uniapp 安卓虹软人脸识别免费版原生插件

目　录

使用方法

介绍

虹软人脸识别支持图片人脸识别（可识别网络图片），活体检测，离线识别，相机预览旋转，相机人脸识别，批量注册（支持网络图片）等，支持保存用户的id和名称

> 本插件是免费版SDK插件，仅支持安卓10(包含)以下版本，如果您想使用增值版SDK(支持安卓10以上版本)插件请点击这里

联系作者

关注微信公众号可联系作者

官方文档

https://ai.arcsoft.com.cn/manual/docs#/139

SDK版本

ArcFace v3.0

插件地址

https://ext.dcloud.net.cn/plugin?id=15096

权限

1. android.permission.READ_EXTERNAL_STORAGE
2. android.permission.READ_PHONE_STATE
3. android.permission.WRITE_EXTERNAL_STORAGE
4. android.permission.CAMERA

API使用

- 用法

  在需要使用插件的页面加载以下代码

```
const module = uni.requireNativePlugin("leven-arcFace-ArcFaceModule");
```

- 示例

```
    <template>
  <view>
    <uni-card title="虹软人脸识别原生插件">
      <button type="primary" @click="onlineActive">激活引擎</button>
      <button type="primary" @click="getImageFace">获取图片人脸信息</button>
      <button type="primary" @click="imageFaceRegister">图片注册人脸</button>
      <button type="primary" @click="clearFace">清空人脸库</button>
      <button type="primary" @click="deleteFace">删除人脸</button>
      <button type="primary" @click="getFace">获取人脸</button>
      <button type="primary" @click="getAllFace">获取所有人脸信息</button>
      <button type="primary" @click="batchRegister">批量注册</button>
      <button type="primary" @click="getFaceCount">获取注册的人脸数量</button>
    </uni-card>
  </view>
</template>

<script>
  const module = uni.requireNativePlugin("leven-arcFace-ArcFaceModule");
  export default {
    data() {
      return {

      }
    },
    methods: {
      // 激活引擎
      onlineActive() {
        module.activeEngine({
          appId: "7eHk2fnhwZ4aNHeXBrPceHS8K442TY7d27o1bvfGniod",
```

```
      sdkKey: "ax9B3CzadBJJ2W8LAHFDWk6JMMQ5bdBtfAwSN5joA4w"
    }, res => {
      console.log(res)
    })
  },
  // 获取图片人脸信息
  getImageFace() {
    module.getImageFace({
      //本地或网络url地址
      url: "/sdcard/DCIM/arcface/1.jpg",
    }, res => {
      console.log(res)
    })
  },
  // 图片注册人脸信息
  imageFaceRegister() {
    module.imageFaceRegister({
      //本地或网络url地址
      url: "/sdcard/DCIM/arcface/2.jpg",
      // 保存的id（可以不传该参数，默认时间戳）
      id: 123,
      //保存的姓名（可以不传该参数，默认时间戳）
      name: "leven",
      // 同一人是否可以多次注册，默认true
      registerMultiple: false
    }, res => {
      console.log(res)
    })
  },
  // 清空人脸库
  clearFace() {
    module.clearFace(res => {
      console.log(res)
    })
  },
  // 删除人脸
  deleteFace() {
    module.deleteFace({
      id: "123"
    }, res => {
      console.log(res)
    })
  },
  // 获取人脸
  getFace() {
    module.getFace({
      id: "123"
    }, res => {
      console.log(res)
    })
```

```
    },
    // 批量注册
    batchRegister() {
      module.batchRegister({
        // 同一人是否可以多次注册，默认true
        registerMultiple: false,
        list: [{
          //本地或网络url地址
          url: "/sdcard/DCIM/arcface/1.jpg",
          // 保存的id（可以不传该参数，默认时间戳）
          id: 10001,
          //保存的姓名（可以不传该参数，默认时间戳）
          name: "leven1"
        }, {
          //本地或网络url地址
          url: "http://www.yeyuboke.com/svga/2.jpg",
          // 保存的id（可以不传该参数，默认时间戳）
          id: 10002,
          //保存的姓名（可以不传该参数，默认时间戳）
          name: "leven2"
        }, {
          //本地或网络url地址
          url: "/sdcard/DCIM/arcface/3.jpg",
          // 保存的id（可以不传该参数，默认时间戳）
          id: 10003,
          //保存的姓名（可以不传该参数，默认时间戳）
          name: "leven3"
        }, {
          //本地或网络url地址
          url: "http://www.yeyuboke.com/svga/4.jpg",
          // 保存的id（可以不传该参数，默认时间戳）
          id: 10004,
          //保存的姓名（可以不传该参数，默认时间戳）
          name: "leven4"
        }, {
          //本地或网络url地址
          url: "/sdcard/DCIM/arcface/5.jpg",
          // 保存的id（可以不传该参数，默认时间戳）
          id: 10005,
          //保存的姓名（可以不传该参数，默认时间戳）
          name: "leven5"
        }, {
          //本地或网络url地址
          url: "/sdcard/DCIM/arcface/6.jpg",
          // 保存的id（可以不传该参数，默认时间戳）
          id: 10006,
          //保存的姓名（可以不传该参数，默认时间戳）
          name: "leven6"
        }]
      }, res => {
```

```
            console.log(res)
          })
        },
        // 获取所有人脸信息
        getAllFace() {
          module.getAllFace(res => {
            console.log(res)
          })
        },
        // 获取注册的人脸数量
        getFaceCount() {
          module.getFaceCount(res => {
            console.log(res)
          })
        }
      }
    }
</script>

<style>

</style>
```

人脸识别组件使用

- 用法

  在需要使用插件的页面添加以下代码

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" :camera="c
    @onCameraOpened="onCameraOpened" @onCameraClosed="onCameraClosed" @onFaceR
```

- 示例

```
    <template>
  <view>
    <uni-card title="人脸识别">
      <view style="flex:1; height: 500px; position: relative;">
        <leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" :c
          @onCameraOpened="onCameraOpened" @onCameraClosed="onCameraClosed" @o
        </leven-arcFace>
        <!-- 组件内部自定义内容 -->
        <cover-view style="position: absolute; left: 0; top: 0;">
          <view><text>这里是自定义文字内容</text></view>
        </cover-view>
      </view>
```

```
      <button type="primary" @click="register">注册人脸</button>
      <button type="primary" @click="switchCamera">切换相机</button>
      <button type="primary" @click="stop">关闭预览</button>
      <button type="primary" @click="start">开启预览</button>
      <button type="primary" @click="closeFace">关闭人脸检测</button>
      <button type="primary" @click="openFace">开启人脸检测</button>
    </uni-card>
  </view>
</template>

<script>
  export default {
    data() {
      return {
        // 摄像机配置，所有的参数都可以不传，不传则按默认的
        camera: {
          // 相机预览旋转角度
          rotation: 0,
          //相机预览模式，0:后置，1:前置(默认)
          facing: 1,
          //预览分辨率
          // size: [800, 600],
          // 摄像机预览圆角
          radius: 50,
          // 是否开启预览，默认：true
          preview: true
        },
        // 视频检测配置，所有参数都可以不传，不传则按默认的
        video: {
          // 视频检测角度，可接收参数，0,90,180,270(默认),360(全方位检测)
          orient: 360,
          // 是否进行活体检测(默认为true)
          liveness: true,
          // 人脸注册同一人是否可以多次注册（默认：true）
          registerMultiple: false,
          // 人脸识别成功后是否展示左上角人脸识别图片（默认：true）
          showIdentifyImage: false,
          // 人脸框是否处于X反向状态，如果未设置该参数人脸框和人脸处于反向请将该参数设置为t
          // isContraryX: true,
          // 人脸框是否处于Y反向状态，如果未设置该参数人脸框和人脸处于反向请将该参数设置为t
          // isContraryY: true,
          // 识别阈值
          similar: 0.8,
          // 识别的最小人脸比例，如果失败比较敏感可以适当调小，默认：16
          detectFaceScaleVal: 10
        }
      }
    },
    methods: {
      // 注册人脸
```

```
register() {
  if (this.$refs.refLevenArcFace) {
    this.$refs.refLevenArcFace.register({
      // 注册后保存的id（可以不传该参数，默认时间戳）
      id: "456",
      //注册后保存的名字（可以不传该参数，默认时间戳）
      name: "leven1"
    }, res => {
      console.log(res)
    });
  }
},
// 切换相机
switchCamera() {
  if (this.$refs.refLevenArcFace) {
    this.$refs.refLevenArcFace.switchCamera(res => {
      console.log(res)
    });
  }
},
// 关闭预览
stop() {
  if (this.$refs.refLevenArcFace) {
    this.$refs.refLevenArcFace.stop(res => {
      console.log(res)
    });
  }
},
// 开启预览
start() {
  if (this.$refs.refLevenArcFace) {
    this.$refs.refLevenArcFace.start(res => {
      console.log(res)
    });
  }
},
// 关闭人脸检测
closeFace() {
  if (this.$refs.refLevenArcFace) {
    this.$refs.refLevenArcFace.closeFace(res => {
      console.log(res)
    });
  }
},
// 开启人脸检测
openFace() {
  if (this.$refs.refLevenArcFace) {
    this.$refs.refLevenArcFace.openFace(res => {
      console.log(res)
    });
```

```
          }
        },
        // 错误事件
        onError(e) {
          console.log(e)
        },
        // 相机打开事件
        onCameraOpened(e) {
          console.log(e)
        },
        // 相机关闭事件
        onCameraClosed(e) {
          console.log(e)
        },
        // 相机配置改变事件
        onCameraConfigurationChanged(e) {
          console.log(e)
        },
        // 人脸识别结果
        onFaceResult(e) {
          console.log(e)
        }
      }
    }
</script>

<style>

</style>
```

更新日志

2024-09-13 `v1.2.1`

1. 更新虹软sdk

2024-06-27 `v1.2.0`

1. [优化] 优化删除人脸
2. [新增] 人脸识别组件新增方法【监听识别结果】
3. [新增] 组件属性的视频检测属性新增参数【识别结果是否返回当前识别的人脸图片】
4. [新增] 识别结果新增返回字段 `recognitionImage` 当前识别的人脸图片base64格式

2023-12-18 `v1.1.3`

1. [新增]视频检测属性新增是否展示人脸识别框文本内容
2. [优化]更新视频检测属性方法新增参数是否展示人脸识别框文本内容

2023-11-30 `v1.1.2`

1. [优化] 人脸识别成功返回相似度
2. [优化] 关闭预览

2023-11-12 `v1.1.1`

[优化] 优化批量注册和单个注册闪退问题

2023-11-05 `v1.1.0`

1. [新增]人脸识别组件新增接口【更新摄像机属性】【更新视频检测属性】【重载组件】
2. [优化]优化人脸识别组件开启和关闭人脸检测
3. [优化]优化批量注册
4. [新增]API接口新增【停止批量注册】

2023-10-30

1. [新增]人脸识别组件新增接口【关闭人脸检测】【开启人脸检测】
2. [新增]人脸识别组件camera属性新增 摄像机预览圆角设置 是否开启预览
3. [新增]人脸识别组件video属性新增 识别阈值 识别的最小人脸比例
4. [优化]优化批量注册

2023-10-24

首次发布

激活引擎

方法名

`activeEngine`

用法

- 用法如下：

```
module.activeEngine({
  appId: "7eHk2fnhwZ4aNHeXBrPceHS8K442TY7d27o1bvfGniod",
  sdkKey: "ax9B3CzadBJJ2W8LAHFDWk6JMMQ5bdBtfAwSN5joA4w"
}, res => {
  console.log(res)
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
appId	String	是	无	虹软申请的appId
sdkKey	String	是	无	虹软申请的sdkKey

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {
        "sdkType": "ArcFace",
        "sdkKey": "ax9B3CzadBJJ2W8LAHFDWk6JMMQ5bdBtfAwSN5joA4w",
        "fileVersion": "2.0",
        "appId": "7eHk2fnhwZ4aNHeXBrPceHS8K442TY7d27o1bvfGniod",
        "endTime": "1726185600",
        "startTime": "1694563200",
        "platform": "android"
```

```
    }
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.sdkType	String	SDK类型
data.sdkKey	String	SDK_KEY
data.fileVersion	String	激活文件版本号
data.appId	String	APP_ID
data.endTime	String	SDK截止时间
data.startTime	String	SDK开始时间
data.platform	String	平台版本
code	Integer	返回类型，0.成功，其他：失败

获取图片人脸信息

方法名

`getImageFace`

用法

- 用法如下：

```
module.getImageFace({
  //本地或网络url地址
  url: "/sdcard/DCIM/arcface/1.jpg",
}, res => {
  console.log(res)
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
url	String	是	无	本地或网络url地址

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {
        "list": [
            {
                "gender": 0,
                "faceFeature": "AID6RAAAoEFK1iq7ZYzDvG/FdD2Yyzi9VV+aOp1oZLxrcF
                "face3D": {
                    "yaw": 0.44348010420799255,
                    "roll": 0.7263081073760986,
                    "pitch": -3.520747423171997,
                    "status": 0
                },
```

```
                "age": 29,
                "liveness": -3
            }
        ]
    }
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	String	人脸列表
data.list.gender	Integer	性别，0：男性，1：女性，-1：未知
data.list.faceFeature	String	人脸特征数据，base64加密
data.list.face3D	Object	3D角度信息
data.list.face3D.yaw	Float	偏航角
data.list.face3D.roll	Float	横滚角
data.list.face3D.pitch	Float	俯仰角
data.list.face3D.status	Integer	0: 正常，非0: 异常
data.list.age	Integer	年龄
data.list.liveness	Integer	活体信息，0：非真人，1.真人，-1.不确定，-2.传入人脸数 > 1，-3. 人脸过小，-4.角度过大，-5.人脸超出边界
code	Integer	返回类型，0.成功，其他：失败

图片注册人脸

方法名

`imageFaceRegister`

用法

- 用法如下：

```
module.imageFaceRegister({
  //本地或网络url地址
  url: "/sdcard/DCIM/arcface/2.jpg",
  // 保存的id（可以不传该参数，默认时间戳）
  id: 123,
  //保存的姓名（可以不传该参数，默认时间戳）
  name: "leven",
  // 同一人是否可以多次注册，默认true
  registerMultiple: false
}, res => {
  console.log(res)
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
url	String	是	无	本地或网络url地址
id	String	否	时间戳	保存的id
name	String	否	时间戳	保存的名称
registerMultiple	Boolean	否	true	同一个人脸是否可以多次注册

回调

- 示例

```
{
    "message": "",
```

```
    "code": 0,
    "data": {
        "head": "/data/data/test.arcface.com/files/register/imgs/123##leven.jp
    }
}
```


- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.head	String	注册后本地保存的头像路径
code	Integer	返回类型，0.成功，其他：失败

清空人脸库

方法名

`clearFace`

用法

- 用法如下：

```
module.clearFace(res => {
  console.log(res)
})
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {
        "deleteCount": 1
    }
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.deleteCount	Integer	删除的人脸数量
code	Integer	返回类型，0.成功，其他：失败

删除人脸

方法名

`deleteFace`

用法

- 用法如下：

```
module.deleteFace({
  id: "123"
}, res => {
  console.log(res)
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
id	String	是	无	保存的id

回调

- 示例

```
{
    "message": "删除成功",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

获取人脸

- 用法如下：

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
id	String	是	无	保存的id

- 示例

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.head	String	图片路径
data.featureData	String	人脸特征数据，base64加密
data.id	String	保存的id
data.name	String	保存的名称
code	Integer	返回类型，0.成功，其他：失败

获取所有人脸信息

方法名

`getAllFace`

用法

- 用法如下：

```
module.getAllFace(res => {
  console.log(res)
})
```

- 参数说明

  无

回调

- 示例


```
{
    "message": "",
    "code": 0,
    "data": {
        "list": [
            {
                "head": "/data/data/test.push.yingjyun.com/files/register/imgs
                "featureData": "AID6RAAAoEEGsKU85Ddwva2l1bvFzZe98AJjvULgVL11v4
                "id": "123",
                "name": "leven"
            }
        ]
    }
}
```


- 回调说明：

获取所有人脸信息

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	Array	人脸列表
data.list.head	String	图片路径
data.list.featureData	String	人脸特征数据，base64加密
data.list.id	String	保存的id
data.list.name	String	保存的名称
code	Integer	返回类型，0.成功，其他：失败

批量注册

方法名

`batchRegister`

用法

- 用法如下：

```
module.batchRegister({
  // 同一人是否可以多次注册，默认true
  registerMultiple: false,
  list: [{
    //本地或网络url地址
    url: "/sdcard/DCIM/arcface/1.jpg",
    // 保存的id（可以不传该参数，默认时间戳）
    id: 10001,
    //保存的姓名（可以不传该参数，默认时间戳）
    name: "leven1"
  }, {
    //本地或网络url地址
    url: "http://www.yeyuboke.com/svga/2.jpg",
    // 保存的id（可以不传该参数，默认时间戳）
    id: 10002,
    //保存的姓名（可以不传该参数，默认时间戳）
    name: "leven2"
  }, {
    //本地或网络url地址
    url: "/sdcard/DCIM/arcface/3.jpg",
    // 保存的id（可以不传该参数，默认时间戳）
    id: 10003,
    //保存的姓名（可以不传该参数，默认时间戳）
    name: "leven3"
  }, {
    //本地或网络url地址
    url: "http://www.yeyuboke.com/svga/4.jpg",
    // 保存的id（可以不传该参数，默认时间戳）
    id: 10004,
    //保存的姓名（可以不传该参数，默认时间戳）
    name: "leven4"
  }, {
    //本地或网络url地址
    url: "/sdcard/DCIM/arcface/5.jpg",
    // 保存的id（可以不传该参数，默认时间戳）
```

```
    id: 10005,
    //保存的姓名（可以不传该参数，默认时间戳）
    name: "leven5"
  }, {
    //本地或网络url地址
    url: "/sdcard/DCIM/arcface/6.jpg",
    // 保存的id（可以不传该参数，默认时间戳）
    id: 10006,
    //保存的姓名（可以不传该参数，默认时间戳）
    name: "leven6"
  }]
}, res => {
  console.log(res)
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
registerMultiple	Boolean	否	true	同一人脸是否可以多次注册
list	Array	是	无	注册列表
list.url	String	是	无	注册的图片本地或网络地址
list.id	String	否	时间戳	注册时保存的id
list.name	String	否	时间戳	注册时保存的名称

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {
        "successCount": 6,
        "head": "/data/data/test.push.yingjyun.com/files/register/imgs/10004##
        "failedCount": 0,
        "id": "10004",
        "name": "leven4",
        "progress": 0.9993098688750862,
        "url": "http://www.yeyuboke.com/svga/1.jpg",
```

```
        "status": "registering"
    }
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.successCount	Integer	当前已成功注册的数量
data.head	String	当前注册的人脸图片路径
data.failedCount	Integer	当前注册失败的数量
data.id	String	保存的id
data.name	String	保存的名称
data.progress	Float	当前注册的进度
data.url	String	当前注册的url
data.status	String	当前注册的状态，registering：注册中，end：注册完成
code	Integer	返回类型，0.成功，其他：失败

获取注册的人脸数量

方法名

`getFaceCount`

用法

- 用法如下：

```
module.getFaceCount(res => {
  console.log(res)
})
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {
        "count": 6
    }
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.count	Integer	人脸数量
code	Integer	返回类型，0.成功，其他：失败

停止批量注册 v1.1.0

方法名

`stopBatchRegister`

用法

- 用法如下：

```
module.stopBatchRegister(res => {
  console.log(res)
})
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.count	Integer	人脸数量
code	Integer	返回类型，0.成功，其他：失败

识别组件

> 仅支持在nvue下使用

```
```

> 组件中具体的方法、属性以及事件请参考各自文档

组件方法

相机注册人脸

- 用法如下：

```
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
id	String	否	时间戳	注册后保存的id
name	String	否	时间戳	注册后保存的名字

- 示例

```
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

切换相机

方法名

`switchCamera`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  this.$refs.refLevenArcFace.switchCamera(res => {
    console.log(res)
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

关闭预览

方法名

`stop`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  this.$refs.refLevenArcFace.stop(res => {
    console.log(res)
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

开启预览

方法名

`start`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  this.$refs.refLevenArcFace.start(res => {
    console.log(res)
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

关闭人脸检测 v1.0.1

方法名

`closeFace`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  this.$refs.refLevenArcFace.closeFace(res => {
    console.log(res)
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

开启人脸检测 v1.0.1

方法名

`openFace`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  this.$refs.refLevenArcFace.openFace(res => {
    console.log(res)
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

更新摄像机属性 v1.1.0

方法名

`setCamera`

> 相机属性设置后续调用一次重载组件方法后才能生效

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  let params = JSON.parse(JSON.stringify(this.camera));
  params.rotation = 270;
  params.facing = 1;
  params.radius = 10;
  params.size = [800, 600];
  params.preview = true;
  this.$refs.refLevenArcFace.setCamera(params, res => {
    console.log(res)
  });
}
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
rotation	Integer	否	无	相机旋转角度
facing	Integer	否	1	相机预览模式，0:后置，1:前置(默认)
radius	Integer	否	0	组件圆角值
size	Array	否	无	预览分辨率
preview	Boolean	否	true	是否开启预览

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

更新视频检测属性 v1.1.0

方法名

`setVideo`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  let params = JSON.parse(JSON.stringify(this.video));
  params.orient = 0;
  params.liveness = false;
  params.registerMultiple = false;
  params.showIdentifyImage = true;
  params.isContraryX = true;
  params.isContraryY = true;
  params.similar = 0.9;
  params.detectFaceScaleVal = 8; //此属性设置后需调用一次重载组件方可生效
  params.showIdentifyText = true;
  this.$refs.refLevenArcFace.setVideo(params, res => {
    console.log(res)
  });
}
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
orient	Integer	否	无	视频检测角度，可接收参数，0,90,180,270(默认),360(全方位检测)
liveness	Boolean	否	true	是否进行活体检测
registerMultiple	Boolean	否	true	人脸注册同一人是否可以多次注册

更新视频检测属性 v1.1.0

参数名	参数类型	是否必填	默认值	参数描述
showIdentifyImage	Boolean	否	true	人脸识别成功后是否展示左上角人脸识别图片
isContraryX	Boolean	否	false	人脸框是否处于X反向状态
isContraryY	Boolean	否	false	人脸框是否处于Y反向状态
similar	Float	否	0.8	识别阈值
detectFaceScaleVal	Integer	否	16	识别的最小人脸比例

> detectFaceScaleVal 设置后需要调用一次重载组件方可生效，其他参数实时生效

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

重载组件 v1.1.0

方法名

`reInit`

用法

- 用法如下：

```
if (this.$refs.refLevenArcFace) {
  this.$refs.refLevenArcFace.reInit(res => {
    console.log(res)
  });
}
```

- 参数说明

  无

回调

- 示例

```
{
    "message": "",
    "code": 0,
    "data": {}
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

监听识别结果 vue3 v1.2.0

方法名

> 由于vue3对组件的事件返回不支持，该方法可以再vue3中使用，用于注册人脸识别结果，识别后的返回参数可参考【组件事件】> 【人脸识别结果】

用法

- 用法如下：

```
```

- 参数说明
  无

回调

- 示例

```
```

- 回调说明：
  请参考【组件事件】> 【人脸识别结果】

组件属性

相机属性需要在组件主题中使用，如：

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" :camera="came
```

属性配置可以放到data中，如：

```
data() {
      return {
        // 摄像机配置，所有的参数都可以不传，不传则按默认的
        camera: {
          // 相机预览旋转角度
          rotation: 0,
          //相机预览模式，0:后置，1:前置(默认)
          facing: 1,
          //预览分辨率
          // size: [800, 600]
        },
        // 视频检测配置，所有参数都可以不传，不传则按默认的
        video: {
          // 视频检测角度，可接收参数，0,90,180,270(默认),360(全方位检测)
          orient: 360,
          // 是否进行活体检测(默认为true)
          liveness: true,
          // 人脸注册同一人是否可以多次注册（默认：true）
          registerMultiple: false,
          // 人脸识别成功后是否展示左上角人脸识别图片（默认：true）
          showIdentifyImage: false,
          // 人脸框是否处于X反向状态，如果未设置该参数人脸框和人脸处于反向请将该参数设置为true
          // isContraryX: true,
          // 人脸框是否处于Y反向状态，如果未设置该参数人脸框和人脸处于反向请将该参数设置为true
          // isContraryY: true
        }
      }
    },
```

摄像机配置

属性名称 `camera`
具体属性参数请参考【摄像机配置】

视频检测配置

属性名称 `video`
具体属性参数请参考【视频检测配置】

摄像机配置

摄像机配置参数，所有的参数都可以不传，即：该属性可以不用设置，如果不设置则按照默认处理

相机旋转角度

参数名称： `rotation`

相机预览模式

参数名称： `facing`

> 0:后置，1:前置(默认)

预览分辨率

参数名称： `size`

> 使用示例：[800, 600]

摄像机预览圆角 `v1.0.1`

参数名称： `radius`

> 默认：0

是否开启预览 `1.0.1`

参数名称： `preview`

> 初始化组件时是否默认开启预览，默认：true

视频检测配置

视频检测配置参数，所有的参数都可以不传，即：该属性可以不用设置，如果不设置则按照默认处理

视频检测角度

参数名称： `orient`

> 可接收参数，0,90,180,270(默认),360(全方位检测)

是否进行活体检测

参数名称： `liveness`

> 默认为true

是否可以多次注册

参数名称： `registerMultiple`

> 人脸注册同一人是否可以多次注册（默认：true）

是否展示左上角人脸识别图片

参数名称： `showIdentifyImage`

> 人脸识别成功后是否展示左上角人脸识别图片（默认：true）

人脸框是否处于X反向状态

参数名称： `isContraryX`

> 人脸框是否处于X反向状态，如果未设置该参数人脸框和人脸处于反向请将该参数设置为true

人脸框是否处于Y反向状态

参数名称： `isContraryY`

> 人脸框是否处于Y反向状态，如果未设置该参数人脸框和人脸处于反向请将该参数设置为true

识别阈值 `v1.0.1`

参数名称： `similar`

> 默认：0.8

识别的最小人脸比例 `v1.0.1`

参数名称： `detectFaceScaleVal`

> 默认：16，取值范围1-32，识别的最小人脸比例，如果失败比较敏感可以适当调小

是否展示识别结果文字 `v1.1.3`

参数名称： `showIdentifyText`

> 默认：true，是否展示人脸识别框中文本

识别结果是否返回当前识别的人脸图片 `v1.2.0`

参数名称： `isRecognitionImage`

> 默认：false，返回结果为base64格式

组件事件

错误事件

事件名称

`onError`

用法

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" @onError="onE
```

返回示例

```
onError(e) {
    console.log(e.detail)
  }
```

回调说明：

参数名	参数类型	参数描述
message	String	消息提示
code	Integer	虹软的错误码

相机打开事件

事件名称

`onCameraOpened`

用法

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" @onCameraOpene
```

返回示例

```
onCameraOpened(e) {
    console.log(e.detail)
  }
```

回调说明：

参数名	参数类型	参数描述
message	String	消息提示

相机关闭事件

事件名称

`onCameraClosed`

用法

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" @onCameraClose
```

返回示例

```
onCameraClosed(e) {
    console.log(e.detail)
  }
```

回调说明：

参数名	参数类型	参数描述
message	String	消息提示

相机配置改变事件

事件名称

`onCameraConfigurationChanged`

用法

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" @onCameraConf
```

返回示例

```
onCameraConfigurationChanged(e) {
    console.log(e.detail)
  }
```

回调说明：

参数名	参数类型	参数描述
cameraID	Integer	相机id
displayOrientation	Integer	相机旋转角度

人脸识别结果

事件名称

`onFaceResult`

用法

```
<leven-arcFace ref="refLevenArcFace" style="flex:1; height: 500px;" @onFaceResult=
```

示例

```
onFaceResult(e) {
    console.log(e.detail)
  }
```

回调示例

```
"detail": {
        "head": "/data/data/test.push.yingjyun.com/files/register/imgs/456##leven1
        "status": 1,
        "message": "",
        "id": "456",
        "name": "leven1"
    },
```

回调说明：

参数名	参数类型	参数描述
head	String	注册时的头像
status	Integer	识别结果，1.识别成功，其他.识别失败

人脸识别结果

参数名	参数类型	参数描述
message	String	消息提示
id	String	注册时的id
name	String	注册时的名称
similar	Float	当前失败的相似度
headBase64 `v1.2.0`	String	注册时的头像的base64格式
recognitionImage `v1.2.0`	String	当前识别的人脸图片base64格式，需在再视频检测属性中将isRecognitionImage参数设置为true