### 基于QT的小型PS应用需求分析

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 修订历史记录   |  |  |  |  | | --- | --- | --- | --- | | **日期** | **版本** | **说明** | **作者** | | 2021年4月13日 | 1.0 | 创建 | 2018141463026王雨一 | | 2021年4月18日 | 2.0 | 修改，加入图表 | 2018141463026王雨一 | |

目录

[基于QT的小型PS应用需求分析 1](#_Toc69829872)

[1引言 2](#_Toc69829873)

[1.1研究背景 2](#_Toc69829874)

[1.2开发littlePS的意义 3](#_Toc69829875)

[1.3环境，开发工具 3](#_Toc69829877)

[2功能模块 3](#_Toc69829878)

[2.0功能综述 3](#_Toc69829879)

[2.0.1 系统描述（功能实现） 4](#_Toc69829880)

[2.1基本功能 7](#_Toc69829881)

[2.1.1标题栏 7](#_Toc69829882)

[2.1.2菜单栏 8](#_Toc69829883)

[2.1.3绘图区 8](#_Toc69829884)

[2.1.4图层区 9](#_Toc69829885)

[2.1.5 工具栏 9](#_Toc69829886)

[2.2特殊图片处理 10](#_Toc69829887)

[2.2.1转换 10](#_Toc69829888)

[2.2.2压缩 11](#_Toc69829889)

[2.3 基于深度学习的图像处理 11](#_Toc69829890)

[2.3.1超分辨率 11](#_Toc69829891)

[3 非功能性需求分析 12](#_Toc69829892)

[3.1 综述 12](#_Toc69829893)

[3.1.1可靠性 12](#_Toc69829894)

[3.1.2易用性 12](#_Toc69829895)

[3.1.3性能 12](#_Toc69829896)

[4外部接口需求分析 13](#_Toc69829897)

[4.1用户与系统 13](#_Toc69829898)

[4.1.1用户接口 13](#_Toc69829899)

[4.1.2程序接口 13](#_Toc69829900)

# 1引言

## 1.1研究背景

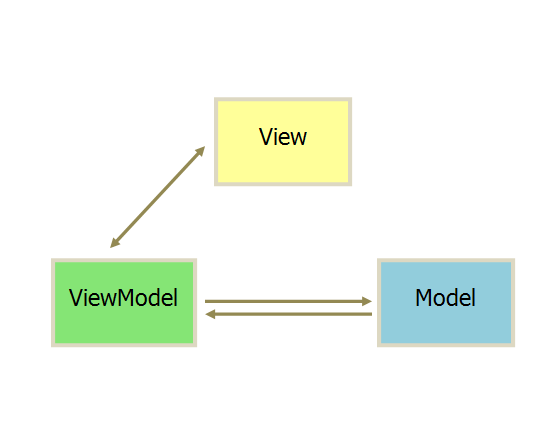
Photoshop软件收费而且功能复杂，不容易上手，而我们有时候只需要做一些简单的日常性的图像处理。

## 1.2开发littlePS的意义

## 相比于PS工作流程的繁琐，littlePS简单易用，同时一样有一定量的一键处理图像功能，如图像增强、颜色模型转换、变长编码图像压缩、综合图像处理特效等，同时小组成员通过做littlePS的过程，获取做实用软件项目的经验。

## 1.3环境，开发工具

该项目的整体架构如下：



图表1 整体架构图

架构说明：

**Model：**保存图层数据，当前画笔等参数

**ViewModel：**将model中的各个图层整合成一张可显示的图片

**ViewUI：**布局显示ViewModel整合完毕的图片

使用工具以及语言：

|  |  |  |  |
| --- | --- | --- | --- |
| 所属类别 | 名称 | 语言 | 工具 |
| 客户端 | 客户端 | C++,Python | qt6.0.3,wvvm框架 |

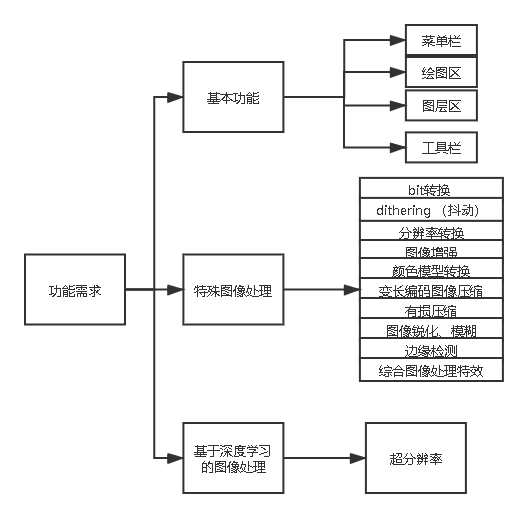
使用语言及工具表

# 2功能模块

## 2.0功能综述

本项目旨在设计一款便于使用的图像编辑和处理软件，要求具备图层管理、简单几何操作、导入导出、图像处理滤镜等功能，满足日常图片处理的需求。

系统界面分为三个模块：基本功能、特殊图像处理、基于深度学习的图像处理



图表2 需求模块总体框架

### 2.0.1 系统描述（功能实现）

功能一：

转换8-bit灰度图像为1-bit二值图像

实现方法：

1.Single Threshold(单阈值法)

2.Multi threshold(多阈值法)

功能二：

dithering （抖动）

亮度分辨率与空间分辨率之间转换

实现：

1.dither

2.ordered dither

功能三：

读取不同格式图像文件（BMP,JPEG）

实现：略

功能四：

图像增强

实现：

1.直方图统计以及显示

2.利用直方图进行均衡化

2.幂次变换

功能五：

颜色模型转换

实现：

1.RGB to HSI and HSI to RGB

2.RGB to CMY and CMY to RGB

3.RGB to YCbCr

功能六：

变长编码图像压缩

实现：

1.香农-凡诺算法

2.霍夫曼编码

3.自适应霍夫曼编码

功能七：

无损图像压缩，通过图像差分编码

实现：

1.无损JPEG

功能八：

有损压缩算法值量化

均匀量化（Uniform quantization）:指把输入信号的取值域等间隔分割的量化。

实现：

1. CR为1，2，3，4的均匀量化

2.CR为2改进的IGS的均匀量化

功能九：

有损压缩之变换编码之离散余弦变换(Discrete Cosine Transform)

实现：

1.DCT变化

2.反DCT变化

功能十：

图像锐化使模糊图像变清晰（采用微分法）

实现：

1.梯度法

2.Sobel算子法

3.Laplace算子法

功能十一：

图像模糊（平滑）

实现：

1.高斯模糊（Gaussian Blur）

2.方框模糊（Box Blur）也即盒式模糊

3. Kawase模糊

功能十二：

图像浮雕：浮雕效果是将图像的变化部分突出的表现出来，而相同的颜色部分则被淡化掉，使图像出现纵深感，从而达到浮雕的效果。

实现：

1.基本可选方向浮雕：像素与周围八个像素差值确定方向  
2.调和浮雕：像素与左上，左下，右上，右下四个方向差值确定新像素。

功能十三：

边缘检测

实现：

1.sobel算子

2.prewitt 算子

3.roberts算子

4.canny算子

功能十四：

综合图像处理特效：

实现：

1.素描特效：

* 彩色图像灰度化处理、
* 高斯滤波降噪、
* 边缘检测采用Canny算子、
* 反二进制阈值化处理实现素描特效

2.怀旧特效：

图像的RGB三个分量分别按照一定比例进行处理

3.光照特效：

指图像存在一个类似于灯光的光晕特效，图像像素值围绕光照中心点呈圆形范围内的增强。

4.流年特效：

将原始图像的蓝色（B）通道的像素值开根号，再乘以一个权重参数

5.积木效果：

对图像中的各个像素点着重(即加大分像素的颜色值)着色

6.油画效果：采用毛玻璃算法：用当前点四周一定范围内任意一点的颜色来替代当前点颜色，最常用的是随机的采用相邻点进行替代。

7.图像滤镜特效：基于颜色查找表（Look up Table）的滤镜处理方法，通过将每一个原始颜色进行转换之后得到新的颜色。

功能十五:

超分辨率

实现：

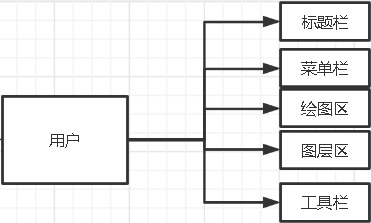
SRCNN方法的网络结构仅仅用了三个卷积层SRCNN首先使用双三次(bicubic)插值将低分辨率图像放大成目标尺寸，接着通过三层卷积网络拟合非线性映射，最后输出高分辨率图像结果。本文中，作者将三层卷积的结构解释成三个步骤：图像块的提取和特征表示，特征非线性映射和最终的重建。

三个卷积层使用的卷积核的大小分为为9x9,，1x1和5x5，前两个的输出特征个数分别为64和32。用Timofte数据集（包含91幅图像）和ImageNet大数据集进行训练。使用均方误差(Mean Squared Error, MSE)作为损失函数，有利于获得较高的PSNR。

## 2.1基本功能

基本功能是整个系统的主体功能，包括

1. 标题栏②菜单栏③工具栏④绘图区域⑤图层区



图表3 用户用例

**使用者：**所有用户

### 2.1.1标题栏

显示littlePS字样加版本号

**用户故事：**

作为一个<用户>,

我想要<看标题>,

以便于<理解软件名称和版本>

**基本事件流：**

1、用户打开软件，本用例开始；

2、用户看到软件上方的标题栏**;**

### 2.1.2菜单栏

文件：打开，新建，保存，另存为

绘制：画线，椭圆，矩形等几何图形

图层：平移、旋转等结合变换，移动图层、删除图层等图层操作

滤镜：反色、直方图均衡、亮度适应、卷积变换、拉普拉斯增强、灰阶化、二值化、亮度调整、双边滤波

特殊处理：对图像进行特殊处理，详见2.2

帮助：关于

**用户故事：**

作为一个<用户>,

我想要<给图片加反色滤镜>,

以便于<对图片进行修剪>

**基本事件流：**

1、用户点击文件按钮进入文件栏，本用例开始；

2、用户选择“打开”点击，点击后弹出文件系统**;**

3、用户在文件系统中选择所需图片，双击打开图片；

4、图层区选中图层后，点击菜单栏滤镜，然后点击下拉栏中的“反色”；

5、图片成功加上反色滤镜。

### 2.1.3绘图区

绘图工作区域，所有操作在绘图区域执行和展现。

**用户故事：**

作为一个<用户>,

我想要<放置图片到绘图区>,

以便于<编辑图片>

**基本事件流：**

1、用户点击文件按钮进入文件栏，本用例开始；

2、用户选择“打开”点击，点击后弹出文件系统;

3、用户在文件系统中选择所需图片，双击打开图片；

4、绘图区中展现打开的图片。

### 2.1.4图层区

展示图层的上下层叠关系。

从上到下按层级关系显示图层，点击图层可以选中该图层的图片，图层区上方有“上移图层”、“下移图层”和“删除图层”按钮。

**用户故事：**

作为一个<用户>,

我想要<置某图片至底层>,

以便于<层次化编辑图片组合>

**基本事件流：**

1、导入两张图片后，点击右方图层区；

2、选中需要置于底层的图层，点击鼠标右键，弹出菜单；

3、选择“下移图层”，该图片被置于底层。

### 2.1.5 工具栏

直接点击按钮可以使用相应功能的工具栏，相当于菜单栏的图形化快捷方式。

添加图片：十字架图案

直线工具：直线状

矩形工具：矩形状

椭圆工具：椭圆状

图片平移工具：带箭头十字架状

旋转工具：两个首尾相接的环状箭头

删除图片：×状

**用户故事：**

作为一个<用户>,

我想要<使用椭圆工具>,

以便于<画一个椭圆>

**基本事件流：**

1、用户点击工具栏的椭圆形状按钮；

2、光标变成十字架标识后，用户在绘图区点击并拖动；

3、绘图区随着光标出现一个椭圆。

## 2.2特殊图片处理

**使用者：**所有用户

此模块为区别于基本功能的特殊功能，旨在用算法对图像进行特殊处理，具体功能包括 ：

转换：bit转换、抖动（dithering）、分辨率转换；

压缩：变成编码图像压缩、有损压缩；

图像增强；

图像锐化、模糊、浮雕；

边缘检测；

综合图像处理特效（素描、怀旧、流年、光照、积木、油画、滤镜）。

### 2.2.1转换

转换8-bit灰度图像为1-bit二值图像:

>>>>>>>>> 

dithering （抖动）:在低分辨率下看较高分辨率的图像、低色模式下显示更多色模式

颜色模型转换:实现RGB和HIS,CMY,YcbCr几种颜色模型之间的转换

**用户故事：**

作为一个<用户>,

我想要<使用bit转换功能>,

以便于<将图片简化/模糊化>

**基本事件流：**

1. 用户点击想要转换的图片所在的图层；
2. 点击菜单栏的“特殊”栏；
3. 选择转换，然后选择8bit to 1bit 转换。
4. 图片完成bit转换

### 2.2.2压缩

随着多媒体的广泛应用,图像的存储、处理、传送成了计算机应用的一个非常重要的问题,由于受计算机硬件的限制,图像的压缩成为多媒体应用的十分重要的问题之一。“压缩功能”主要包括：变长编码图像压缩和无损图像压缩

**用户故事：**

作为一个<用户>,

我想要<无损压缩图片>,

以便于<低占用率保存图片>

**基本事件流：**

1. 点击菜单栏中“特殊”栏，然后点击“压缩”；
2. 选择“无损压缩”，弹出文件系统窗口；
3. 选择导出图片到某磁盘某文件夹，编辑图片名称，点击确定；
4. 弹出“压缩中”窗口，压缩完成后弹出“完成”窗口；
5. 点击确定完成无损压缩

## 2.3 基于深度学习的图像处理

超分辨率技术（Super-Resolution, SR）是指从观测到的低分辨率图像重建出相应的高分辨率图像。本功能使用超分辨率技术，帮助用户用时间带宽(获取同一场景的多帧图像序列)换取空间分辨率,实现时间分辨率向空间分辨率的转换。

**使用者：**所有用户

### 2.3.1超分辨率

**用户故事：**

作为一个<用户>,

我想要<使用超分辨率功能>,

以便于<提高原有图像的分辨率>

**基本事件流：**

1. 选择要提高分辨率图像所在的图层；
2. 在菜单栏选择“提高分辨率”栏；
3. 弹出“超分辨率”窗口，显示“提高分辨率中”；
4. 显示“提高分辨率成功！”，完成本用例。

# 3 非功能性需求分析

## 3.1 综述

在应用开发过程中，非功能性需求是需求的一个重要组成部分，它同时也会影响了系统的架构设计,本系统的非功能性需求包括**可靠性**、**易用性、性能**

### 3.1.1可靠性

对输入以及写入有一个严格的检查过程来防止数据异常。可以处理系统运行过程中出现的各种异常情况，如:人为错误，输入非法数据，硬件设备故障等，系统应能正确处理并妥善避免。

### 3.1.2易用性

交互简单友好、所呈现的界面逻辑清晰而不会混淆。

### 3.1.3性能

在测试用机

（Windows10家庭中文版

处理器:Intel(R)Core(TM) i5-8300H CPU@ 2.30GHz 2.30 GHz

内存(RAM):8.00 GB (7.88 GB可用)

系统类型:64位操作系统，基于x64的处理器）

的环境下：

（1）软件一般处理（编辑/删除图层、绘制图像、平移图像）响应时间不超过1秒

（2）特殊图像处理（转换、压缩等）响应时间不超过10秒

（3）图片导入、导出响应时间不超过1秒

# 4外部接口需求分析

## 4.1用户与系统

### 4.1.1用户接口

方便用户直接或间接控制作业。使用IO设备进行命令解释和键盘命令的输入输出，如编辑用户信息，管理订单等。

### 4.1.2程序接口

留给开发人员用来实现功能的接口

一、view模块

* **MainWindow类**
* MainWindow类是程序的主窗口，也是View模块与其他部分对接的类。有一些设置命令的函数，并且通过指针绑定了ViewModel层的displayImage用于显示。另外还有一些槽函数用于与其他控件通信。通过通知的方式接受ViewModel的消息以同步数据。在按键被按下时调用StateManager来处理事件并改变状态。通过ViewModel层绑定了Model中的pen和brush变量，并且将变量传递给了MainDisplayWidget控件。
* **ImageWidget类**
* ImageWidget类是主窗口中用于显示的显示区域，负责将ViewModel计算好的displayImage显示在窗口上
* **ConvolutionDialog类**
* ConvolutionDialog类是用于设置卷积核的窗口，该类有一个静态成员函数static Params GetConvolutionCore(QWidget\*,int)，用于设置卷积核。调用该函数后，会弹出一个阻塞窗口，当该窗口被关闭时函数返回，并且用Params类返回结果。
* **NewCanvasDialog类**
* NewCanvasDialog类是用于设置画布尺寸的窗口，该类有一个静态成员static Params GetCanvasSize(QWidget \*);。调用该函数后，会弹出一个阻塞式窗口，当该窗口被关闭时函数返回，并且用Params类返回结果。
* View包括4个窗体类(MainWindow , ImageWidget , ConvolutionDialog , NewCanvasDialog)、1个状态基类(BaseState)、13个状态类、1个状态通用操作类(StateCommonAction)与1个状态管理类(StateManager)。

二、Model模块

* 添加直线
* void addLine(double, double, double, double, double, double)
* 添加椭圆
* void addEllipse(double, double, double, double)
* 添加矩形
* void addRect(double, double, double, double)
* 添加图片
* void addImage(string)
* 添加文本
* void addText(int, int, string)
* 图层格式转换
* void LayoutTransform(int, int)
* 图层顺序调整
* void LayoutOrderChange(int, int, int = 0)
* 删除图层
* void DeleteLayout(int)
* 清除操作记录
* void clearDoneEvent()
* 添加操作记录
* void addDoneEvent(int, int, shared\_ptr<BaseShape>, shared\_ptr<BaseShape>, int)
* 添加形状基类
* void addBaseShape(vector<shared\_ptr<BaseShape>>, shared\_ptr<BaseShape>)
* 重做
* void redo()
* 撤销
* void undo()
* 新建工程
* void newProject(int, int)
* 保存工程
* void saveProject(string)
* 打开工程
* void loadProject(string)
* 获取画布宽
* int getCanvasWidth()
* 获取画布高
* int getCanvasHeight()
* 设置笔刷
* void SetPen(Pen)
* 设置笔刷颜色
* void SetPenColor(unsigned char, unsigned char, unsigned char)
* 设置笔刷大小
* void SetPenWidth(int)
* 设置笔刷模式
* void SetPenStyle(int)
* 设置填充色
* void SetBrushColor(unsigned char, unsigned char, unsigned char)
* 设置填充模式
* void SetBrushStyle(int)
* 工程为空
* bool isProjectEmpty()
* 获取填充笔刷
* const Brush\* GetBrush()
* 获取笔刷
* const Pen\* GetPen()
* 获取图层表
* const Layouts\* GetLayouts()

三、ViewModel模块

* void ViewModel::SaveAsPicture(string path)
* 保存图片图层
* void RefreshDisplayImage(int index=-1)
* 重新渲染图像，保存在viewmodel内存空间中
* void NewCanvas(unsigned int width,unsigned int height)
* 新建画布
* void SetSelectedLayout(int selectedLayout)
* 设置选择的图层
* void SetPen(const Pen\* pen)
* 设置画笔
* void SetBrush(const Brush\* brush)
* 设置笔刷
* void LayoutMove(int x,int y)
* 移动图层
* void LayoutRotate(double angle);
* 旋转图层
* void LayoutScale(double scaleX,double scaleY);
* 缩放图层
* void ClearViewModel()
* 清空ViewModel画布