需求报告

2020年暑假短学期C++项目管理和工程实践

小组名称：图片浏览器组

小组成员：朱紫涵、聂俊哲、张琦

小组选题：图片浏览器组

任务要求：

1. 多人协同工作，完成MVVM工程
2. 多次迭代，功能逐步加入

项目描述

项目具体内容介绍

本项目最初是想复现siggraph96的一篇论文 **View Morphing** 。用户可以输入两张不同角度拍的图片，标注上特征点后，实现人脸morph。但是考虑到本次课程名为项目管理和工程实践，为了加深我们对多人合作完成项目这一过程的理解，我们重新制定了项目目标。

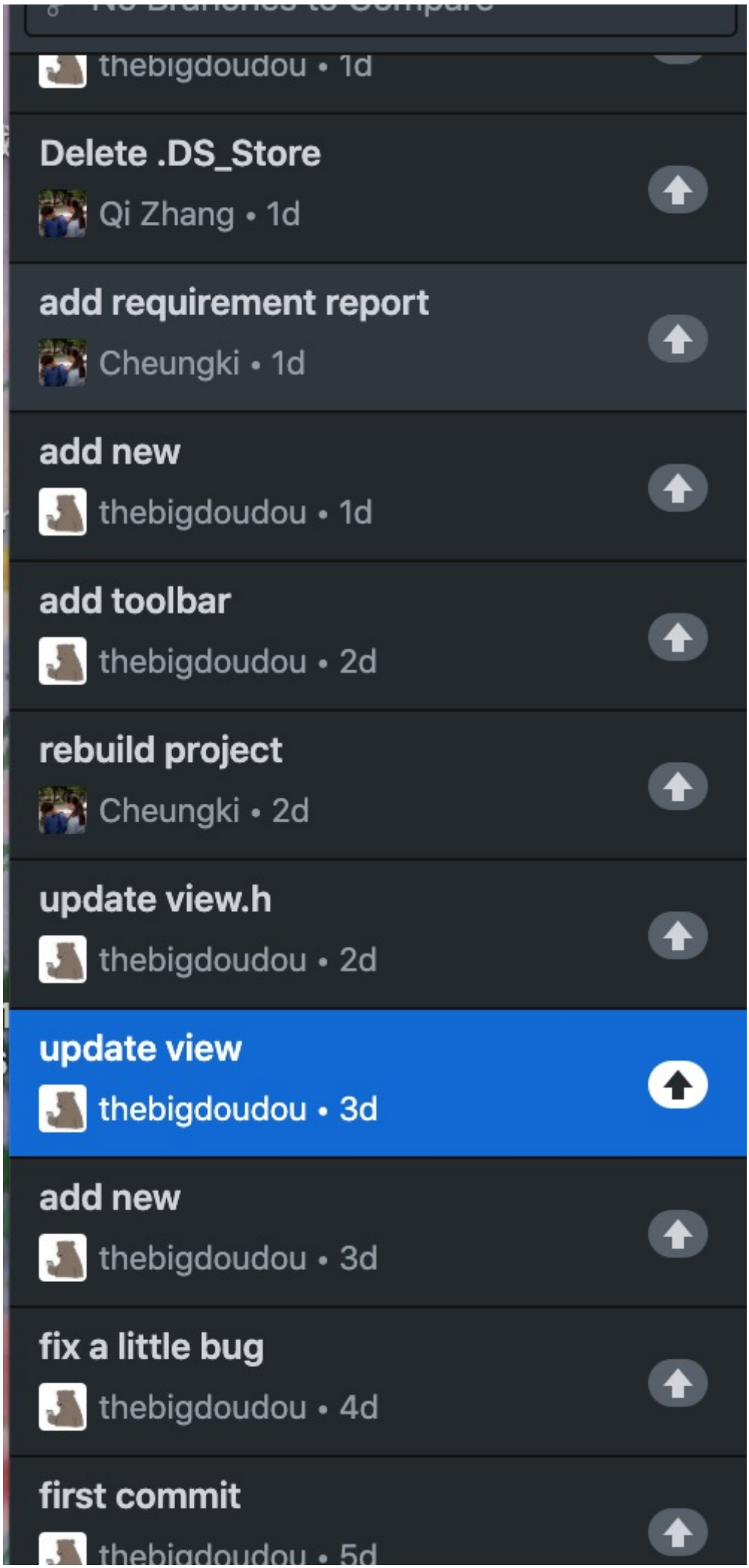
本项目将设计一个图像浏览器及处理器，具备最基本的文件夹内图像查看、放大缩小、顺时针逆时针旋转、水平或垂直翻转、保存、删除等功能，还可以快速的查看图片信息，幻灯片形式播放图片。如果时间充裕，View Morphing将变成它的子功能。软件用户界面良好、操作简单、功能明确。

拟实现功能

- [x] 图片导入
- [x] 图片所在的同个文件夹下，能够前一张后一张的查看图片
- [x] 更改后保存图片
- [x] 放大、缩小功能，显示缩放比例
- [x] 旋转功能，将图片向左旋转或向右旋转
- [x] 镜像功能，将图片上下翻转或左右翻转
- [x] 以幻灯片形式播放图片功能
- [x] 查看图片大小、创建日期等信息
- [ ] view-morphing
- [ ] 调用摄像头拍照
- [ ] ……

项目管理

- 版本控制：Git，Github
- 持续集成： Jenkins
- 开发环境：QT 5.12.1
- 编程语言： C++ 17
- 第三方库： opencv
- 框架结构：MVVM（前后端分离，复用高，低耦合）
- 文档撰写：markdown，latex
- 交流讨论：钉钉视频会议、QQ


thebigdoudou • 1d
Delete .DS_Store
Qi Zhang • 1d
add requirement report
Cheungki • 1d
add new
thebigdoudou • 1d
add toolbar
thebigdoudou • 2d
rebuild project
Cheungki • 2d
update view.h
thebigdoudou • 2d
update view
thebigdoudou • 3d
add new
thebigdoudou • 3d
fix a little bug
thebigdoudou • 4d
first commit
thebigdoudou • 5d

prev next pic supported
Zzh2000 • 6h

fix flip bug
thebigdoudou • 8h

change photomap to ptr
thebigdoudou • 9h

add flip (still bugs)
Zzh2000 • 9h

The second iteration
thebigdoudou • 11h

rotation bug fixed
Zzh2000 • 19h

change to multi images
Zzh2000 • 20h

now supports macOS and adds ...
Zzh2000 • 23h

Create README.md
Qi Zhang • 1d

update view
thebigdoudou • 1d

Delete .DS_Store

系统需求

软件目前在Windows和mac系统下都能运行。

分工

聂俊哲：APP层，统筹协作，各层都有参与

朱紫涵：model和viewmodel层，负责逻辑的实现

张琦：view层，负责界面的显示

- 第一轮迭代
  - 做出主窗口
  - 实现图片的导入
  - 实现软件退出与关于信息弹出功能
- 第二轮迭代
  - 基本的放大缩小功能
  - 全屏查看功能
  - 信息查看功能
  - 图片保存功能
  - 图片删除功能
- 第三轮迭代
  - 图片顺时针、逆时针翻转功能
  - 图片上下、左右镜像功能
- 第四轮迭代
  - 图片播放功能
- 第五轮迭代
  - 页面美化

MVVM架构

common层

- 无，没有各层共用的数据结构，只包含了两个头文件

model层

- 实现各类方法，包括图片翻转与镜像。

view层

- 对`view`层进行操作，会触发对应的槽函数。本身提供一个用于更新的`notification`，并提供`get()`方法交给`ViewModel`层进行绑定，如此可以实现`ViewModel`通知`View`进行更新。
- 同时，本身提供很多`Command`的成员变量，这些变量本身属于`ViewModel`层，并在`ViewModel`层提供`get`方法给`View`层进行绑定，这样就实现了`View`发送`command`给

`ViewModel`层，`View`就可以在不知道`Command`具体派生类的情况下写代码。

viewmodel层

- `viewmodel`层利用`command`解析出的参数，调用`model`里对应的方法。

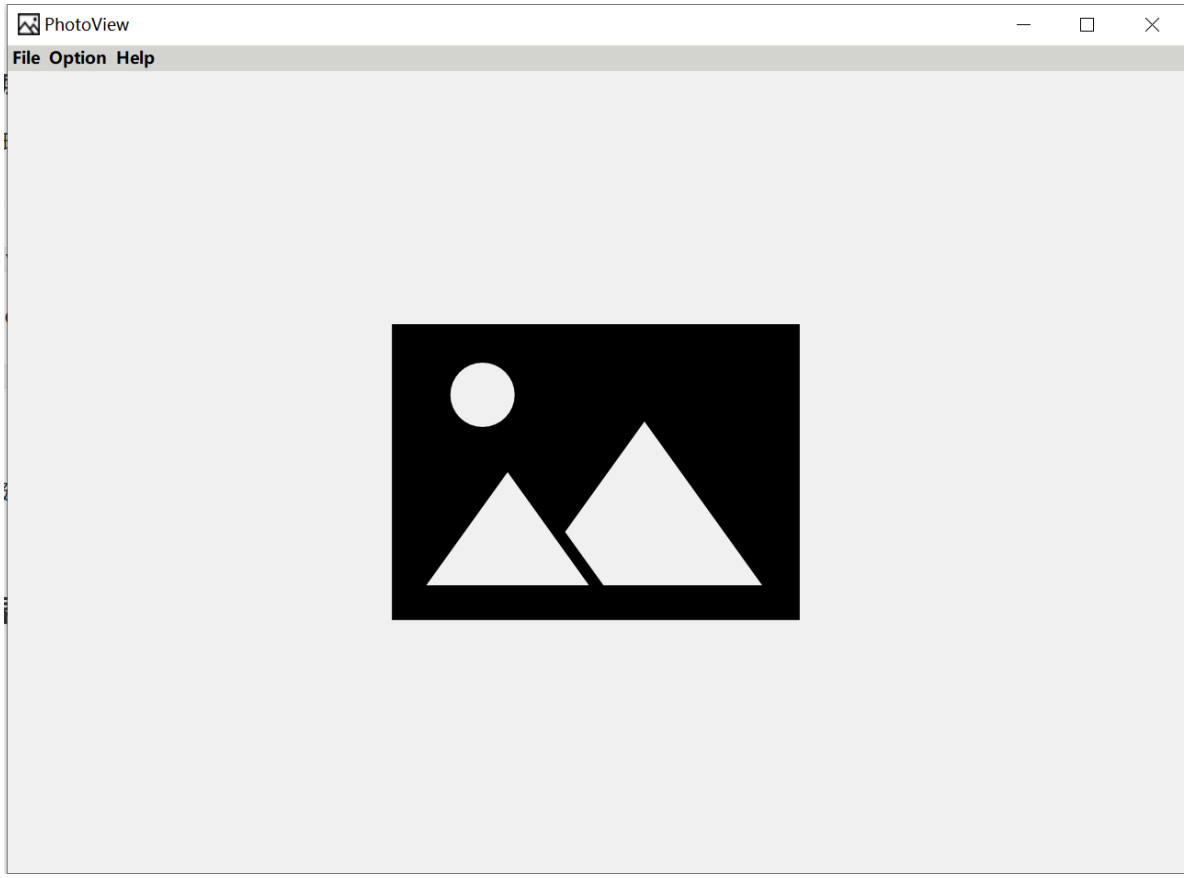
app层

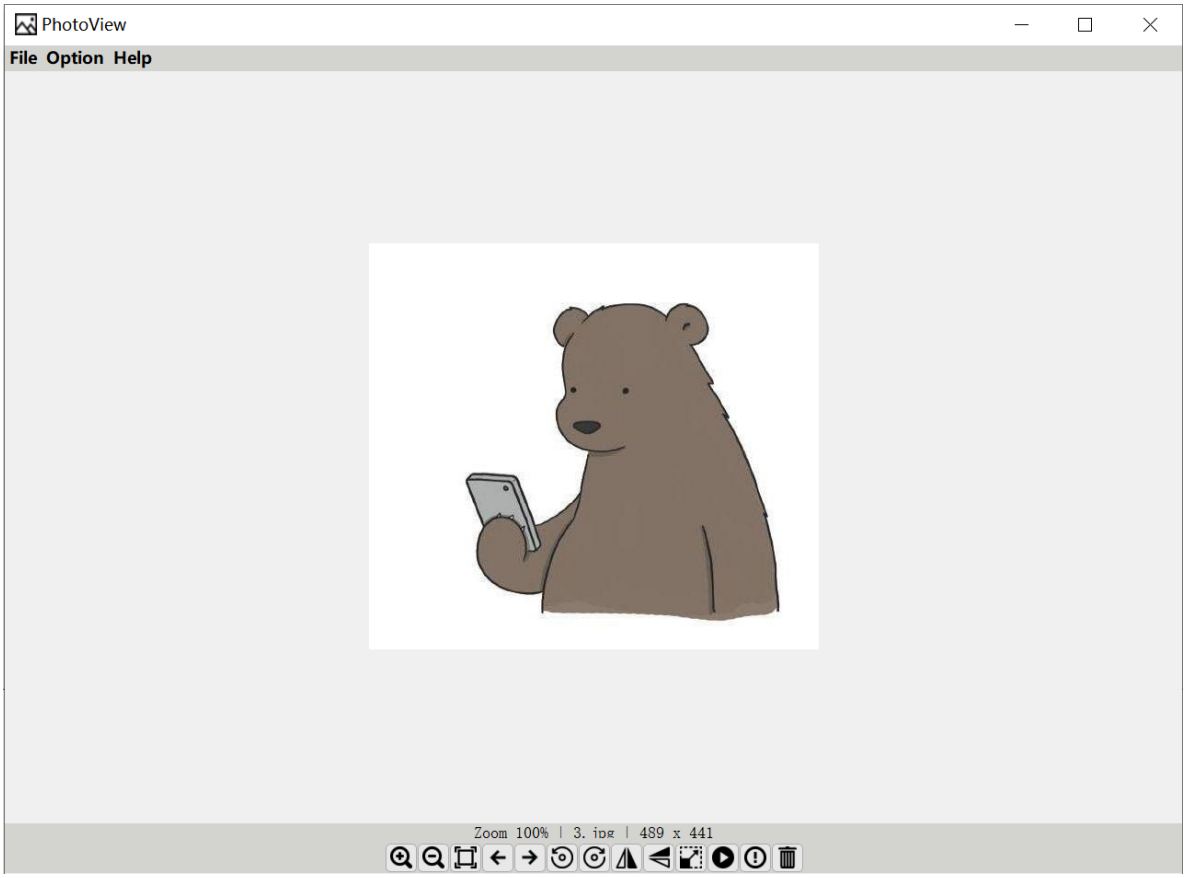
- 创建一个包含`view`、`model`和`viewmodel`三个对象的类，并调用三个对象的函数完成初始化、绑定操作。

后续改进思路

- 因为lambda表达式我们用的还不是很熟，所以刚开始的开发过程中未使用。后续会进一步优化代码，减少command类的继承，取消exec_这种执行方式。

初步界面




PhotoView
File Option Help
Zoom 100% | 3.jpg | 489 x 441