLayoutDirection(Qt::LeftToRight);

horizontalSlider->setAutoFillBackground(true);

horizontalSlider->setOrientation(Qt::Horizontal);

layoutWidget = new QWidget(widget);

layoutWidget->setObjectName(QString::fromUtf8("layoutWidget"));

layoutWidget->setGeometry(QRect(30, 320, 451, 23));

horizontalLayout = new QHBoxLayout(layoutWidget);

horizontalLayout->setObjectName(QString::fromUtf8("horizontalLayout"));

horizontalLayout->setContentsMargins(0, 0, 0, 0);

label = new QLabel(layoutWidget);

label->setObjectName(QString::fromUtf8("label"));

label->setLayoutDirection(Qt::LeftToRight);

horizontalLayout->addWidget(label);

label\_2 = new QLabel(layoutWidget);

label\_2->setObjectName(QString::fromUtf8("label\_2"));

horizontalLayout->addWidget(label\_2);

label\_3 = new QLabel(layoutWidget);

label\_3->setObjectName(QString::fromUtf8("label\_3"));

horizontalLayout->addWidget(label\_3);

frame = new QFrame(tab);

frame->setObjectName(QString::fromUtf8("frame"));

frame->setGeometry(QRect(710, 10, 141, 131));

frame->setFrameShape(QFrame::StyledPanel);

frame->setFrameShadow(QFrame::Raised);

verticalLayout = new QVBoxLayout(frame);

verticalLayout->setObjectName(QString::fromUtf8("verticalLayout"));

pushButton = new QPushButton(frame);

pushButton->setObjectName(QString::fromUtf8("pushButton"));

pushButton->setLayoutDirection(Qt::RightToLeft);

pushButton->setAutoDefault(false);

pushButton->setFlat(false);

verticalLayout->addWidget(pushButton);

pushButton\_2 = new QPushButton(frame);

pushButton\_2->setObjectName(QString::fromUtf8("pushButton\_2"));

pushButton\_2->setCursor(QCursor(Qt::ArrowCursor));

pushButton\_2->setAcceptDrops(false);

verticalLayout->addWidget(pushButton\_2);

toolButton = new QToolButton(tab);

toolButton->setObjectName(QString::fromUtf8("toolButton"));

toolButton->setGeometry(QRect(830, 450, 30, 28));

frame\_4 = new QFrame(tab);

frame\_4->setObjectName(QString::fromUtf8("frame\_4"));

frame\_4->setGeometry(QRect(540, 230, 311, 171));

frame\_4->setFrameShape(QFrame::StyledPanel);

frame\_4->setFrameShadow(QFrame::Raised);

horizontalLayoutWidget = new QWidget(frame\_4);

horizontalLayoutWidget->setObjectName(QString::fromUtf8("horizontalLayoutWidget"));

horizontalLayoutWidget->setGeometry(QRect(10, 10, 291, 151));

horizontalLayout\_2 = new QHBoxLayout(horizontalLayoutWidget);

horizontalLayout\_2->setObjectName(QString::fromUtf8("horizontalLayout\_2"));

horizontalLayout\_2->setContentsMargins(0, 0, 0, 0);

verticalLayout\_2 = new QVBoxLayout();

verticalLayout\_2->setObjectName(QString::fromUtf8("verticalLayout\_2"));

label\_6 = new QLabel(horizontalLayoutWidget);

label\_6->setObjectName(QString::fromUtf8("label\_6"));

verticalLayout\_2->addWidget(label\_6);

label\_5 = new QLabel(horizontalLayoutWidget);

label\_5->setObjectName(QString::fromUtf8("label\_5"));

verticalLayout\_2->addWidget(label\_5);

label\_7 = new QLabel(horizontalLayoutWidget);

label\_7->setObjectName(QString::fromUtf8("label\_7"));

verticalLayout\_2->addWidget(label\_7);

label\_13 = new QLabel(horizontalLayoutWidget);

label\_13->setObjectName(QString::fromUtf8("label\_13"));

verticalLayout\_2->addWidget(label\_13);

horizontalLayout\_2->addLayout(verticalLayout\_2);

verticalLayout\_3 = new QVBoxLayout();

verticalLayout\_3->setObjectName(QString::fromUtf8("verticalLayout\_3"));

textBrowser\_2 = new QTextBrowser(horizontalLayoutWidget);

textBrowser\_2->setObjectName(QString::fromUtf8("textBrowser\_2"));

verticalLayout\_3->addWidget(textBrowser\_2);

textBrowser\_3 = new QTextBrowser(horizontalLayoutWidget);

textBrowser\_3->setObjectName(QString::fromUtf8("textBrowser\_3"));

verticalLayout\_3->addWidget(textBrowser\_3);

textBrowser = new QTextBrowser(horizontalLayoutWidget);

textBrowser->setObjectName(QString::fromUtf8("textBrowser"));

verticalLayout\_3->addWidget(textBrowser);

textBrowser\_16 = new QTextBrowser(horizontalLayoutWidget);

textBrowser\_16->setObjectName(QString::fromUtf8("textBrowser\_16"));

verticalLayout\_3->addWidget(textBrowser\_16);

horizontalLayout\_2->addLayout(verticalLayout\_3);

tabWidget->addTab(tab, QString());

tab\_2 = new QWidget();

tab\_2->setObjectName(QString::fromUtf8("tab\_2"));

label\_12 = new QLabel(tab\_2);

label\_12->setObjectName(QString::fromUtf8("label\_12"));

label\_12->setGeometry(QRect(30, 20, 80, 21));

frame\_3 = new QFrame(tab\_2);

frame\_3->setObjectName(QString::fromUtf8("frame\_3"));

frame\_3->setGeometry(QRect(30, 50, 661, 171));

frame\_3->setFrameShape(QFrame::StyledPanel);

frame\_3->setFrameShadow(QFrame::Raised);

horizontalLayoutWidget\_2 = new QWidget(frame\_3);

horizontalLayoutWidget\_2->setObjectName(QString::fromUtf8("horizontalLayoutWidget\_2"));

horizontalLayoutWidget\_2->setGeometry(QRect(10, 10, 641, 151));

horizontalLayout\_3 = new QHBoxLayout(horizontalLayoutWidget\_2);

horizontalLayout\_3->setObjectName(QString::fromUtf8("horizontalLayout\_3"));

horizontalLayout\_3->setContentsMargins(0, 0, 0, 0);

verticalLayout\_12 = new QVBoxLayout();

verticalLayout\_12->setObjectName(QString::fromUtf8("verticalLayout\_12"));

label\_8 = new QLabel(horizontalLayoutWidget\_2);

label\_8->setObjectName(QString::fromUtf8("label\_8"));

verticalLayout\_12->addWidget(label\_8);

label\_9 = new QLabel(horizontalLayoutWidget\_2);

label\_9->setObjectName(QString::fromUtf8("label\_9"));

verticalLayout\_12->addWidget(label\_9);

label\_10 = new QLabel(horizontalLayoutWidget\_2);

label\_10->setObjectName(QString::fromUtf8("label\_10"));

verticalLayout\_12->addWidget(label\_10);

label\_11 = new QLabel(horizontalLayoutWidget\_2);

label\_11->setObjectName(QString::fromUtf8("label\_11"));

verticalLayout\_12->addWidget(label\_11);

horizontalLayout\_3->addLayout(verticalLayout\_12);

verticalLayout\_9 = new QVBoxLayout();

verticalLayout\_9->setObjectName(QString::fromUtf8("verticalLayout\_9"));

textBrowser\_4 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_4->setObjectName(QString::fromUtf8("textBrowser\_4"));

verticalLayout\_9->addWidget(textBrowser\_4);

textBrowser\_7 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_7->setObjectName(QString::fromUtf8("textBrowser\_7"));

verticalLayout\_9->addWidget(textBrowser\_7);

textBrowser\_10 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_10->setObjectName(QString::fromUtf8("textBrowser\_10"));

verticalLayout\_9->addWidget(textBrowser\_10);

textBrowser\_13 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_13->setObjectName(QString::fromUtf8("textBrowser\_13"));

verticalLayout\_9->addWidget(textBrowser\_13);

horizontalLayout\_3->addLayout(verticalLayout\_9);

verticalLayout\_10 = new QVBoxLayout();

verticalLayout\_10->setObjectName(QString::fromUtf8("verticalLayout\_10"));

textBrowser\_5 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_5->setObjectName(QString::fromUtf8("textBrowser\_5"));

verticalLayout\_10->addWidget(textBrowser\_5);

textBrowser\_8 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_8->setObjectName(QString::fromUtf8("textBrowser\_8"));

verticalLayout\_10->addWidget(textBrowser\_8);

textBrowser\_11 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_11->setObjectName(QString::fromUtf8("textBrowser\_11"));

verticalLayout\_10->addWidget(textBrowser\_11);

textBrowser\_14 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_14->setObjectName(QString::fromUtf8("textBrowser\_14"));

verticalLayout\_10->addWidget(textBrowser\_14);

horizontalLayout\_3->addLayout(verticalLayout\_10);

verticalLayout\_6 = new QVBoxLayout();

verticalLayout\_6->setObjectName(QString::fromUtf8("verticalLayout\_6"));

textBrowser\_6 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_6->setObjectName(QString::fromUtf8("textBrowser\_6"));

verticalLayout\_6->addWidget(textBrowser\_6);

textBrowser\_9 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_9->setObjectName(QString::fromUtf8("textBrowser\_9"));

verticalLayout\_6->addWidget(textBrowser\_9);

textBrowser\_12 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_12->setObjectName(QString::fromUtf8("textBrowser\_12"));

verticalLayout\_6->addWidget(textBrowser\_12);

textBrowser\_15 = new QTextBrowser(horizontalLayoutWidget\_2);

textBrowser\_15->setObjectName(QString::fromUtf8("textBrowser\_15"));

verticalLayout\_6->addWidget(textBrowser\_15);

horizontalLayout\_3->addLayout(verticalLayout\_6);

tabWidget->addTab(tab\_2, QString());

tab\_3 = new QWidget();

tab\_3->setObjectName(QString::fromUtf8("tab\_3"));

groupBox = new QGroupBox(tab\_3);

groupBox->setObjectName(QString::fromUtf8("groupBox"));

groupBox->setGeometry(QRect(700, 10, 151, 131));

verticalLayoutWidget = new QWidget(groupBox);

verticalLayoutWidget->setObjectName(QString::fromUtf8("verticalLayoutWidget"));

verticalLayoutWidget->setGeometry(QRect(10, 40, 131, 80));

verticalLayout\_4 = new QVBoxLayout(verticalLayoutWidget);

verticalLayout\_4->setObjectName(QString::fromUtf8("verticalLayout\_4"));

verticalLayout\_4->setContentsMargins(0, 0, 0, 0);

pushButton\_3 = new QPushButton(verticalLayoutWidget);

pushButton\_3->setObjectName(QString::fromUtf8("pushButton\_3"));

verticalLayout\_4->addWidget(pushButton\_3);

pushButton\_4 = new QPushButton(verticalLayoutWidget);

pushButton\_4->setObjectName(QString::fromUtf8("pushButton\_4"));

verticalLayout\_4->addWidget(pushButton\_4);

widget\_2 = new QWidget(tab\_3);

widget\_2->setObjectName(QString::fromUtf8("widget\_2"));

widget\_2->setGeometry(QRect(10, 10, 671, 441));

frame\_5 = new QFrame(tab\_3);

frame\_5->setObjectName(QString::fromUtf8("frame\_5"));

frame\_5->setGeometry(QRect(700, 150, 151, 231));

frame\_5->setFrameShape(QFrame::StyledPanel);

frame\_5->setFrameShadow(QFrame::Raised);

verticalLayoutWidget\_2 = new QWidget(frame\_5);

verticalLayoutWidget\_2->setObjectName(QString::fromUtf8("verticalLayoutWidget\_2"));

verticalLayoutWidget\_2->setGeometry(QRect(10, 9, 134, 211));

verticalLayout\_5 = new QVBoxLayout(verticalLayoutWidget\_2);

verticalLayout\_5->setObjectName(QString::fromUtf8("verticalLayout\_5"));

verticalLayout\_5->setContentsMargins(0, 0, 0, 0);

checkBox\_2 = new QCheckBox(verticalLayoutWidget\_2);

checkBox\_2->setObjectName(QString::fromUtf8("checkBox\_2"));

verticalLayout\_5->addWidget(checkBox\_2);

checkBox = new QCheckBox(verticalLayoutWidget\_2);

checkBox->setObjectName(QString::fromUtf8("checkBox"));

verticalLayout\_5->addWidget(checkBox);

checkBox\_3 = new QCheckBox(verticalLayoutWidget\_2);

checkBox\_3->setObjectName(QString::fromUtf8("checkBox\_3"));

verticalLayout\_5->addWidget(checkBox\_3);

checkBox\_4 = new QCheckBox(verticalLayoutWidget\_2);

checkBox\_4->setObjectName(QString::fromUtf8("checkBox\_4"));

verticalLayout\_5->addWidget(checkBox\_4);

checkBox\_5 = new QCheckBox(verticalLayoutWidget\_2);

checkBox\_5->setObjectName(QString::fromUtf8("checkBox\_5"));

verticalLayout\_5->addWidget(checkBox\_5);

checkBox\_6 = new QCheckBox(verticalLayoutWidget\_2);

checkBox\_6->setObjectName(QString::fromUtf8("checkBox\_6"));

verticalLayout\_5->addWidget(checkBox\_6);

tabWidget->addTab(tab\_3, QString());

MainWindow->setCentralWidget(centralwidget);

retranslateUi(MainWindow);

tabWidget->setCurrentIndex(0);

pushButton->setDefault(true);

QMetaObject::connectSlotsByName(MainWindow);

} // setupUi

void retranslateUi(QMainWindow \*MainWindow)

{

MainWindow->setWindowTitle(QApplication::translate("MainWindow", "FlightSimulator2024", nullptr));

label\_4->setText(QApplication::translate("MainWindow", "<html><head/><body><p align=\"center\">Flight Simulator 2024</p></body></html>", nullptr));

label->setText(QApplication::translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-weight:600;\">Left-Right</span></p></body></html>", nullptr));

label\_2->setText(QApplication::translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-weight:600;\">Up-Down</span></p></body></html>", nullptr));

label\_3->setText(QApplication::translate("MainWindow", "<html><head/><body><p align=\"center\"><span style=\" font-weight:600;\">Throttle</span></p></body></html>", nullptr));

pushButton->setText(QApplication::translate("MainWindow", "start", nullptr));

pushButton\_2->setText(QApplication::translate("MainWindow", "stop", nullptr));

toolButton->setText(QApplication::translate("MainWindow", "...", nullptr));

label\_6->setText(QApplication::translate("MainWindow", "height :", nullptr));

label\_5->setText(QApplication::translate("MainWindow", "acceleration :", nullptr));

label\_7->setText(QApplication::translate("MainWindow", "velocity :", nullptr));

label\_13->setText(QApplication::translate("MainWindow", "frequency :", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab), QApplication::translate("MainWindow", "Main", nullptr));

label\_12->setText(QApplication::translate("MainWindow", "Moments:", nullptr));

label\_8->setText(QApplication::translate("MainWindow", "Total moments : ", nullptr));

label\_9->setText(QApplication::translate("MainWindow", "Acceleration :", nullptr));

label\_10->setText(QApplication::translate("MainWindow", "velocity :", nullptr));

label\_11->setText(QApplication::translate("MainWindow", "Position :", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_2), QApplication::translate("MainWindow", "More", nullptr));

groupBox->setTitle(QApplication::translate("MainWindow", "Graph", nullptr));

pushButton\_3->setText(QApplication::translate("MainWindow", "Print", nullptr));

pushButton\_4->setText(QApplication::translate("MainWindow", "clear", nullptr));

checkBox\_2->setText(QApplication::translate("MainWindow", "Height", nullptr));

checkBox->setText(QApplication::translate("MainWindow", "Acceleration", nullptr));

checkBox\_3->setText(QApplication::translate("MainWindow", "velocity", nullptr));

checkBox\_4->setText(QApplication::translate("MainWindow", "Force x", nullptr));

checkBox\_5->setText(QApplication::translate("MainWindow", "Force y", nullptr));

checkBox\_6->setText(QApplication::translate("MainWindow", "force z", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_3), QApplication::translate("MainWindow", "View", nullptr));

} // retranslateUi

};

namespace Ui {

class MainWindow: public Ui\_MainWindow {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // TEST1GVMRVN\_H

import os

from launch import LaunchDescription

from launch.actions import IncludeLaunchDescription

from launch.launch\_description\_sources import PythonLaunchDescriptionSource

from ament\_index\_python.packages import get\_package\_share\_directory

def generate\_launch\_description():

flapping\_pkg\_path = get\_package\_share\_directory('flapping\_wing')

gazebo\_ros\_path = get\_package\_share\_directory('gazebo\_ros')

world\_file = os.path.join(flapping\_pkg\_path, 'worlds', 'simple.world')

return LaunchDescription([

IncludeLaunchDescription(

PythonLaunchDescriptionSource(

os.path.join(gazebo\_ros\_path, 'launch', 'gazebo.launch.py')

),

launch\_arguments={

'world': world\_file,

'paused': 'false',

'use\_sim\_time': 'true',

'gui': 'true'

}.items()

)

])

<sdf version='1.7'>

<model name='bird2'>

<link name='dummy'>

<inertial>

<pose>0.018317 0.0192 -0.007466 0 -0 0</pose>

<!--weight of the gears, battery and components was added to dummy link-->

<mass>0.4080716</mass>

<inertia>

<ixx>5.79725e-05</ixx>

<ixy>3.35936e-05</ixy>

<ixz>2.34318e-05</ixz>

<iyy>0.000884561</iyy>

<iyz>-7.85447e-06</iyz>

<izz>0.0008822</izz>

</inertia>

</inertial>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_0\_collision'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_0.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_1\_collision\_1'>

<pose>0.14 0.0215 0.005 -1.5708 0.57201 -0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_1.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_2\_collision\_2'>

<pose>1e-06 0.043 -0.005 -1.5708 0 2e-06</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_2.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_3\_collision\_3'>

<pose>0 0.024 0.075 -1.5708 0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_3.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_5\_collision\_4'>

<pose>0.2 0.022 0.075 -1.5708 0 -3.14159</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_5.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_7\_collision\_5'>

<pose>0 0.023 0 1.5708 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_7.STL</uri>

</mesh>

</geometry>

</collision>

<collision name='dummy\_fixed\_joint\_lump\_\_link\_9\_collision\_6'>

<pose>0 0.023 0 1.5708 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_9.STL</uri>

</mesh>

</geometry>

</collision>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_0\_visual'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_0.STL</uri>

</mesh>

</geometry>

</visual>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_1\_visual\_1'>

<pose>0.14 0.0215 0.005 -1.5708 0.57201 -0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_1.STL</uri>

</mesh>

</geometry>

</visual>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_2\_visual\_2'>

<pose>1e-06 0.043 -0.005 -1.5708 0 2e-06</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_2.STL</uri>

</mesh>

</geometry>

</visual>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_3\_visual\_3'>

<pose>0 0.024 0.075 -1.5708 0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_3.STL</uri>

</mesh>

</geometry>

</visual>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_5\_visual\_4'>

<pose>0.2 0.022 0.075 -1.5708 0 -3.14159</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_5.STL</uri>

</mesh>

</geometry>

</visual>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_7\_visual\_5'>

<pose>0 0.023 0 1.5708 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_7.STL</uri>

</mesh>

</geometry>

</visual>

<visual name='dummy\_fixed\_joint\_lump\_\_link\_9\_visual\_6'>

<pose>0 0.023 0 1.5708 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_9.STL</uri>

</mesh>

</geometry>

</visual>

<self\_collide>1</self\_collide>

</link>

<joint name='joint\_4' type='revolute'>

<pose relative\_to='dummy'>0 0.0315 0.0375 1.5708 -0 0</pose>

<parent>dummy</parent>

<child>link\_4</child>

<axis>

<xyz>-1 0 0</xyz>

<limit>

<lower>-0.34906</lower>

<upper>0.34906</upper>

<effort>10</effort>

<velocity>0</velocity>

</limit>

<dynamics>

<spring\_reference>0</spring\_reference>

<spring\_stiffness>0</spring\_stiffness>

</dynamics>

</axis>

</joint>

<link name='link\_4'>

<pose relative\_to='joint\_4'>0 0 0 0 -0 0</pose>

<inertial>

<pose>0.034546 0.022687 -0.22421 0 -0 0</pose>

<!--original mass was added to dummy link-->

<mass>0.000002</mass>

<inertia>

<ixx>0.00039334</ixx>

<ixy>-3.8751e-06</ixy>

<ixz>3.8497e-05</ixz>

<iyy>0.00043498</iyy>

<iyz>3.9291e-05</iyz>

<izz>4.9572e-05</izz>

</inertia>

</inertial>

<collision name='link\_4\_collision'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_4.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<visual name='link\_4\_visual'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_4.STL</uri>

</mesh>

</geometry>

</visual>

<self\_collide>1</self\_collide>

</link>

<joint name='joint\_6' type='revolute'>

<pose relative\_to='dummy'>0.109 0.014499 0.0375 1.5708 7e-06 7e-06</pose>

<parent>dummy</parent>

<child>link\_6</child>

<axis>

<xyz>-1 0 0</xyz>

<limit>

<lower>-0.34906</lower>

<upper>0.34906</upper>

<effort>10</effort>

<velocity>0</velocity>

</limit>

<dynamics>

<spring\_reference>0</spring\_reference>

<spring\_stiffness>0</spring\_stiffness>

</dynamics>

</axis>

</joint>

<link name='link\_6'>

<pose relative\_to='joint\_6'>0 0 0 0 -0 0</pose>

<inertial>

<pose>-0.076465 0.022688 0.22422 0 -0 0</pose>

<!--original mass was added to dummy link-->

<mass>0.000002</mass>

<inertia>

<ixx>0.0003933</ixx>

<ixy>-3.8767e-06</ixy>

<ixz>-3.8513e-05</ixz>

<iyy>0.00043493</iyy>

<iyz>-3.9288e-05</iyz>

<izz>4.9561e-05</izz>

</inertia>

</inertial>

<collision name='link\_6\_collision'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_6.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<visual name='link\_6\_visual'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_6.STL</uri>

</mesh>

</geometry>

</visual>

<self\_collide>1</self\_collide>

</link>

<joint name='joint\_8' type='revolute'>

<pose relative\_to='dummy'>-0.14971 0.023 0.0265 1.5708 -0 0</pose>

<parent>dummy</parent>

<child>link\_8</child>

<axis>

<xyz>0 0 1</xyz>

<limit>

<lower>-0.52</lower>

<upper>0.52</upper>

<effort>10</effort>

<velocity>0</velocity>

</limit>

<dynamics>

<spring\_reference>0</spring\_reference>

<spring\_stiffness>0</spring\_stiffness>

</dynamics>

</axis>

</joint>

<link name='link\_8'>

<pose relative\_to='joint\_8'>0 0 0 0 -0 0</pose>

<inertial>

<pose>-0.14737 5.2e-05 -0.042891 0 -0 0</pose>

<!--original mass was added to dummy link-->

<mass>0.0000005</mass>

<inertia>

<ixx>2.8137e-06</ixx>

<ixy>4.3881e-09</ixy>

<ixz>-1.4142e-06</ixz>

<iyy>1.9017e-05</iyy>

<iyz>1.5548e-09</iyz>

<izz>1.6205e-05</izz>

</inertia>

</inertial>

<collision name='link\_8\_collision'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_8.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<visual name='link\_8\_visual'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_8.STL</uri>

</mesh>

</geometry>

</visual>

<self\_collide>1</self\_collide>

</link>

<joint name='joint\_10' type='revolute'>

<pose relative\_to='dummy'>-0.14971 0.023 0.0265 1.5708 -0 0</pose>

<parent>dummy</parent>

<child>link\_10</child>

<axis>

<xyz>0 0 1</xyz>

<limit>

<lower>-0.52</lower>

<upper>0.52</upper>

<effort>10</effort>

<velocity>0</velocity>

</limit>

<dynamics>

<spring\_reference>0</spring\_reference>

<spring\_stiffness>0</spring\_stiffness>

</dynamics>

</axis>

</joint>

<link name='link\_10'>

<pose relative\_to='joint\_10'>0 0 0 0 -0 0</pose>

<inertial>

<pose>-0.14738 5.2e-05 0.042883 0 -0 0</pose>

<!--original mass was added to dummy link-->

<mass>0.0000005</mass>

<inertia>

<ixx>2.8136e-06</ixx>

<ixy>4.4027e-09</ixy>

<ixz>1.414e-06</ixz>

<iyy>1.9016e-05</iyy>

<iyz>-1.5539e-09</iyz>

<izz>1.6204e-05</izz>

</inertia>

</inertial>

<collision name='link\_10\_collision'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_10.STL</uri>

</mesh>

</geometry>

<surface>

<contact>

<ode/>

</contact>

<friction>

<ode/>

</friction>

</surface>

</collision>

<visual name='link\_10\_visual'>

<pose>0 0 0 0 -0 0</pose>

<geometry>

<mesh>

<scale>1 1 1</scale>

<uri>model://bird2/meshes/link\_10.STL</uri>

</mesh>

</geometry>

</visual>

<self\_collide>1</self\_collide>

</link>

<frame name='joint\_2' attached\_to='link\_1'>

<pose>-0.1123 0.084193 0.0215 0 0 -0.57201</pose>

</frame>

<frame name='link\_2' attached\_to='joint\_2'/>

<frame name='joint\_1' attached\_to='link\_0'>

<pose>0.14 0.0215 0.005 -1.5708 0.57201 0</pose>

</frame>

<frame name='link\_1' attached\_to='joint\_1'/>

<frame name='joint\_3' attached\_to='link\_0'>

<pose>0 0.024 0.075 -1.5708 0 0</pose>

</frame>

<frame name='link\_3' attached\_to='joint\_3'/>

<frame name='joint\_5' attached\_to='link\_0'>

<pose>0.2 0.022 0.075 -1.5708 -0 -3.14159</pose>

</frame>

<frame name='link\_5' attached\_to='joint\_5'/>

<frame name='joint\_7' attached\_to='link\_0'>

<pose>0 0.023 0 1.5708 -0 0</pose>

</frame>

<frame name='link\_7' attached\_to='joint\_7'/>

<frame name='joint\_9' attached\_to='link\_0'>

<pose>0 0.023 0 1.5708 -0 0</pose>

</frame>

<frame name='link\_9' attached\_to='joint\_9'/>

<frame name='dummy\_joint' attached\_to='dummy'>

<pose>0 0 0 0 -0 0</pose>

</frame>

<frame name='link\_0' attached\_to='dummy\_joint'/>

<plugin name="flapping\_controller\_plugin" filename="libflapping\_controller.so">

<namespace\_model>bird2</namespace\_model>

</plugin>

<plugin name="pitching\_controller\_plugin" filename="libpitching\_controller.so">

<namespace\_model>bird2</namespace\_model>

</plugin>

<plugin name="rolling\_controller\_plugin" filename="librolling\_controller.so">

<namespace\_model>bird2</namespace\_model>

</plugin>

<plugin name="lift\_drag\_plugin" filename="liblift\_drag\_plugin.so">

<namespace\_model>bird2</namespace\_model>

</plugin>

</model>

</sdf>

#include "flapping\_controller.hpp"

#include <rclcpp/rclcpp.hpp>

#include <std\_msgs/msg/float32.hpp>

#include <iostream>

#include <cmath>

namespace gazebo

{

void FlappingController::Load(physics::ModelPtr \_model, sdf::ElementPtr \_sdf)

{

try

{

std::cout << "[FlappingController] Load() starting..." << std::endl;

model\_ = \_model;

joint\_left\_ = model\_->GetJoint("joint\_8");

joint\_right\_ = model\_->GetJoint("joint\_10");

if (!joint\_left\_ || !joint\_right\_)

{

std::cerr << "[FlappingController] Failed to get joints!" << std::endl;

return;

}

// 设置最大施加力

joint\_left\_->SetParam("fmax", 0, 10.0);

joint\_right\_->SetParam("fmax", 0, 10.0);

// 初始化 ROS2 节点

if (!rclcpp::ok())

{

rclcpp::init(0, nullptr);

}

node\_ = std::make\_shared<rclcpp::Node>("flapping\_controller\_node");

std::string ns = \_sdf->Get<std::string>("namespace\_model");

sub\_ = node\_->create\_subscription<std\_msgs::msg::Float32>(

"/" + ns + "/flapping\_freq", 10,

[this](const std\_msgs::msg::Float32::SharedPtr msg)

{

freq\_ = msg->data;

});

update\_connection\_ = event::Events::ConnectWorldUpdateBegin(

std::bind(&FlappingController::OnUpdate, this));

std::cout << "[FlappingController] Load() completed." << std::endl;

}

catch (const std::exception &e)

{

std::cerr << "[FlappingController] Load() exception: " << e.what() << std::endl;

}

}

void FlappingController::OnUpdate()

{

double t = model\_->GetWorld()->SimTime().Double();

double force = std::sin(t \* freq\_ \* 2 \* M\_PI) \* 2.0;

joint\_left\_->SetForce(0, force);

joint\_right\_->SetForce(0, -force);

}

GZ\_REGISTER\_MODEL\_PLUGIN(FlappingController)

} // namespace gazebo

#include "flight\_controller\_gui.h"

// 注意：这里不要重复包含 ui\_test1.h，如果 flight\_controller\_gui.h 中已包含就不用再包含一次

#include <QApplication>

#include <QStyleFactory>

#include <QFile>

MainWindow::MainWindow(QWidget \*parent)

: QMainWindow(parent),

ui(new Ui::MainWindow),

node\_(std::make\_shared<rclcpp::Node>("flight\_controller\_gui\_node"))

{

// 通过 ui 指针初始化控件

ui->setupUi(this);

// 设置样式、初始化 ROS2 组件以及连接信号槽

setupStyle();

initROSComponents();

connectUiComponents();

}

void MainWindow::setupStyle()

{

qApp->setStyle(QStyleFactory::create("Fusion"));

QPalette darkPalette;

darkPalette.setColor(QPalette::Window, QColor(53,53,53));

darkPalette.setColor(QPalette::WindowText, Qt::white);

qApp->setPalette(darkPalette);

this->setStyleSheet(R"(

QTextBrowser, QSlider, QPushButton {

background-color: #404040;

color: #FFFFFF;

}

QTabWidget::pane {

border: 1px solid #606060;

}

)");

}

void MainWindow::initROSComponents()

{

freq\_pub\_ = node\_->create\_publisher<std\_msgs::msg::Float32>("/bird2/flapping\_freq", 10);

pos\_pub\_ = node\_->create\_publisher<std\_msgs::msg::Float32>("/bird2/pitch\_pos", 10);

dir\_pub\_ = node\_->create\_publisher<std\_msgs::msg::Float32>("/bird2/roll\_pos", 10);

stat\_pub\_ = node\_->create\_publisher<std\_msgs::msg::Bool>("/bird2/shutdown", 10);

l4\_sub\_ = node\_->create\_subscription<geometry\_msgs::msg::Vector3>(

"/bird2/l4", 10,

[this](const geometry\_msgs::msg::Vector3::SharedPtr msg){

QMetaObject::invokeMethod(this, [this, msg](){

// 通过 ui-> 来访问控件

ui->textBrowser->setText(QString::number(msg->x, 'f', 2));

ui->textBrowser\_3->setText(QString::number(msg->y, 'f', 2));

ui->textBrowser\_2->setText(QString::number(msg->z, 'f', 2));

});

}

);

height\_sub\_ = node\_->create\_subscription<std\_msgs::msg::Float32>(

"/bird2/height", 10,

[this](const std\_msgs::msg::Float32::SharedPtr msg){

QMetaObject::invokeMethod(this, [this, msg](){

ui->textBrowser\_16->setText(QString::number(msg->data, 'f', 1) + " m");

});

}

);

}

void MainWindow::connectUiComponents()

{

// 通过 ui-> 访问各控件进行信号槽连接

connect(ui->pushButton, &QPushButton::clicked, this, &MainWindow::onStartClicked);

connect(ui->pushButton\_2, &QPushButton::clicked, this, &MainWindow::onStopClicked);

connect(ui->verticalSlider, &QSlider::valueChanged, this, &MainWindow::onFrequencyChanged);

connect(ui->horizontalSlider, &QSlider::valueChanged, this, &MainWindow::onPositionChanged);

connect(ui->verticalSlider\_2, &QSlider::valueChanged, this, &MainWindow::onDirectionChanged);

}

void MainWindow::onStartClicked()

{

std\_msgs::msg::Bool msg;

msg.data = true;

stat\_pub\_->publish(msg);

ui->textBrowser\_16->setText("[STATUS] System ACTIVE");

ui->textBrowser\_16->setStyleSheet("color: #00FF00;");

}

void MainWindow::onStopClicked()

{

std\_msgs::msg::Bool msg;

msg.data = false;

stat\_pub\_->publish(msg);

ui->textBrowser\_16->setText("[STATUS] System STOPPED");

ui->textBrowser\_16->setStyleSheet("color: #FF0000;");

// 强制启动时发布初始参数

std\_msgs::msg::Float32 freqMsg;

freqMsg.data = static\_cast<float>(ui->verticalSlider->value());

freq\_pub\_->publish(freqMsg);

std\_msgs::msg::Float32 pitchMsg;

pitchMsg.data = static\_cast<float>(ui->verticalSlider\_2->value()) \* 0.01;

pos\_pub\_->publish(pitchMsg);

std\_msgs::msg::Float32 rollMsg;

rollMsg.data = static\_cast<float>(ui->horizontalSlider->value()) \* 0.01;

dir\_pub\_->publish(rollMsg);

std\_msgs::msg::Bool statMsg;

statMsg.data = true; // 开启系统

stat\_pub\_->publish(statMsg);

}

void MainWindow::onFrequencyChanged(int value) {

std\_msgs::msg::Float32 msg;

msg.data = static\_cast<float>(value);

freq\_pub\_->publish(msg);

}

void MainWindow::onPositionChanged(int value) {

std\_msgs::msg::Float32 msg;

msg.data = static\_cast<float>(value);

pos\_pub\_->publish(msg);

}

void MainWindow::onDirectionChanged(int value) {

std\_msgs::msg::Float32 msg;

msg.data = static\_cast<float>(value);

dir\_pub\_->publish(msg);

}

MainWindow::~MainWindow()

{

delete ui;

}

int main(int argc, char \*\*argv)

{

QApplication app(argc, argv);

rclcpp::init(argc, argv);

MainWindow window;

window.show();

// ✅ 创建 ROS2 executor 并添加节点

auto executor = std::make\_shared<rclcpp::executors::MultiThreadedExecutor>();

executor->add\_node(window.get\_node());

// ✅ 启动线程持续 spin

std::thread ros\_spin\_thread([executor]() {

executor->spin();

});

int result = app.exec();

rclcpp::shutdown();

ros\_spin\_thread.join();

return result;

}

#include "moc\_flight\_controller\_gui.cpp"

#include "lift\_drag\_plugin.hpp"

#include <rclcpp/rclcpp.hpp>

#include <geometry\_msgs/msg/vector3.hpp>

#include <std\_msgs/msg/float32.hpp>

#include <iostream>

namespace gazebo

{

void LiftDragPlugin::Load(physics::ModelPtr \_model, sdf::ElementPtr \_sdf)

{

try {

std::cout << "[LiftDragPlugin] Load() starting..." << std::endl;

model\_ = \_model;

link\_ = model\_->GetLink("dummy");

if (!link\_) {

std::cerr << "[LiftDragPlugin] Failed to find link dummy!" << std::endl;

return;

}

//rclcpp::init(0, nullptr);

node\_ = std::make\_shared<rclcpp::Node>("lift\_drag\_plugin\_node");

std::string ns = \_sdf->Get<std::string>("namespace\_model");

pub\_force\_ = node\_->create\_publisher<geometry\_msgs::msg::Vector3>(

"/" + ns + "/l4", 10);

pub\_height\_ = node\_->create\_publisher<std\_msgs::msg::Float32>(

"/" + ns + "/height", 10);

update\_connection\_ = event::Events::ConnectWorldUpdateBegin(

std::bind(&LiftDragPlugin::OnUpdate, this));

std::cout << "[LiftDragPlugin] Load() completed." << std::endl;

} catch (const std::exception &e) {

std::cerr << "[LiftDragPlugin] Load() exception: " << e.what() << std::endl;

}

}

void LiftDragPlugin::OnUpdate()

{

ignition::math::Vector3d vel = link\_->RelativeLinearVel();

ignition::math::Vector3d lift(0, 0, vel.Length() \* 0.5);

link\_->AddForce(lift);

geometry\_msgs::msg::Vector3 msg;

msg.x = lift.X();

msg.y = lift.Y();

msg.z = lift.Z();

pub\_force\_->publish(msg);

std\_msgs::msg::Float32 height\_msg;

height\_msg.data = link\_->WorldPose().Pos().Z();

pub\_height\_->publish(height\_msg);

}

GZ\_REGISTER\_MODEL\_PLUGIN(LiftDragPlugin)

}

#include "pitching\_controller.hpp"

#include <iostream>

namespace gazebo

{

void PitchingController::Load(physics::ModelPtr \_model, sdf::ElementPtr \_sdf)

{

try {

std::cout << "[PitchingController] Load() starting..." << std::endl;

model\_ = \_model;

joint\_ = model\_->GetJoint("joint\_8");

if (!joint\_) {

std::cerr << "[PitchingController] Failed to find joint\_8!" << std::endl;

return;

}

joint\_->SetParam("fmax", 0, 10.0);

node\_ = std::make\_shared<rclcpp::Node>("pitching\_controller\_node");

std::string ns = \_sdf->Get<std::string>("namespace\_model");

sub\_ = node\_->create\_subscription<std\_msgs::msg::Float32>(

"/" + ns + "/pitch\_pos", 10,

[this](const std\_msgs::msg::Float32::SharedPtr msg) {

pitch\_pos\_ = msg->data;

});

update\_connection\_ = event::Events::ConnectWorldUpdateBegin(

std::bind(&PitchingController::OnUpdate, this));

std::cout << "[PitchingController] Load() completed." << std::endl;

} catch (const std::exception &e) {

std::cerr << "[PitchingController] Load() exception: " << e.what() << std::endl;

}

}

void PitchingController::OnUpdate()

{

joint\_->SetPosition(0, pitch\_pos\_);

}

GZ\_REGISTER\_MODEL\_PLUGIN(gazebo::PitchingController)

}

#include "rolling\_controller.hpp"

#include <iostream>

namespace gazebo

{

void RollingController::Load(physics::ModelPtr \_model, sdf::ElementPtr \_sdf)

{

try {

std::cout << "[RollingController] Load() starting..." << std::endl;

model\_ = \_model;

joint\_ = model\_->GetJoint("joint\_10");

if (!joint\_) {

std::cerr << "[RollingController] Failed to find joint\_10!" << std::endl;

return;

}

joint\_->SetParam("fmax", 0, 10.0);

node\_ = std::make\_shared<rclcpp::Node>("rolling\_controller\_node");

std::string ns = \_sdf->Get<std::string>("namespace\_model");

sub\_ = node\_->create\_subscription<std\_msgs::msg::Float32>(

"/" + ns + "/roll\_pos", 10,

[this](const std\_msgs::msg::Float32::SharedPtr msg) {

roll\_pos\_ = msg->data;

});

update\_connection\_ = event::Events::ConnectWorldUpdateBegin(

std::bind(&RollingController::OnUpdate, this));

std::cout << "[RollingController] Load() completed." << std::endl;

} catch (const std::exception &e) {

std::cerr << "[RollingController] Load() exception: " << e.what() << std::endl;

}

}

void RollingController::OnUpdate()

{

joint\_->SetPosition(0, roll\_pos\_);

}

GZ\_REGISTER\_MODEL\_PLUGIN(gazebo::RollingController)

}

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>MainWindow</class>

<widget class="QMainWindow" name="MainWindow">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>827</width>

<height>530</height>

</rect>

</property>

<property name="windowTitle">

<string>MainWindow</string>

</property>

<widget class="QWidget" name="centralwidget">

<widget class="QFrame" name="frame">

<property name="geometry">

<rect>

<x>680</x>

<y>10</y>

<width>141</width>

<height>131</height>

</rect>

</property>

<property name="frameShape">

<enum>QFrame::StyledPanel</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Raised</enum>

</property>

<layout class="QVBoxLayout" name="verticalLayout">

<item>

<widget class="QPushButton" name="pushButton">

<property name="layoutDirection">

<enum>Qt::RightToLeft</enum>

</property>

<property name="text">

<string>start</string>

</property>

<property name="autoDefault">

<bool>false</bool>

</property>

<property name="default">

<bool>true</bool>

</property>

<property name="flat">

<bool>false</bool>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="pushButton\_2">

<property name="cursor">

<cursorShape>ArrowCursor</cursorShape>

</property>

<property name="acceptDrops">

<bool>false</bool>

</property>

<property name="text">

<string>stop</string>

</property>

</widget>

</item>

</layout>

</widget>

<widget class="QWidget" name="horizontalLayoutWidget">

<property name="geometry">

<rect>

<x>530</x>

<y>250</y>

<width>291</width>

<height>141</height>

</rect>

</property>

<layout class="QHBoxLayout" name="horizontalLayout\_2">

<item>

<layout class="QVBoxLayout" name="verticalLayout\_2">

<item>

<widget class="QLabel" name="label\_6">

<property name="text">

<string>height :</string>

</property>

</widget>

</item>

<item>

<widget class="QLabel" name="label\_5">

<property name="text">

<string>acceleration :</string>

</property>

</widget>

</item>

<item>

<widget class="QLabel" name="label\_7">

<property name="text">

<string>velocity :</string>

</property>

</widget>

</item>

</layout>

</item>

<item>

<layout class="QVBoxLayout" name="verticalLayout\_3">

<item>

<widget class="QTextBrowser" name="textBrowser\_2"/>

</item>

<item>

<widget class="QTextBrowser" name="textBrowser\_3"/>

</item>

<item>

<widget class="QTextBrowser" name="textBrowser"/>

</item>

</layout>

</item>

</layout>

</widget>

<widget class="QFrame" name="frame\_2">

<property name="geometry">

<rect>

<x>20</x>

<y>20</y>

<width>491</width>

<height>401</height>

</rect>

</property>

<property name="frameShape">

<enum>QFrame::StyledPanel</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Raised</enum>

</property>

<widget class="QWidget" name="widget" native="true">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>491</width>

<height>401</height>

</rect>

</property>

<widget class="QSlider" name="verticalSlider\_2">

<property name="geometry">

<rect>

<x>250</x>

<y>110</y>

<width>20</width>

<height>191</height>

</rect>

</property>

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

</widget>

<widget class="QSlider" name="verticalSlider">

<property name="geometry">

<rect>

<x>370</x>

<y>110</y>

<width>71</width>

<height>191</height>

</rect>

</property>

<property name="orientation">

<enum>Qt::Vertical</enum>

</property>

</widget>

<widget class="QSlider" name="horizontalSlider">

<property name="geometry">

<rect>

<x>30</x>

<y>200</y>

<width>161</width>

<height>18</height>

</rect>

</property>

<property name="layoutDirection">

<enum>Qt::LeftToRight</enum>

</property>

<property name="autoFillBackground">

<bool>true</bool>

</property>

<property name="orientation">

<enum>Qt::Horizontal</enum>

</property>

</widget>

<widget class="QWidget" name="layoutWidget">

<property name="geometry">

<rect>

<x>30</x>

<y>320</y>

<width>451</width>

<height>23</height>

</rect>

</property>

<layout class="QHBoxLayout" name="horizontalLayout">

<item>

<widget class="QLabel" name="label">

<property name="layoutDirection">

<enum>Qt::LeftToRight</enum>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p align=&quot;center&quot;&gt;&lt;span style=&quot; font-weight:600;&quot;&gt;Left-Right&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

</item>

<item>

<widget class="QLabel" name="label\_2">

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p align=&quot;center&quot;&gt;&lt;span style=&quot; font-weight:600;&quot;&gt;Up-Down&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

</item>

<item>

<widget class="QLabel" name="label\_3">

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p align=&quot;center&quot;&gt;&lt;span style=&quot; font-weight:600;&quot;&gt;Throttle&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

</item>

</layout>

</widget>

</widget>

</widget>

<widget class="QLabel" name="label\_4">

<property name="geometry">

<rect>

<x>-100</x>

<y>470</y>

<width>1051</width>

<height>51</height>

</rect>

</property>

<property name="font">

<font>

<pointsize>21</pointsize>

</font>

</property>

<property name="text">

<string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p align=&quot;center&quot;&gt;Flight Simulator2024&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

</widget>

</widget>

</widget>

<resources/>

<connections/>

</ui>

<?xml version="1.0" ?>

<sdf version="1.7">

<world name="default">

<include>

<uri>model://ground\_plane</uri>

</include>

<include>

<uri>model://sun</uri>

</include>

<include>

<uri>model://bird2</uri>

<name>bird2</name>

<pose>0 0 1 0 0 0</pose>

</include>

</world>

</sdf>

cmake\_minimum\_required(VERSION 3.8)

project(flapping\_wing)

# 启用Qt自动处理

set(CMAKE\_AUTOMOC ON)

set(CMAKE\_AUTOUIC ON)

set(CMAKE\_AUTORCC ON)

# 查找依赖包

find\_package(ament\_cmake REQUIRED)

find\_package(rclcpp REQUIRED)

find\_package(std\_msgs REQUIRED)

find\_package(geometry\_msgs REQUIRED)

find\_package(gazebo\_ros REQUIRED)

find\_package(gazebo REQUIRED)

find\_package(Qt5 COMPONENTS Core Widgets REQUIRED)

# 包含目录

include\_directories(

${CMAKE\_CURRENT\_SOURCE\_DIR}/include

${CMAKE\_CURRENT\_BINARY\_DIR}/\_autogen/include

${GAZEBO\_INCLUDE\_DIRS}

${Qt5Widgets\_INCLUDE\_DIRS}

${rclcpp\_INCLUDE\_DIRS}

)

# 查找UI文件

file(GLOB UI\_FILES "src/\*.ui")

###################################################

# 编译所有插件（独立编译）

###################################################

# 1. Flapping Controller插件

add\_library(flapping\_controller SHARED src/flapping\_controller.cpp)

target\_link\_libraries(flapping\_controller

${GAZEBO\_LIBRARIES}

${rclcpp\_LIBRARIES}

)

ament\_target\_dependencies(flapping\_controller

rclcpp

std\_msgs

geometry\_msgs

gazebo\_ros

)

set\_target\_properties(flapping\_controller PROPERTIES

CXX\_STANDARD 17

CXX\_STANDARD\_REQUIRED ON

CXX\_VISIBILITY\_PRESET hidden

VISIBILITY\_INLINES\_HIDDEN ON

)

# 2. Lift-Drag插件

add\_library(lift\_drag\_plugin SHARED src/lift\_drag\_plugin.cpp)

target\_link\_libraries(lift\_drag\_plugin

${GAZEBO\_LIBRARIES}

${rclcpp\_LIBRARIES}

)

ament\_target\_dependencies(lift\_drag\_plugin

rclcpp

std\_msgs

geometry\_msgs

gazebo\_ros

)

# 3. Pitching Controller插件

add\_library(pitching\_controller SHARED src/pitching\_controller.cpp)

target\_link\_libraries(pitching\_controller

${GAZEBO\_LIBRARIES}

${rclcpp\_LIBRARIES}

)

ament\_target\_dependencies(pitching\_controller

rclcpp

std\_msgs

geometry\_msgs

gazebo\_ros

)

# 4. Rolling Controller插件

add\_library(rolling\_controller SHARED src/rolling\_controller.cpp)

target\_link\_libraries(rolling\_controller

${GAZEBO\_LIBRARIES}

${rclcpp\_LIBRARIES}

)

ament\_target\_dependencies(rolling\_controller

rclcpp

std\_msgs

geometry\_msgs

gazebo\_ros

)

###################################################

# 编译GUI节点

###################################################

add\_executable(flight\_controller\_gui

src/flight\_controller\_gui.cpp

${UI\_FILES}

)

target\_link\_libraries(flight\_controller\_gui

Qt5::Widgets

${rclcpp\_LIBRARIES}

${std\_msgs\_LIBRARIES}

)

ament\_target\_dependencies(flight\_controller\_gui

rclcpp

std\_msgs

geometry\_msgs

)

###################################################

# 安装配置

###################################################

install(TARGETS

flapping\_controller

lift\_drag\_plugin

pitching\_controller

rolling\_controller

flight\_controller\_gui

DESTINATION lib

)

install(DIRECTORY

launch

worlds

models

DESTINATION share/${PROJECT\_NAME}

)

install(DIRECTORY

include/

DESTINATION include

)

ament\_package()

<?xml version="1.0"?>

<?xml-model href="http://download.ros.org/schema/package\_format3.xsd" schematypens="http://www.w3.org/2001/XMLSchema"?>

<package format="3">

<name>flapping\_wing</name>

<version>0.0.0</version>

<description>Flapping wing controller for simulation in Gazebo and ROS2</description>

<maintainer email="dong@todo.todo">dong</maintainer>

<license>Apache-2.0</license>

<!-- Build dependencies -->

<buildtool\_depend>ament\_cmake</buildtool\_depend>

<build\_depend>rclcpp</build\_depend>

<build\_depend>std\_msgs</build\_depend>

<build\_depend>geometry\_msgs</build\_depend>

<build\_depend>std\_srvs</build\_depend>

<build\_depend>gazebo\_ros</build\_depend>

<build\_depend>gazebo</build\_depend>

<depend>qt\_gui\_cpp</depend>

<!-- Runtime dependencies -->

<exec\_depend>rclcpp</exec\_depend>

<exec\_depend>std\_msgs</exec\_depend>

<exec\_depend>geometry\_msgs</exec\_depend>

<exec\_depend>std\_srvs</exec\_depend>

<exec\_depend>gazebo\_ros</exec\_depend>

<exec\_depend>gazebo</exec\_depend>

<!-- Test dependencies -->

<test\_depend>ament\_lint\_auto</test\_depend>

<test\_depend>ament\_lint\_common</test\_depend>

<!-- Export the build type -->

<export>

<build\_type>ament\_cmake</build\_type>

</export>

</package>