/\*\* @file demo\_flight\_control.h

\* @version 3.3

\* @date May, 2017

\*

\* @brief

\* demo sample of how to use flight control APIs

\*

\* @copyright 2017 DJI. All rights reserved.

\*

\*/

#ifndef DEMO\_FLIGHT\_CONTROL\_H

#define DEMO\_FLIGHT\_CONTROL\_H

// ROS includes

#include <ros/ros.h>

#include <geometry\_msgs/QuaternionStamped.h>

#include <geometry\_msgs/Vector3Stamped.h>

#include <sensor\_msgs/NavSatFix.h>

#include <std\_msgs/UInt8.h>

// DJI SDK includes

#include <dji\_sdk/DroneTaskControl.h>

#include <dji\_sdk/SDKControlAuthority.h>

#include <dji\_sdk/QueryDroneVersion.h>

#include <dji\_sdk/SetLocalPosRef.h>

#include <tf/tf.h>

#include <sensor\_msgs/Joy.h>

#define C\_EARTH (double)6378137.0

#define C\_PI (double)3.141592653589793

#define DEG2RAD(DEG) ((DEG) \* ((C\_PI) / (180.0)))

/\*!

\* @brief a bare bone state machine to track the stage of the mission

\*/

class Mission

{

public:

// The basic state transition flow is:

// 0---> 1 ---> 2 ---> ... ---> N ---> 0

// where state 0 means the mission is note started

// and each state i is for the process of moving to a target point.

int state;

int inbound\_counter;

int outbound\_counter;

int break\_counter;

float target\_offset\_x;

float target\_offset\_y;

float target\_offset\_z;

float target\_yaw;

sensor\_msgs::NavSatFix start\_gps\_location;

geometry\_msgs::Point start\_local\_position;

bool finished;

Mission() : state(0), inbound\_counter(0), outbound\_counter(0), break\_counter(0),

target\_offset\_x(0.0), target\_offset\_y(0.0), target\_offset\_z(0.0),

finished(false)

{

}

void step();

void setTarget(float x, float y, float z, float yaw)

{

target\_offset\_x = x;

target\_offset\_y = y;

target\_offset\_z = z;

target\_yaw = yaw;

}

void reset()

{

inbound\_counter = 0;

outbound\_counter = 0;

break\_counter = 0;

finished = false;

}

};

void localOffsetFromGpsOffset(geometry\_msgs::Vector3& deltaNed,

sensor\_msgs::NavSatFix& target,

sensor\_msgs::NavSatFix& origin);

geometry\_msgs::Vector3 toEulerAngle(geometry\_msgs::Quaternion quat);

void display\_mode\_callback(const std\_msgs::UInt8::ConstPtr& msg);

void flight\_status\_callback(const std\_msgs::UInt8::ConstPtr& msg);

void gps\_callback(const sensor\_msgs::NavSatFix::ConstPtr& msg);

void attitude\_callback(const geometry\_msgs::QuaternionStamped::ConstPtr& msg);

void local\_position\_callback(const geometry\_msgs::PointStamped::ConstPtr& msg);

bool takeoff\_land(int task);

bool obtain\_control();

bool is\_M100();

bool monitoredTakeoff();

bool M100monitoredTakeoff();

bool set\_local\_position();

#endif // DEMO\_FLIGHT\_CONTROL\_H