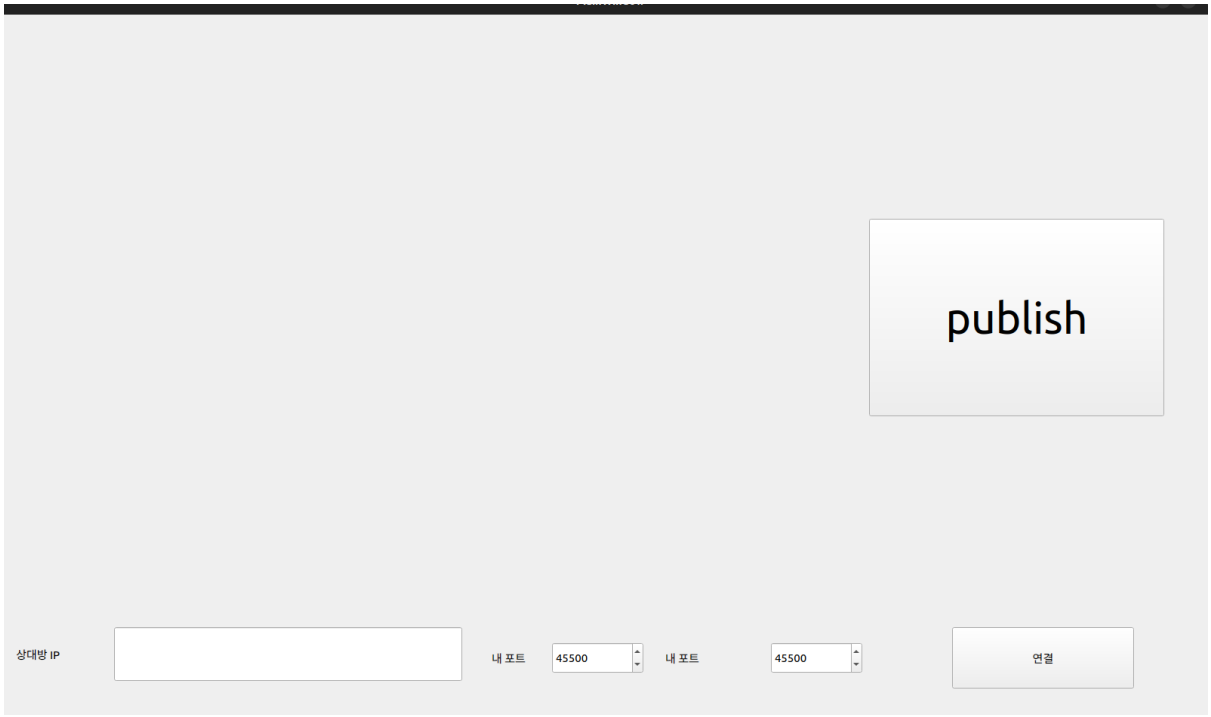
OPEN_CV DAY1
과제 2번 보고서

2023741024
로봇학부 박건후

목차

(1) ui 설명



이 부분은 패키지를 실행했을 때 화면에 뜨는 ui 화면입니다. 기존에 qt 프로젝트로 udp 통신을 했을 때처럼 lineEdit으로 상대방 ip를 입력받고, 자신의 포트와 상대방의 포트를 스핀박스를 통해 바꿀 수 있도록 하고 이를 설정한 뒤 연결 버튼이 눌리면 소켓의 bind 메서드가 호출되어 상대방 ip와 자신의 포트를 엮는 작업을 진행하도록 구성했습니다. 그리고 사진을 보내는 경우 resources 폴더 내에 이미지를 넣고 이 이미지를 경로를 통해 받아와 사진을 화면에 띄우지 않고 udp로 사진을 바로 보내는 것으로 만들었고 publish 버튼 옆의 넓은 창은 사진을 받았을 경우에만 그 받은 사진이 화면에 뜨도록 만들어서 사진을 보낼 때는 바로 보내어 사진을 받는 노트북에 사진이 문제 없이 전송되었는지 확인하고, 반대로 사진을 받았을 경우 그 사진이 화면에 크게 뜨도록 만든 것입니다.

(2) main_window.hpp 파일 설명

이번에는 qt의 udp를 사용하며 ui를 사용하기에 qt 쪽과 관련이 매우 깊습니다. 그렇기에 qnode.hpp와 qnode.cpp를 사용하지 않고 main_window 쪽의 파일로만 작업하였습니다.

```
private slots:
    void on_connection_clicked();   // 연결 버튼
    void on_publish_clicked(); // publish 버튼
    void readyRead();  // UDP 수신했을 때 콜백 함수
```

이 부분은 슬롯 메서드들을 선언한 것입니다. 제 ui를 보셨듯이 연결 버튼, 그리고 publish 버튼이 있었습니다. 그래서 각 버튼에 대한 슬롯 메서드 2개를 만들었습니다. 그리고 udpsocket의 readyRead 시그널이 발생하면 그에 따라 읽는 것을 처리하는 것이 필요하기에 그에 대한 readyRead라는 슬롯 메서드 또한 만들었습니다.

멤버변수로는 ui포인터, udpsocket에 대한 포인터를 사용하였고 private 헬퍼 메서드로 받은 사진 정보를 화면에 띄우는 헬퍼 메서드를 만들었습니다. 이처럼 필수적인 멤버들로 구성하였습니다.

(3) main_window.cpp 파일 설명

생성자:

ui에서 보듯이 스핀 박스를 통해 포트 값을 설정하게 했습니다. 하지만 스핀 박스의 값은 아마 0~99로 값이 제한되어 있는 걸로 알고 있습니다. 그래서 이 범위를 포트 범위까지 늘리기 위해 setRange를 통해 사용할 수 있는 포트 범위를 잡았습니다. 이를 각각의 스핀박스에 대해 적용하였습니다. udpsocket 포인터를 통해 동적할당하여 객체를 생성하였고, connect를 사용하여 시그널과 슬롯을 모두 연결해주었습니다.

연결 버튼을 눌렀을 때 호출되는 슬롯 메서드:

여기서는 바인드만 진행하게 됩니다. 연결을 하기 위해서입니다. 이 때 특정 ip를 선택하여 그 ip에서만 들어오게 설정할 수도 있지만 혹여나 ip 오류로 인해 사진을 받지 못 받는 것을 방지하고자 이렇게 모든 ip로부터 받을 수 있도록 AnyIPv4로 설정하였습니다.

publish 버튼을 눌렀을 때 호출되는 슬롯 메서드:

lineEdit에서 받아온 ip 정보를 가져오고 스핀 박스에 있는 값을 알아야 사진 데이터를 writeDatagram으로 전송할 수 있게 됩니다. 그래서 text()메서드로 ip를 받아오고 value라는 메서드를 통해 이를 받아왔습니다. 그리고 사진을 어떻게 writeDatagram으로 전송할지에 대해 서칭을 했었습니다. 그러다 https://stackoverflow.com/questions/45116049/save-qimage-to-array-of-bytes 이 링크를 통해 해답을 찾았습니다.

```
QFile* imageFile = new QFile("image.png");
imageFile->open(QIODevice::ReadOnly);
QByteArray ba = imageFile->readAll();
imageFile->close();
delete imageFile;
```

이 부분이 댓글로 적혀있었는데 이 부분이 확실히 명확하게 사진을 바이트로 보내는 방식이었기에 이를 참고하여 코드를 작성하였습니다. 그래서 resources 폴더의 이미지 폴더에 이미지를 저장해놓고 이에 대한 상대 경로를 적어서 사진을 가져오고 이를 readAll이라는 메서드를 통해 바이너리로 받아올 수 있었습니다. 그리고 이를 writeDatagram에 넣어서 데이터를 보냈습니다.

readyRead() 슬롯 메서드:

데이터를 받는 시그널이 오면 데이터를 받고 처리를 해야 하기에 그 작업을 여기서 구현한 것입니다. 이전에 해왔던 대로 pendingDatagramSize를 통해 받아올 값의 크기를 통해 받아올 변수의 size를 바꾸고 readDatagram으로 읽어오는 것입니다. 그리고 이 바이트 정보를 어떻게 이미지로 바꾸느냐에 대해 서칭을 진행했습니다. https://blog.naver.com/laonple/220643243718 이 url의 아래 사진 부분을 참고하였고

```
Mat (int ndims, const int *sizes, int type)
Mat (int ndims, const int *sizes, int type, void *data, const size_t *steps=0)
```

이렇게 별도로 메모리를 할당하지 않고 남의 메모리를 가져다 Mat 을 생성하는 경우.

release() 는 아무런 처리를 하지 않습니다.

즉, 남의 메모리이므로 관리는 전적으로 외부에서 이루어 져야 합니다.

```
1 unsigned char tmp[100];
2
3 Mat tmp1(2, 50, cv_8U, tmp);
4 tmp1.release();
5
6 tmp[0] = 0;    // 문제 없음
```

아래의 2개 url로 3번째 매개변수가 어떤 의미인지 공부하게 되었습니다.

https://m.blog.naver.com/dorergiverny/223035698739

https://blog.naver.com/sees111/222345251509

코드에서는 0~255값으로 rgb를 받는 형식이기에 CV_8U를 썼고 이 때 rgb색깔이 모두 나와야 하기에 3번째 채널을 사용하여 CV_8UC3으로 작성하게 되었습니다. 이렇게 이미지로 바꾸기 위해 Mat생성자에 이처럼 값을 기입한 것입니다.

그리고 이 이미지를 qt화면에 출력하면 되니 showImage를 호출했습니다.

현재 시간이 없어 보고서를 미완성하게 되었습니다. 아래 url은 제가 참고한 image를 qt에 출력할 때 참고한 url입니다. 감사합니다

https://remnant24c1.tistory.com/130#google_vignette

```
cv::Mat mat = cv::imread("/path/to/an/image.png");
QImage image(
    mat.data,
    mat.cols,
    mat.rows,
    mat.step,
    QImage::Format_RGB888
);
```

QImage로 변환하기 전에 R과 B 채널을 바꿔야 합니다.

```
cv::Mat mat = cv::imread("/path/to/an/image.png");
cv::cvtColor(mat, mat, cv::COLOR_BGR2RGB);

QImage image(
    mat.data,
    mat.cols,
    mat.rows,
    mat.step,
    QImage::Format_RGB888
);
```