8.人脸识别模块封装

新建QObject库

- 挂载在`AttendanceServer`工程
- ？？？？在头文件中加载`FaceEngine.h`文件没有找到？？？？
  编译seetaface2文件库和教程不一样
  在网上找到别人编译好的文件夹，复制过来的。

```
QFaceObject::QFaceObject(QObject *parent) : QObject(parent)
{
    // 初始化
    seeta::ModelSetting
FDmodel("H:/MCU/CMAKE_seetaface2/bin/model/fd_2_00.dat",seeta::ModelSetting:
:CPU, 0);
    seeta::ModelSetting
PDmodel("H:/MCU/CMAKE_seetaface2/bin/model/pd_2_00_pts5.dat",seeta::ModelSet
ting::CPU, 0);
    seeta::ModelSetting
FRmodel("H:/MCU/CMAKE_seetaface2/bin/model/fr_2_10.dat",seeta::ModelSetting:
:CPU, 0);
    this→fengineptr = new seeta::FaceEngine(FDmodel, PDmodel, FRmodel);
}
```

`FaceEngine.h`文件中包含新建模块。

```
 FaceEngine( const SeetaModelSetting &FD_model, const SeetaModelSetting
&PD_model, const SeetaModelSetting &FR_model )
            : FD( FD_model ), PD( PD_model ), FDB( FR_model ) {
        }
```

其中，FD,PD,FR三个模型在github中附有连接，代表三个模块FD:(FaceDetector)，PD(FaceLandmarker)，FDB(FaceDatabase)。

```
struct SeetaModelSetting
{
    enum SeetaDevice device;
    int id; // when device is GPU, id means GPU id
    const char **model; // model string terminate with nullptr
};
```

对数据进行处理时需要先将mat转化为seetaface的数据

```
struct SeetaImageData
{
    int width;
    int height;
    int channels;
    unsigned char *data;
};
```

注册

```
int64_t QFaceObject::face_register(cv::Mat &faceimage)
{
    // 把Mat数据转为seetaface的数据
    SeetaImageData simage;
    simage.data = faceimage.data;
    simage.width = faceimage.cols;
    simage.height = faceimage.rows;
    simage.channels = faceimage.channels();
    int64_t faceid = this->fengineptr->Register(simage);//注册返回一个人脸id
    if(faceid >= 0){
        fengineptr->Save("./face.db");
    }
    return faceid;
}
```

调用的是 `FaceEngine.h` 中的 `Register` 函数，封装成自己的函数。

```
int64_t Register( const SeetaImageData &image ) {
    auto faces = this->DetectFaces( image );
    if( faces.empty() ) return -1;
    return this->Register( image, faces[0] );
}
```

查询

```
int QFaceObject::face_query(cv::Mat &faceimage)
{
    // 把Mat数据转为seetaface的数据
    SeetaImageData simage;
```

```
    simage.data = faceimage.data;
    simage.width = faceimage.cols;
    simage.height = faceimage.rows;
    simage.channels = faceimage.channels();
    float similarity = 0;
    int64_t faceid = fengineptr->Query(simage, &similarity);
    return faceid;
}
```

调用的是 `FaceEngine.h` 中的 `Register` 函数，封装成自己的函数。

```
int64_t Query( const SeetaImageData &image, float *similarity = nullptr )
const {
    auto faces = this->DetectFaces( image );
    if( faces.empty() ) return -1;
    return this->Query( image, faces[0], similarity );
}
```