4.3 界面设计

4.3.1 首页

首页是整个程序的入口，页面包括了应用名称和两个主要功能的卡片式的按钮，其中“换底”按钮用于将证件照背景替换为其他图片或颜色，而“证件照制作”按钮用于制作个人证件照。

图 4-5 界面-首页

4.3.2 图片选择-相机

图片选择界面是用于用户选择或采集图片的界面，其中包括了图像采集功能，即使用相机采集图片。页面的顶部包含了应用名称和返回按钮，方便用户返回上一级页面。页面的中央部分是图片展示区域，用户可以在此浏览和选择已有的图片。同时，在页面底部添加了相机按钮，用户可以点击此按钮进入相机界面进行图像采集。预览区域按钮点击可以切换摄像头。

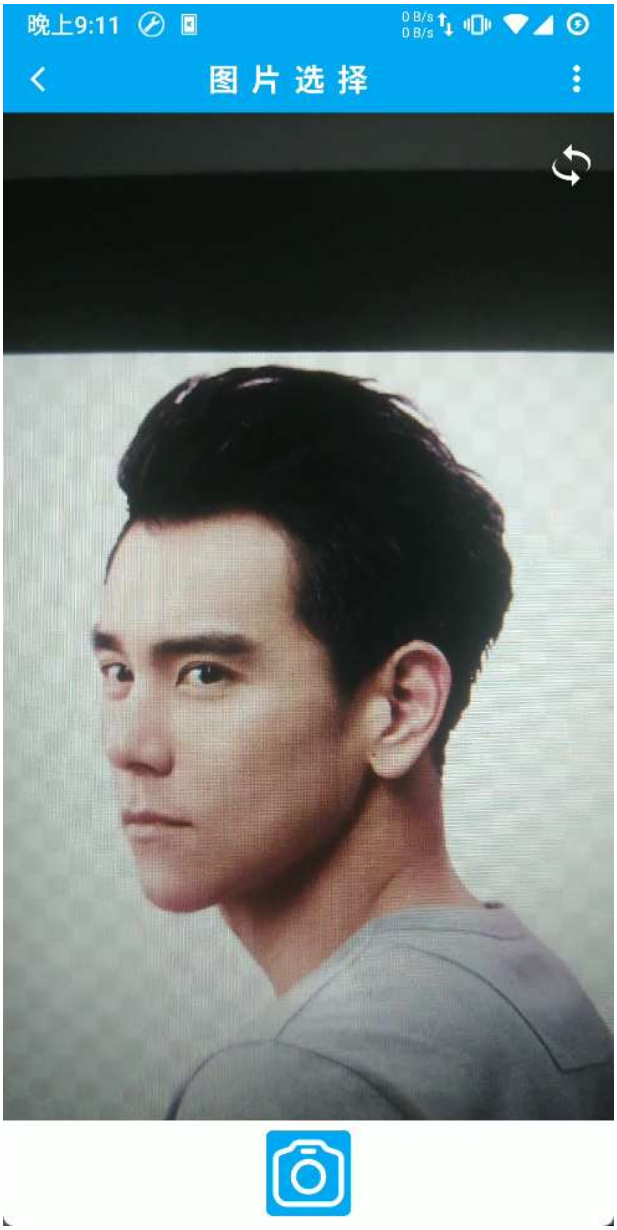

图 4-6 界面-图片选择-拍照

4.3.3 图像选择-从图库获取

图片选择界面是用于用户选择或采集图片的界面，其中包括了从图库获取图片功能。页面的顶部包含了应用名称和返回按钮，方便用户返回上一级页面。页面的中央部分是图片展示区域，用户可以在此浏览和选择已有的图片。

图 4-7 界面-图片选择-从图库获取

4.4.4 图像编辑页面-换底

图像编辑页面是用户进行证件照制作的核心界面，其中包括了换底功能。

在本文中，我们实现了红、蓝、白三种颜色的换底功能，用户可以选择不同的底色来进行证件照的制作。

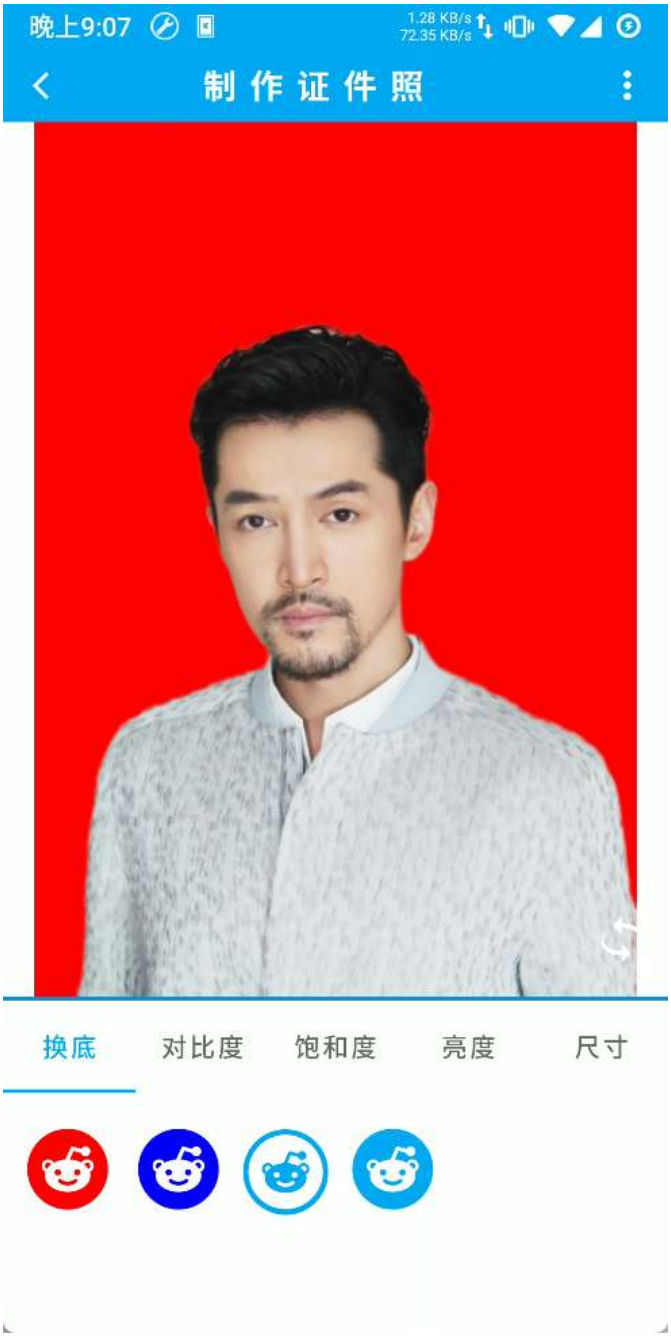

图 4-8 界面-图片编辑-换底

4.4.5 图像编辑页面-对比度调节

对比度调整界面，我们将对比度调整功能放置在图像编辑页面，通过一个滑动条来控制对比度的调整。滑动条的初始位置为 0，表示原始的对比度。用户可以通过左右滑动滑动条来调整对比度的数值，实时预览调整效果。

图 4-9 界面-图片编辑-对比度

4.4.6 图像编辑页面-饱和度调节

饱和度调整界面，我们将饱和度调整功能放置在图像编辑页面，通过一个滑动条来控制对比度的调整。滑动条的初始位置为 0，表示原始的饱和度。用户可以通过左右滑动滑动条来调整饱和度的数值，实时预览调整效果。

图 4-10 界面-图片编辑-饱和度

4.4.7 图像编辑页面-亮度调节

亮度调整界面，我们将亮度调整功能放置在图像编辑页面，通过一个滑动条来控制对比度的调整。滑动条的初始位置为 0，表示原始的亮度。用户可以通过左右滑动滑动条来调整亮度的数值，实时预览调整效果。

图 4-11 界面-图片编辑-亮度

4.4.8 图像编辑页面-尺寸调节

尺寸调节界面，通过一个卡片列表来选择调节的尺寸，包含了常见的证件照尺寸[10]，例如 1 寸、2 寸等等，用户可以根据需要选择相应的尺寸。

图 4-12 界面-图片编辑-尺寸调节

5 软件实现

5.1 开发环境和工具

Android Studio：开发 Android 应用的官方 IDE，集成了 Android SDK 工具包和便捷的界面设计工具。

OpenCV SDK：进行图像处理的库文件，需要进行下载并配置。

Kotlin：Android 前端开发的主要语言，无需额外配置。

C++：进行 OpenCV 图像处理的后端开发的主要语言，需要使用 Android NDK 进行原生开发。

Android NDK：用于支持 C++ 开发的工具，需要进行相关配置和引入。

5.2 功能实现

5.2.1 图片选择-从相机获取

图片选择-从相机获取，用于从相机中获取图片。在本软件中，该功能是通过第三方库 CameraView 实现的。我们通过 CameraView 库进行相机的监听，并在用户点击拍照按钮时获取相机数据。用户可以通过点击首页上的入口按钮进入该页面，然后点击中间的拍照按钮，就可以进行图片截取。 截取后我们通过 OpenCV 进行人脸检测，并判断选择的图片是否符合检测条件。如果检测到该图片中不存在人脸，那么将会禁止掉下一步的按钮。

主要代码如下：

```
/**
 * 添加 camera view 的监听
 */
binding.photoSelectCameraview.addCameraListener(object : CameraListener() {
    override fun onPictureTaken(result: PictureResult) {
        Log.d(
            TAG,
            "onPictureTaken: have taken picture:
${(System.currentTimeMillis() - time) / 1000f}s"
        )
        time = System.currentTimeMillis()
        // 拍照 获取 bitmap
        result.toBitmap {
            Log.d(
                TAG,
                "onPictureTaken: have convert to bitmap:
${(System.currentTimeMillis() - time) / 1000f}s"
            )
            time = System.currentTimeMillis()
            it ?: return@toBitmap
```

```
            originBitmap = when (binding.photoSelectCameraview.facing) {
                Facing.FRONT -> {
                    // 前置摄像头 拍摄会 画面翻转 需经过再次翻转 改善视觉体验
                    Bitmap.createBitmap(
                        it,
                        0,
                        0,
                        it.width,
                        it.height,
                        Matrix().apply { postScale(-1f, 1f) },
                        false
                    )
                }
                Facing.BACK -> it
            }
            lifecycleScope.launch {
                compressPhoto()
                Log.d(
                    TAG,
                    "onPictureTaken: have been compressed at
${(System.currentTimeMillis() - time) / 1000f}s"
                )
                toggleDisplay(false)
            }
        }
    }
})
```

5.2.2 图片选择-从图库获取

图片选择-从图库获取，用于从图库中获取图片。在本软件中，我们通过 Android 系统提供的 Intent.ACTION_PICK[14]进行调用，打开系统图库，然后用户可以选择想要的图片并返回到应用中。用户可以通过点击首页上的入口按钮进入该页面，然后从图库中选择自己喜欢的图片。获取到图片后我们通过 OpenCV 进行人脸检测，并判断选择的图片是否符合检测条件。如果检测到该图片中不存在人脸，那么将会禁止掉下一步的按钮。

主要代码如下：

```
/**
 * 从图库获取照片
 */
private var url: String? = null
private val getContent =
registerForActivityResult(ActivityResultContracts.GetContent()) {
    it ?: return@registerForActivityResult
    requireActivity().getImageFileFromUri(it)!!.run {
```

```
            url = absolutePath
            bitmap = BitmapFactory.decodeFile(url)
            if (bitmap!!.width > bitmap!!.height) {
                val matrix = Matrix()
                matrix.setRotate(-90f)
                bitmap = Bitmap.createBitmap(
                    bitmap!!,
                    0,
                    0,
                    bitmap!!.width,
                    bitmap!!.height,
                    matrix,
                    true
                )
            }
            lifecycleScope.launch {
                compressPhoto()
                toggleDisplay(false)
            }
        }
    }
```

5.2.3 图像处理-对比度

图像处理-对比度，用于增强或减弱图片的对比度。在本软件中，我们通过使用 Android 系统自带的 Bitmap 类，对图片的像素进行修改，从而改变图片的对比度。用户可以在编辑页面上，通过滑动对比度条来调整图片的对比度。

主要代码如下：

```
private var alpha = 50
private var alphaPhoto = object : IPhotoParamsChangedListener {
    override fun onChanged(slider: Slider?, value: Float, index: Int) {
        alpha = value.toInt()
        setLightAndAlpha()
    }
}
/**
 * 设置亮度 对比度
 */
private fun setLightAndAlpha() {
    if (System.currentTimeMillis() - time < 300 || isOpting) return
    setOptBit()
    lifecycleScope.launch {
        var bitmap: Bitmap? = null
        isOpting = true
        withContext(Dispatchers.IO) {
            bitmap = OpencvUtils.setLightAndAlpha(optBitmap!!, beta,
alpha)!!
        }
        Log.d("TAG_", "setLightAndAlpha: success ")
        optComplete(bitmap)
        isOpting = false
```

```
            time = System.currentTimeMillis()
        }
    }
```

5.2.4 图像处理-亮度

图像处理-亮度，用于增强或减弱图片的亮度。在本软件中，我们同样通过使用Android 系统自带的 Bitmap 类，对图片的像素进行修改，从而改变图片的亮度。用户可以在编辑页面上，通过滑动亮度条来调整图片的亮度。

主要代码如下：

```
private var beta = 50
private var lightPhoto = object : IPhotoParamsChangedListener {
    override fun onChanged(slider: Slider?, value: Float, index: Int) {
        beta = value.toInt()
        setLightAndAlpha()
    }
}
```

5.2.5 图像处理-饱和度

图像处理-饱和度是一个小功能，用于增强或减弱图片的饱和度。在本软件中，我们同样通过使用 Android 系统自带的 Bitmap 类，对图片的像素进行修改，从而改变图片的饱和度。用户可以在编辑页面上，通过滑动饱和度条来调整图片的饱和度。

主要代码如下：

```
/**
 * 改变饱和度
 */
private var changeFullColor = object : IPhotoParamsChangedListener {
    override fun onChanged(slider: Slider?, value: Float, index: Int) {
        if (System.currentTimeMillis() - time < 300 || isOpting) return
        setOptBit()
        lifecycleScope.launch {
            var bitmap: Bitmap? = null
            isOpting = true
            withContext(Dispatchers.IO) {
                bitmap = OpencvUtils.adjustImageSaturation(optBitmap!!,
value)
            }
            Log.d("TAG_", "changeFullColor: success ")
            optComplete(bitmap)
            isOpting = false
            time = System.currentTimeMillis()
        }
    }
}
```

5.2.6 图像处理-尺寸

图像处理-尺寸是一个小功能，用于改变图片的尺寸。在本软件中，我们同样通过使用 Android 系统自带的 Bitmap 类，对图片进行缩放，从而改变图片的尺寸。用户可以在编辑页面上，通过手动输入或者拖动滑动条来调整图片的尺寸。

主要代码如下：

```
    private var pSize: PhotoSize? = null
    private var sizeChanged = object : SizePEditor.OnSizeClickListener {
        override fun onSizeClick(index: Int, size: PhotoSize) {
            setOptBit()
            binding.fpePhoto.resetZoom()
            pSize = size
//            var sc = size.wPixel * 1f / optBitmap!!.width
            try{
                optComplete(
                    OpencvUtils.cutBitmap(
                        optBitmap!!, Rect( 0, 0,  size.wPixel, size.hPixel
                        ),BitmapUtils.scalarToColor(color)
                    )
                );
            }catch (e:Exception)
            {
                e.printStackTrace()
                requireActivity().showToast("Error")
            }
        }
    }
```

5.2.7 图像处理-换底

图像处理-换底是本软件的重要功能之一，通过该功能用户可以将照片的背景换成想要的背景。在本软件中，我们首先使用了腾讯云服务提供的 BDA（腾讯云图像处理）进行人像切割，以此来提高图像处理的准确性和效率。然后我们再使用 Opencv 库对切割后的图像进行换底操作，用户可以选择红、蓝、白三种背景色。

主要代码如下：

```
private suspend fun changeColor(): Bitmap? {
    var bt: Bitmap? = null
    val loading = LoadingUtil.showLoading(requireContext(), "正在更换底色
中...")
    withContext(Dispatchers.IO) {
        try {
            val segmentPortraitPicRequest = SegmentPortraitPicRequest()
            segmentPortraitPicRequest.scene = "GEN"
```

```
            // 设置操作图片
            segmentPortraitPicRequest.image =
BitmapUtils.bitmapToBase64(mBitmap!!)
            segmentPortraitPicRequest.rspImgType = "base64"
            Log.d(TAG, "changeColor: 正在使用腾讯云切图。。。。")
            val segmentPortraitPic =

aliyunPhotoEdit.SegmentPortraitPic(segmentPortraitPicRequest)
            bt =
BitmapUtils.base64ToBitmap("${segmentPortraitPic?.resultImage}")
            Log.d(TAG, "changeColor: 腾讯云切图完毕.....。。。。")
        } catch (e: Exception) {
            e.printStackTrace()
        }
    }
    time = System.currentTimeMillis()
    loading.dismiss()
    return bt
}
```

5.2.8 图像输出-保存至相册

图像输出-保存至相册，用于将编辑好的证件照保存到用户的相册中。在本软件中，我们通过使用 Android 系统提供的 MediaStore 类，将编辑好的证件照保存到相册中，并在保存成功后弹出提示。用户可以在编辑页面上，点击保存按钮来保存编辑好的证件照。

主要代码如下：

```
private fun saveImageToGallery(context: Context, bitmap: Bitmap, title:
String)  {
    // 首先检查是否有读写权限
    if (ContextCompat.checkSelfPermission(
            context,
            Manifest.permission.WRITE_EXTERNAL_STORAGE
        ) != PackageManager.PERMISSION_GRANTED
    ) {
        // 如果没有权限，则需要请求权限
        getPermissions.launch(
            arrayOf(
                Manifest.permission.WRITE_EXTERNAL_STORAGE,
                Manifest.permission.READ_EXTERNAL_STORAGE
            )
        )
    }
    // 定义保存图片的路径
    val fileName = "${pSize?.name}_$title.jpg"
    BitmapUtils.saveImageToGallery(requireContext(),optBitmap!!,fileName)

}
```

5.2.9 图像出书-分享

图像输出-分享，用于将编辑好的证件照分享给其他人或者分享到社交媒体上。在本软件中，我们通过使用 Android 系统提供的 ShareCompat 类，将编辑好的证件照分享到其他应用中。用户可以在编辑页面上，点击分享按钮来分享编辑好的证件照。

主要代码如下：

```
private fun shareImage(bitmap: Bitmap) {
    val intent = Intent()
    intent.action = Intent.ACTION_SEND
    val uri = Uri.parse(
        MediaStore.Images.Media.insertImage(
            requireActivity().getContentResolver(),
            bitmap,
            "IMG" + Calendar.getInstance().getTime(),
            null
        )
    )
    intent.type = "image/*"
    intent.putExtra(Intent.EXTRA_STREAM, uri)
    requireActivity().startActivity(Intent.createChooser(intent, "title"))
}
```

6 软件测试

6.1 测试概述

在软件开发这一领域，软件实现之后，必须经过严格的测试才能够进行上线、上架，软件测试在整个开发实现过程中，起着不可替代的作用，经过测试后可以对出现的问题进行修改[15]。根据软件或者系统相应的功能测试，使系统能够满足基本功能需求，并避免程序崩溃、或者业务逻辑等出现问题。一定程度的确保了程序的完整性、正确性。

6.2 测试内容

对于本文来说，需要对基于 Android 的证件照 App 的整体功能进行测试。

主要包括功能测试如下：

图像采集-从相机获取、图像采集-从图库获取、图像处理-对比度调整、图像处理-亮度调整、图片处理-饱和度调整、图片处理-底色更换、图片处理-尺寸更换、图片输出-保存至相册、图片输出-分享。

6.3 通过-失败准则

根据测试用例来进行测试，当测试结果且测试过程没有出现任何异常情况，则可认为测试通过，否则测试失败。

6.4 测试计划

6.4.1 测试目标

证件照制作 app 的各个功能能够正常实现，界面显示正常；

6.4.2 测试范围

证件照制作 app 的全部功能；

6.4.3 测试准备

安装应用到测试设备；确保测试设备具有摄像头功能；清空应用数据，确保测试数据干净。

6.4.4 开始和执行测试时的必要动作

打开应用，进入主界面；

点击相机图标进入拍照页面，或者点击文件获取进入图库页面；

点击照片进行编辑、裁剪、保存、分享等操作。

6.4.5 记录测试结果的动作

所有功能的测试均已按照本文 6.5 测试用例中的步骤全部执行，该应用能够正常运行未出现异常情况，则可开始记录测试结果；

如果因为其他应用对系统造成干扰使得测试结果出现问题，则此测试结果不能进行记录。

6.4.6 停止和最终重新启动测试的条件和动作

如果该应用因其他外部因素干扰或应用本身导致在运行时出现了超时、卡顿、闪退的情况则可以重新启动测试。

6.5 测试用例

6.5.1 图片采集模块测试用例

当用户进入图像采集页面之后，可以以两种不同的方式进行图片的获取，我们需要对这两个功能点进行功能测试，确保功能的完整性、正确性，测试用例如表 6.1 所示。

表 6-1 图片采集模块测试用例

项目名称	基于 Android 的证件照制作软件				
开发人员	刘志春	模块名称	图片采集模块	测试日期	2023-3-26
测试类型	系统测试	测试人员	刘志春	测试方法	黑盒测试
编号	测试项	测试类别	描述/输入/操作	期望结果	测试结果
1	图片采集-从图库获取	功能测试	1）从主页面进入图片选择 2）点击菜单栏，选择从图库选取照片 3）从图库选择照片后，人脸检测	如果检测到人脸则通过，将下一步按钮至为激活状态，否则禁止状态	与期望结果相符
2	图片采集-从相机获取	功能测试	1）从主页面进入图片选择 2）默认为相机拍照状态 3）点击相机旋转，切换镜头 4）拍取图片 5）进行人脸检测	如果检测到人脸则通过，将下一步按钮至为激活状态，否则禁止状态	与期望结果相符

6.5.2 图片处理模块测试用例

当用户进入图片编辑页面，可以进行图片的底色更换、对比度、亮度、饱和度、尺寸等参数进行调整，并能够实时预览，我们需要对这些功能进行测试，确保程序的完整性、正确性。测试用例如表 6.2 所示。

表 6-2 图片处理模块测试用例

项目名称	基于 Android 的证件照制作软件				
开发人员	刘志春	模块名称	图片处理模块	测试日期	2023-3-26
测试类型	系统测试	测试人员	刘志春	测试方法	黑盒测试

编号	测试项	测试类别	描述/输入/操作	期望结果	测试结果
1	图片处理-底色更换	功能测试	1）用户选取图片后进入编辑页面 2）用户点击编辑页面底部操作栏换色 3）选择颜色值 4）等待请求换色	证件照进行换色	与期望结果相符
2	图片处理-对比度	功能测试	1）用户选取图片后进入编辑页面 2）用户点击编辑页面底部操作栏对比度 3）用户滑动滑条进行调整	随着用户操作，实时改变图像对比度	与期望结果相符
3	图片处理-饱和度	功能测试	1）用户选取图片后进入编辑页面 2）用户点击编辑页面底部操作栏饱和度 3）用户滑动滑条进行调整	随着用户操作，实时改变图像饱和度	与期望结果相符
4	图片处理-亮度	功能测试	1）用户选取图片后进入编辑页面 2）用户点击编辑页面底部操作栏饱和度 3）用户滑动滑条进行调整	随着用户操作，实时改变图像亮度	与期望结果相符
5	图片处理-尺寸	功能测试	1）用户选取图片后进入编辑页面 2）用户点击编辑页面底部操作栏尺寸 3）选择尺寸值	实时的显示用户选择的尺寸图片	与期望结果相符

6.5.3 图片输出模块测试用例

当用户进入图片编辑页面，图片编辑完后可以进行输出，用户点击菜单栏可以进行保存、分享等操作，我们需要对这些功能进行测试，确保程序的完整性、正确性。测试用例如表 6.3 所示。

表 6-3 图片输出模块测试用例

项目名称	基于 Android 的证件照制作软件				
开发人员	刘志春	模块名称	图片输出模块	测试日期	2023-3-26
测试类型	系统测试	测试人员	刘志春	测试方法	黑盒测试
编号	测试项	测试类别	描述/输入/操作	期望结果	测试结果
1	图像输出-保存至图库	功能测试	1）用户已经在编辑界面编辑好证件图片 2）点击菜单图标 3）点击保存	保存图片在图库	与期望结果相符
2	图片输出-分享	功能测试	1）用户已经在编辑界面编辑好证件图片 2）点击菜单图标 3）点击分享	分享图片成功	与期望结果相符