**Menjalankan Outdoor.launch**

RpLidar S1

Dual Lens Camera

<workspace>/src/haha/src/Image\_infrared.cpp

<workspace>/src/haha/src/Image\_front.cpp

<workspace>/src/rplidar\_ros/src/node.cpp

<workspace>/src/haha/src/pub\_external.cpp

Pixhawk + GPS

Mavros

<workspace>/src/haha/src/display\_sensor.cpp

1. Mavros -> Package untuk membaca data dari Pixhawk ke Komputer.
2. image\_front.cpp -> Membaca kamera RGB

* mengubah nomer kamera

VideoCapture cap(camera\_number); //camera\_number ada pada header /haha/include/haha/hihi.hpp

1. image\_infrared.cpp -> Membaca kamera IR

* mengubah nomer kamera

VideoCapture cap(camera\_number); //camera\_number ada pada header /haha/include/haha/hihi.hpp

1. node.cpp -> Membaca Lidar, bawaan dari SlamTec
2. display\_sensor.cpp -> Menampilkan dan merekam data

* void imageCallback(const sensor\_msgs::CompressedImageConstPtr& msg)

menerima gambar dari image\_front.cpp

* void irCallback(const sensor\_msgs::CompressedImageConstPtr& msg)

menerima gambar dari image\_infrared.cpp

* void imageProcessing(Mat input\_image)

memberi text pada gambar RGB

* void irProcessing(Mat input\_image)

memberi text pada gambar IR

* void record\_lidar\_cb(const sensor\_msgs::LaserScan& lidar)

menerima dan merekam data dari lidar dari node.cpp

* void vfr\_hud\_cb(const mavros\_msgs::VFR\_HUD& msg)

menerima data ground\_speed dari mavros

* void globalPositionCB(const sensor\_msgs::NavSatFix& msg)

menerima data GPS dari mavros

* void compass\_rc\_cb(const std\_msgs::Float64& msg)

menerima data compass dari mavros

1. pub\_external.cpp -> mengirim data via virtual port (not working)

**Menjalankan Complete.launch**

Pixhawk + GPS

<workspace>/src/hehe/src/record.cpp

Mavros

<workspace>/src/rplidar\_ros/src/node.cpp

RpLidar S1

<workspace>/src/hehe/src/grabber.cpp

<workspace>/src/hehe/src/server.cpp

1. Mavros -> Package untuk membaca data dari Pixhawk ke Komputer.
2. node.cpp -> Membaca Lidar, bawaan dari SlamTec
3. record.cpp -> merekam semua data
4. grabber.cpp -> menerima data Pixhawk dan Lidar

* void grab\_data()

Mengumpulkan semua data

* void sensor\_data()

Membuat augmented data

* void record\_lidar\_cb(const sensor\_msgs::LaserScan& lidar)

menerima data Lidar

* void vfr\_hud\_cb(const mavros\_msgs::VFR\_HUD& msg)

menerima data ground\_speed dari Pixhwak

* void globalPositionCB(const sensor\_msgs::NavSatFix& msg)

menerima data GPS

* void compass\_rc\_cb(const std\_msgs::Float64& msg)

menerima data compass

1. server.cpp -> membuat TCP server, menyusun dan mengirim payload ke QtCreator

* void lidar\_cb(const hehe::lidar& lidar)

menerima data Lidar

* void pixhawk\_cb(const hehe::pixhawk& pixhawk)

menerima data pixhawk

A screenshot of a cell phone

Description automatically generated**Dashboard QtCreator**

* Ver\_3.pro

QT += core gui network multimedia webkit -> module yang digunakan pada QT

* Header
* Mainwindow.h -> header untuk seluruh project
* QMovie \*ir\_movie = new QMovie("../ir.gif");
* QMovie \*front\_movie = new QMovie("../front.gif");
* QMovie \*audio\_movie = new QMovie("../audio.gif");

Alamat video rekaman percobaan

* Playlist.h -> header untuk playlist music
* Track.h -> header untuk track music
* Utils.h -> header untuk lokasi music
* Widget.h -> header untuk google maps
* Sources
* Main.cpp -> untuk menampilkan main window
* Playlist.cpp -> untuk playlist music
* Track.cpp -> untuk track music
* Widget.cpp
* Widget::Widget(QWidget \*parent): QWidget(parent)

Membuat layout untuk koneksi google maps

* void Widget::buildControls()

membuat data logger

* void Widget::log(QString text, QString delimiter)

membuat scroll data logger

* void Widget::log(const char \*text, QString delimiter)

membuat log limiter

* void Widget::onMapLoaded()

menampilkan titik tengah berdasarkan latitude dan longitude

* void Widget::onMapBecameIdle()

kondisi idle

* void Widget::onRegionChanged(QMCoordinateRegion region)

mengubah region

* Mainwindow.cpp
* MainWindow::MainWindow(QWidget \*parent):QMainWindow(parent), ui(new Ui::MainWindow),\_socket(this)
* ir\_movie->start(); -> menjalankan rekaman kamera
* ui->ir\_camera->setMovie(ir\_movie);
* ui->date->setStyleSheet("color: white;"); -> memberi warna pada text
* \_socket.connectToHost(QHostAddress("127.0.0.1"), 8080); -> connect TCP client ke server
* connect(&\_socket, SIGNAL(readyRead()), this, SLOT(onReadyRead()));
* void MainWindow::on\_main\_window\_clicked() -> untuk pindah-pindah panel (stacked widget)
* void MainWindow::onReadyRead()
* QDate date\_str = QDate::currentDate(); -> set date
* QByteArray datas = \_socket.readAll(); -> menerima data TCP
* \_socket.write(request); -> mengirim data TCP
* if (lines[0] == "1") -> membaca jenis packet data
* ui->lidar\_8->move(690+(x-690)\*(40-lines[25].toFloat())/40, 460+(y-460)\*(40-lines[25].toFloat())/40);
* animasi dari masing-masing object Lidar

sisanya fungsi untuk music, jadi abaikan saja

* Forms -> design user interface
* Mainwindow.ui