T-Firefly.com

Face Recognition Interface（SDK）

Overview

The face recognition SDK includes face detection, face recognition, face attribute detection and other related functions. After the SDK is authorized, the face SDK can flexibly develop upper-layer software applications based on its own related business needs and the SDK's related interfaces. Applicable scene:

- The device cannot be connected to the Internet of Things and is offline.

- Private network such as public security intranet
- The network speed is slow, to avoid taking up too much bandwidth and other environments

Binocular Camera:
IR CameraId = Camera.CameraInfo.CAMERA_FACING_FRONT
RGB CameraId = Camera.CameraInfo.CAMERA_FACING_BACK

Historical branch

Date	Version	Explanation	ModifiedBy
2021-06-01	1.3	Improve operation efficiency	XiaoDatao,SongJianfeng
2020-12-21	1.2	New synchronization interface	XiaoDatao,SongJianfeng
2020-06-19	1.1	V2	XiaoDatao,SongJianfeng
2020-05-21	1.0	FirstEditionCompleted	SongJianfeng

Development environment

1. Android Studio >=3.6.2
2. Gradel Version: 3.6.2
3. Gradle Plugin Version: 3.6.2
4. SDK Tool >=25.2.3

SDK Integration instructions

1.Copy the aar package in the faceEngineYtlf/libs/ directory of the Demo project to the libs folder corresponding to the AS project;
2.Copy the ytlf_v2 folder in the faceEngineYtlf/src/main/assets directory of the Demo project to the src/main/assets folder corresponding to the AS project;

For details, please refer to the technical case Demo project configuration or doc/FireflyApi_instructions.png；

T-Firefly.com

Interface Description

1、SDK Access

1.1 SDK Initialize

> When using YTLF Face Manager, you need to indicate the root path of the local SD card file storage directory, for example: "/sdcard/firefly/";
> When the SDK is started for the first time, it will check whether there are files such as models and license public keys in the local SD card. If not, then by default, the necessary files will be copied from the App assets directory to the rootPath directory;

```
// Specify the local SD card file storage directory
YTLFFaceManager.getInstance().initPath(String rootPath);
```

1.2 Activating License

> During initialization, it will check whether there is a local key, if not, it will be activated online, and the activation status will be called back after activation. It is divided into synchronous and asynchronous activation methods. ;

```
// Asynchronous mode
YTLFFaceManager.getInstance().initLicenseByAsync(String apiKey, new InitLicenseListener(){
        @Override
        public void onLicenseSuccess() {
                toast("success");
        }

        @Override
        public void onLicenseFail(final int code,final String msg) {
                toast("failure");
        }
     });


// Synchronization mode true means successful activation
boolean result = YTLFFaceManager.getInstance().initLicense(String apiKey);
```

1.3 Get unique device code

> Get the signature of the current device, the unique encoding of the device；

```
/*
  String :Returns the current device's signature, which is the device's current unique code
*/
String signature = YTLFFaceManager.getInstance().getSignature();
```

1.4 Offline activating License

> If the device cannot connect to the Internet, you can obtain the signature of the device through the interface get Signature() in the SDK. Each hardware device corresponds to a unique signature, for example: 3418 eb 8 df 575 c 92527 b 7 ffc 8 cdaf 34 k 9;
>
> Then send the 32-bit character string of the signature to the platform for authorization, generate a license file, and put it in the "sdcard/firefly/ytlf_v2/license" directory;

2、 SDK StartUp

T-Firefly.com

> When starting the SDK, it will detect the operating environment and initialize the SDK. There are synchronous and asynchronous startup methods.;

2.1 Start the SDK synchronously

```
/*
   Determine the config_path through FACE_PATH, which is the directory where the Config.json file is
stored
   eg:  sdcard/firefly/ytlf_v2/config.json"

  Int: 0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().startFaceSDK();

/*
    config_json Specify the contents of config.json

  Int:0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().startFaceSDKByConfigJson(String config_json);
```

2.2 Start SDK asynchronously

> To prevent too much time when initializing and starting the SDK, causing the main thread to be blocked, you can use the asynchronous method to start the SDK;

```
 /*
    config_json Specify the contents of config.json
    runSDKCallback Callback method after asynchronous start
*/
 YTLFFaceManager.getInstance().startFaceSDKByAsynchronous(String config_json, new
RunSDKCallback(){
   @Override
   public void onRunSDKListener(int i) {   //Int:0 Means success, 1 means failure

   }
});
```

2.3 The SDK reloads the config.json parameter and specifies the directory where the config.json file is stored

> The SDK reloads the config.json file to achieve face parameter reloading;

```
 /*
    configPath  'config.json'  File storage directory;  eg:/sdcard/firefly/ytlf_v2/config.json
    Int:0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().reloadConfig(String configPath);
```

T-Firefly.com

2.4 The SDK reloads the config.json parameter by specifying the content of the config.json file

> The SDK reloads the content of the config.json file to implement face parameter reloading；

```
 /*
   configJson  ‘config.json’  document content
   Int:0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().reloadConfigJson(String configJson);
```

2.5 The SDK reloads the config.json parameter and specifies the content of the config.json file, asynchronously

> The SDK reloads the content of the config.json file to achieve face parameter reloading;
> In order to prevent too much time when reloading, causing blocking of the main thread, asynchronous mode can be used;

```
 /*
   configJson 'config.json'document content
   reloadSDKCallback Asynchronous callback interface
*/
YTLFFaceManager.getInstance().reloadConfigJsonByAsync(String configJson, ReloadSDKCallback
reloadSDKCallback);
```

3、Face real-time detection and tracking

3.1 Face real-time data submission

> Perform face detection and tracking based on the data passed into the detector, and return the results of face detection and face tracking in real time.Note: The face direction of the data input to the face detector should be positive, that is, the face angle Should be 0 degrees and no other angle；
>
> Currently, RGB and IR video streams are used for face detection tracking and related detection of living bodies.When it is not necessary to turn on the IR camera or do not need living body detection, IR data parameters can pass RGB parameters, but not NULL；

```
//Get video stream from camera callback function, input RGB video stream and IR video stream in real
time
 YTLFFaceManager.getInstance().doDelivery(ArcternImage img_rgb,  ArcternImage img_ir)

//Set the RGB and IR monitor callback function
 YTLFFaceManager.getInstance().setOnDetectCallBack(new DetectCallBack() {
      //  RGB:
      @Override
      public void onDetectListener(ArcternImage arcternImage, ArcternRect[] arcternRects, float[]
confidences) {
```

t-Firefly.com

```
     }

   //  IR:
   @Override
   public void onLivingDetectListener(ArcternImage arcternImage, ArcternRect[] arcternRects, float[]
confidences) {

   }
});

/*Parameter Description：
   arcternImage RGB/IR Face image data detected
   arcternRects  RGB/IR Collection of detected face frames
   confidences RGB/IR Detect the confidence of each face
*/
```

3.2 Face real-time tracking callback

> If there is data returned from the callback interface, then the real-time data is being detected and tracked;

```
//Set real-time face detection tracking callback
YTLFFaceManager.getInstance().setOnTrackCallBack(new TrackCallBack() {

/*Face real-time detection and tracking
   arcternImage Face image data detected
   trackIds  Face tracking in images ID
   arcternRects Detected face position
*/
   @Override
   public void onTrackListener(ArcternImage arcternImage, long[] trackIds, ArcternRect[] arcternRects) {

   }
});
```

3.3 Real-time face attribute detection

> When performing real-time face detection tracking, add a face attribute detection callback method, which can receive face attributes in real time, including live detection value, quality, face angle, mask, and image color.

```
//Set real-time face attribute detection callback
YTLFFaceManager.getInstance().setOnAttributeCallBack(new AttributeCallBack() {

/*The face belongs to the monitoring callback:
   arcternImage Face image data detected
   arcternRects Detected face position
   trackIds  Face tracking in images ID
   arcternAttribute All attributes of all faces in the image
*/
   @Override
   public void onAttributeListener(ArcternImage arcternImage, long[] trackIds, ArcternRect[]
arcternRects, ArcternAttribute[][] arcternAttributes, int[] landmarks) {
       StringBuilder s = new StringBuilder();
       for (int i = 0; i < arcternRects.length; i++) {
```

T-Firefly.com

```
            for (int j = 0; j < arcternAttributes[i].length; j++) {
                ArcternAttribute attr = arcternAttributes[i][j];
                switch (j) {
                    case ArcternAttribute.ArcternFaceAttrTypeEnum.POSE_PITCH:
                        s.append("Face angle:\n With the x axis as the center, the angle of the face up and
down:").append(attr.confidence);
                        break;
                    case ArcternAttribute.ArcternFaceAttrTypeEnum.POSE_YAW:
                        s.append("\n Rotate the face left and right around the y-axis:").append(attr.confidence);
                        break;
                    case ArcternAttribute.ArcternFaceAttrTypeEnum.POSE_ROLL:
                        s.append("\n Taking the center point as the center, the x-y plane rotation
angle:").append(attr.confidence);
                        break;

                    case ArcternAttribute.ArcternFaceAttrTypeEnum.QUALITY:
                        s.append("\nFace quality:").append(attr.confidence);
                        break;

                    case ArcternAttribute.ArcternFaceAttrTypeEnum.LIVENESS_IR:
                        if (attr.label != -1) {
                            if (attr.confidence >= 0.5) {
                                s.append("\nBiopsy: live body ").append(attr.confidence);
                            } else {
                                s.append("\nBiopsy: non-living ").append(attr.confidence);
                            }
                        }
                        break;

                    case ArcternAttribute.ArcternFaceAttrTypeEnum.IMAGE_COLOR:
                        if (attr.label == ArcternAttribute.LabelFaceImgColor.COLOR) {
                            s.append("\nColor picture ").append(attr.confidence);
                        } else {
                            s.append("\nBlack and white ").append(attr.confidence);
                        }
                        break;

                    case ArcternAttribute.ArcternFaceAttrTypeEnum.FACE_MASK:
                        if (attr.label == ArcternAttribute.LabelFaceMask.MASK) {
                            s.append("\nMask ").append(attr.confidence);
                        } else {
                            s.append("\nWithout mask ").append(attr.confidence);
                        }
                        break;
                }
            }
        }
    }
});

//Face image detection synchronization method
Rect rect = YTLFFaceManager.getInstance().doDetect(Bitmap bitmap)
```

T-Firefly.com

3.4 Face real-time search

> Perform a real-time search based on the data passed into the detector, and search the database for an ID with a similarity greater than the highest set similarity. The ID can be used to obtain the recognized face and its related information;

```
//Set real-time face search callback
YTLFFaceManager.getInstance().setOnSearchCallBack(new SearchCallBack() {


/*Search callback:
  arcternImage Face image data detected
  trackIds  Face tracking in images ID
  arcternRects Detected face position
  searchId_list A collection of ids that identify faces in images
*/
  @Override
public void onSearchListener(ArcternImage arcternImage, long[] trackId_list, ArcternRect[] arcternRects,
long[] searchId_list, int[] landmarks,float[] socre) {
  if (searchId_list.length > 0 && searchId_list[0] != -1) {
    Person person = DBHelper.get(searchId_list[0]);
    if (person != null) {
      //The recognized face and related information can be obtained by ID;
    }
  } else {
      // Face does not exist;
  }
}


/**
 * Detect current face attributes through face frame Synchronization method
 * @param bitmap
 * @param mask  Face attributes to be detected eg:ArcternAttrResult.ARCTERN_FACE_ATTR_MASK_ALL
 * @return
 */
ArcternAttrResult arcternAttrResult = YTLFFaceManager.getInstance().getFaceAttrs(Bitmap bitmap, int
mask)
```

3.5 Extract face feature values from specified image files

> Extract face feature values based on detected faces, which can be used for face comparison;

```

/*Specify image files to extract face feature values:
  imagePath The map's address
  etractCallBack  Face feature extraction callback
*/
YTLFFaceManager.getInstance().doFeature(imagePath, new ExtractCallBack() {

/*
  acternImage Extract feature image
  bytes  Set of face feature values • • for multiple faces
  arcternRects  Face detection result set
*/
  @Override
```

T-Firefly.com

```
  public void onExtractFeatureListener(ArcternImage arcternImage, byte[][] bytes, ArcternRect[]
arcternRects) {

    if (features.length > 0) {
      debugLog("bitmapFeature.length：  " + features[0].length);
    } else {
      Tools.debugLog("Characteristic value is empty !");
    }
  });

/**
 * Search face information synchronization method
 * Search for the face with the highest comparison score in the database by feature value
 * @param feature
 * @return
 */
long id = doSearch(byte[] feature);

/*Extract facial feature value synchronization method
  bitmap Image bitmap
*/
ArcternFeatureResult featureResult = YTLFFaceManager.getInstance().doFeature(bitmap)
```

3.6 Image Bitmap for feature value extraction

> According to the face image Bitmap to extract the face feature value, it can be used for face comparison;

```

/*Specify face image Bitmap to extract face feature values:
  bitmap Picture Bitmap data
  etractCallBack  Face feature extraction callback
*/
YTLFFaceManager.getInstance().doFeature(Bitmap bitmap, new ExtractCallBack() {

/*
  acternImage Extract feature image
  bytes  Set of face feature values ・ ・for multiple faces
  arcternRects  Face detection result set
*/
  @Override
  public void onExtractFeatureListener(ArcternImage arcternImage, byte[][] bytes, ArcternRect[]
arcternRects) {

    if (features.length > 0) {
      debugLog("bitmapFeature.length：  " + features[0].length);
    } else {
      Tools.debugLog("Characteristic value is empty !");
    }
  });
```

3.7 Get the landmarks coordinates of the eyes, mouth, nose, etc. of the face

T-Firefly.com

> Extract facial feature values based on the Arctern Image of the face, which can be used to obtain the landmarks coordinates of the face, such as eyes, mouth, nose, etc.;

```
/*
  Specify Arctern Image to extract facial feature values, including landmarks
*/
ArcternAttrResult ArcternAttrResult = YTLFFaceManager.getInstance().doFeature(ArcternImage arcternImage))
```

3.8 Face feature comparison

> Compare the two feature values to get the similarity between the two face feature values;

```
/*
  feature1 First face feature value
  Feature2  Second face feature value

  return float :Return the similarity of face comparison
*/
float result = YTLFFaceManager.getInstance().doFeature(byte[] feature1,  byte[] feature2)
```

4、 Face database management

4.1 Face storage

> Add the extracted face feature values and related IDs to the SDK face database;

```
/*
  id Face ID
  feature  Eigenvalues ・ ・ of faces

  0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().dataBaseAdd(long id, byte[] feature)
```

4.2 Face library delete

> According to the specified ID, delete the face information of the SDK face database;

```
/*
  id Face ID

  0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().dataBaseDelete(long id)
```

T-Firefly.com

4.3 Face library update

> According to the specified ID, update the face information of the SDK face database;

```
/*
  id Face ID
  feature  Eigenvalues • • of faces

  0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().dataBaseUpdate(long id, byte[] feature)
```

4.4 Add feature values to faces in batches

> According to the specified multiple IDs, batch update multiple face information corresponding to the SDK face database;

```
/*
  id Face ID Array
  feature  Array of multiple face feature values

  0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().dataBaseAdd(long[] id, byte[][] feature)
```

4.5 Face data removal

> Delete all face information of SDK face database;

```
/*
  0 Means success, 1 means failure
*/
int result = YTLFFaceManager.getInstance().dataBaseClear()
```

5、Parameter Description

5.1 Image data: ArcternImage

T-Firefly.com

```
public static final int ARCTERN_IMAGE_FORMAT_GRAY = 0; //Grayscale
public static final int ARCTERN_IMAGE_FORMAT_RGB = 1; //RGB
public static final int ARCTERN_IMAGE_FORMAT_NV21 = 6; //NV21

public static final int ARCTERN_IMAGE_FORMAT_YUV420sp = ARCTERN_IMAGE_FORMAT_NV21;//YUV
NV21

public static final int ARCTERN_IMAGE_FORMAT_YUV420sp_HS =
ARCTERN_IMAGE_FORMAT_YUV420sp; //YUV420
```

Parametric	Type	Description
FrameID	long	Config.json document content
Image_format	int	Image data format
width	int	Image width
height	int	Image height
gdata	Byte[]	Image data

5.2 Face box: ArcternRect

Parametric	Type	Description
x	int	X coordinate of the upper left corner of the face
y	int	Y coordinate of the upper left corner of the face
width	int	Face frame width
height	int	Face frame height

5.3 Face attributes: ArcternAttribute

Parametric	Type	Description
Label	String	Attribute result, type eg: ArcternFaceAttrTypeEnum
confidence	Float	Attribute result score

5.4 Face attributes ArcternAttrResult

Parametric	Type	Description
arcternAttributes	ArcternAttribute[][]	Attribute result, two-dimensional array, type as above 5.3 Face attribute ArcternAttribute

T-Firefly.com

Parametric	Type	Description
landmark	Int[]	landmarkCoordinate

5.5 Face attributes ArcternFaceAttrTypeEnum

```
ArcternAttribute.ArcternFaceAttrTypeEnum.GENDER = 0;
 ArcternAttribute.ArcternFaceAttrTypeEnum.AGE = 1;
ArcternAttribute.ArcternFaceAttrTypeEnum.POSE_PITCH = 2;
 ArcternAttribute.ArcternFaceAttrTypeEnum.POSE_YAW = 3;
ArcternAttribute.ArcternFaceAttrTypeEnum.POSE_ROLL = 4;
ArcternAttribute.ArcternFaceAttrTypeEnum.QUALITY = 5;
 ArcternAttribute.ArcternFaceAttrTypeEnum.LIVENESS_IR = 6;
 ArcternAttribute.ArcternFaceAttrTypeEnum.IMAGE_COLOR = 7;
 ArcternAttribute.ArcternFaceAttrTypeEnum.FACE_MASK = 8;
```

Value	Description	label Value Int
GENDER	Gender	0: uncertain, 1: male, 2: female
AGE	Age	0: grayscale image, 1: color image
POSE_PITCH	Face angle X axis	Take the x axis as the center, face up and down type
POSE_YAW	Face angle Y axis	The center of the y axis, the type of face rotation left and right
POSE_ROLL	Face angle Center point	Centered on the center point, x-y plane rotation type
QUALITY	Face Quality	confidence Face Value
LIVENESS_IR	Infrared living body	confidence <0.5: non-living body, confidence >= 0.5: living body
IMAGE_COLOR	Picture type	0: grayscale image, 1: color image
FACE_MASK	Mask detection	0: no mask, 1: with mask

T-Firefly.com