浙江大學

魔方模拟器个人分报告

授课教师：	袁昕
姓　　名：	戴卿
组　　别：	魔方模拟组
日　　期：	2023-7-10

1 分工任务

在第一轮迭代中，我负责了ViewModel层和Model层的实现任务。我主要确保视图模型层与数据模型层之间的协调和顺畅交互，以及实现与魔方相关的业务逻辑和数据处理功能。通过编写适当的代码，我实现了视图模型与数据模型之间的数据绑定，以及处理用户输入和魔方状态更新的功能。

在第二轮迭代中，我负责了Common层和App层的开发工作。我的任务是设计和实现通用的功能模块，以及开发应用层的逻辑和交互功能。他通过编写代码，实现了随机打乱魔方、文件存储、输入加载和魔方重置等功能的基础部件。

在第三轮迭代中，我承担View层的开发工作以及文档撰写。我优化了用户界面，确保用户能够直观地操作魔方模拟器，并且优化了渲染方式。我还编写了清晰详细的文档，方便团队成员之间的沟通和交流。

2 设计思路

2.1 第一次迭代

第一次迭代的任务是ViewModel层和Model层，主要确保视图模型层与数据模型层之间的协调和顺畅交互，以及实现与魔方相关的业务逻辑和数据处理功能。

2.1.1 魔方模型实现

在这部分中，实现了Cube类作为整个魔方的数据底层。为了方便操作和逻辑处理，将魔方分为三层，并实现了相应的数据结构和算法。这样的分层结构为之后的魔方还原算法的实现奠定了基础。在模型中，实现了魔方旋转的矩阵函数，通过调用这些函数可以进行魔方的旋转操作，并将最新的魔方状态传递给View层进行渲染。

2.1.2 命令实现

在ViewModel层中，我实现了各种命令的具体操作，并继承了Common中的命令虚类进行细化。这些命令包括魔方旋转、透明度改变等操作，通过执行相应的命令，可以更新魔方的状态并在界面上呈现出相应的效果。我将这些命令的具体实现代码组织在commands文件夹中，以便于管理和维护。

2.2 第二次迭代

第二次迭代的任务是Common层和App层，主要是优化common层和添加config文件内容。

2.2.1 优化Common层实现

在这次迭代中，对Common层进行了优化。重新构建了Notification基类，使其更加灵活和易于使用。通过优化，提升了通知机制的效率和可扩展性。

2.2.2 Config添加

添加魔方控制器的通用参数，使之更加方便快捷地更改显示效果，如窗口宽度、应用程序背景色、初始视角、旋转速度系数、序列化格式标准和魔方颜色材质等。

2.3 第三次迭代

第三次迭代的任务是View层，主要是优化渲染逻辑。

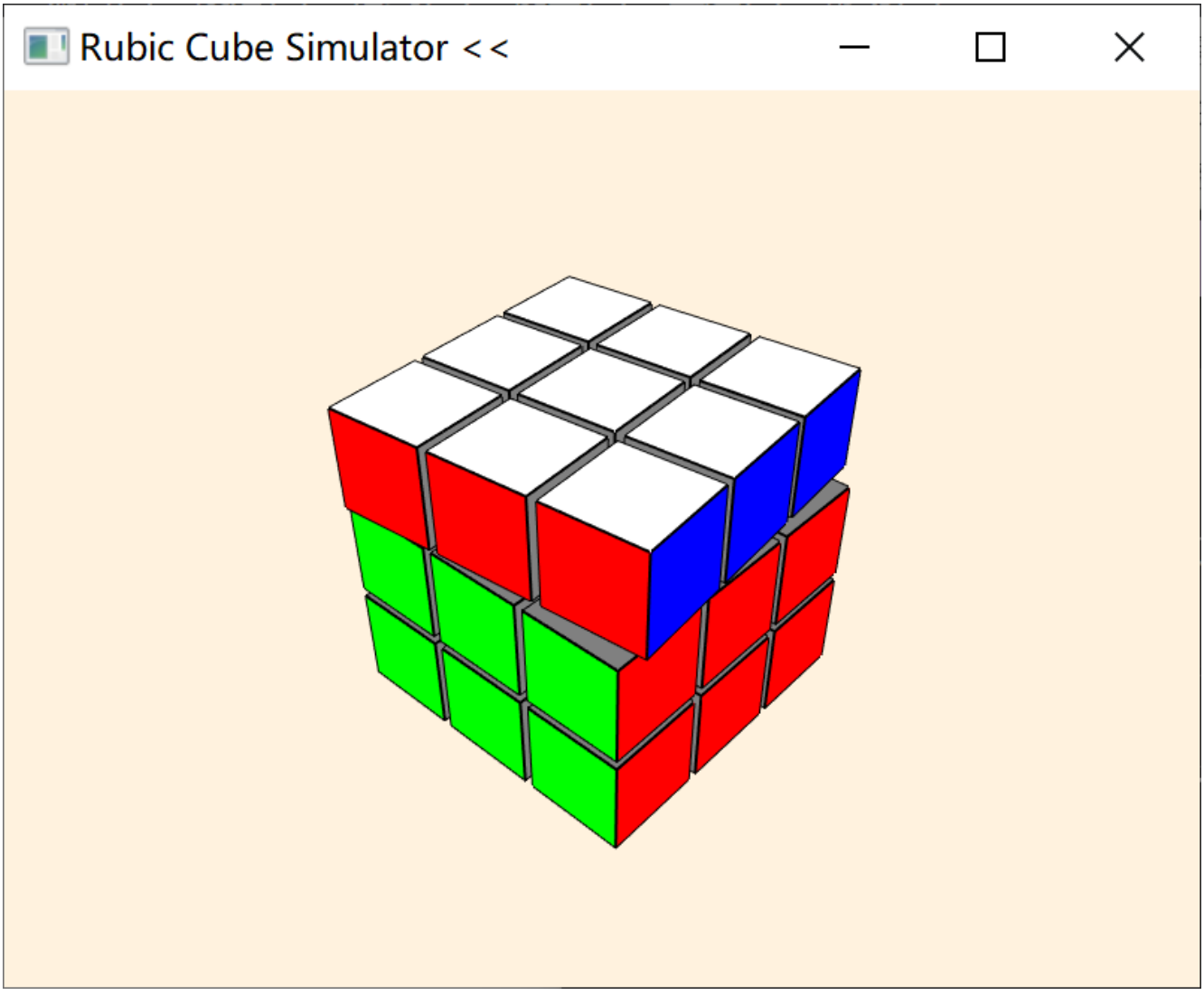
2.3.1 优化渲染逻辑

在View层中，之前的实现方式是一个主循环不断遍历，若有不同的结构变化，则在下一帧进行渲染。然而，这种方式会消耗大量的性能资源。在这次迭代中，对渲染逻辑进行了优化。改为事件触发的方式，即当有指令进入时，通过App层通知View层，View层接收到指令后进行渲染。这样可以避免不必要的渲染，提高渲染效率。

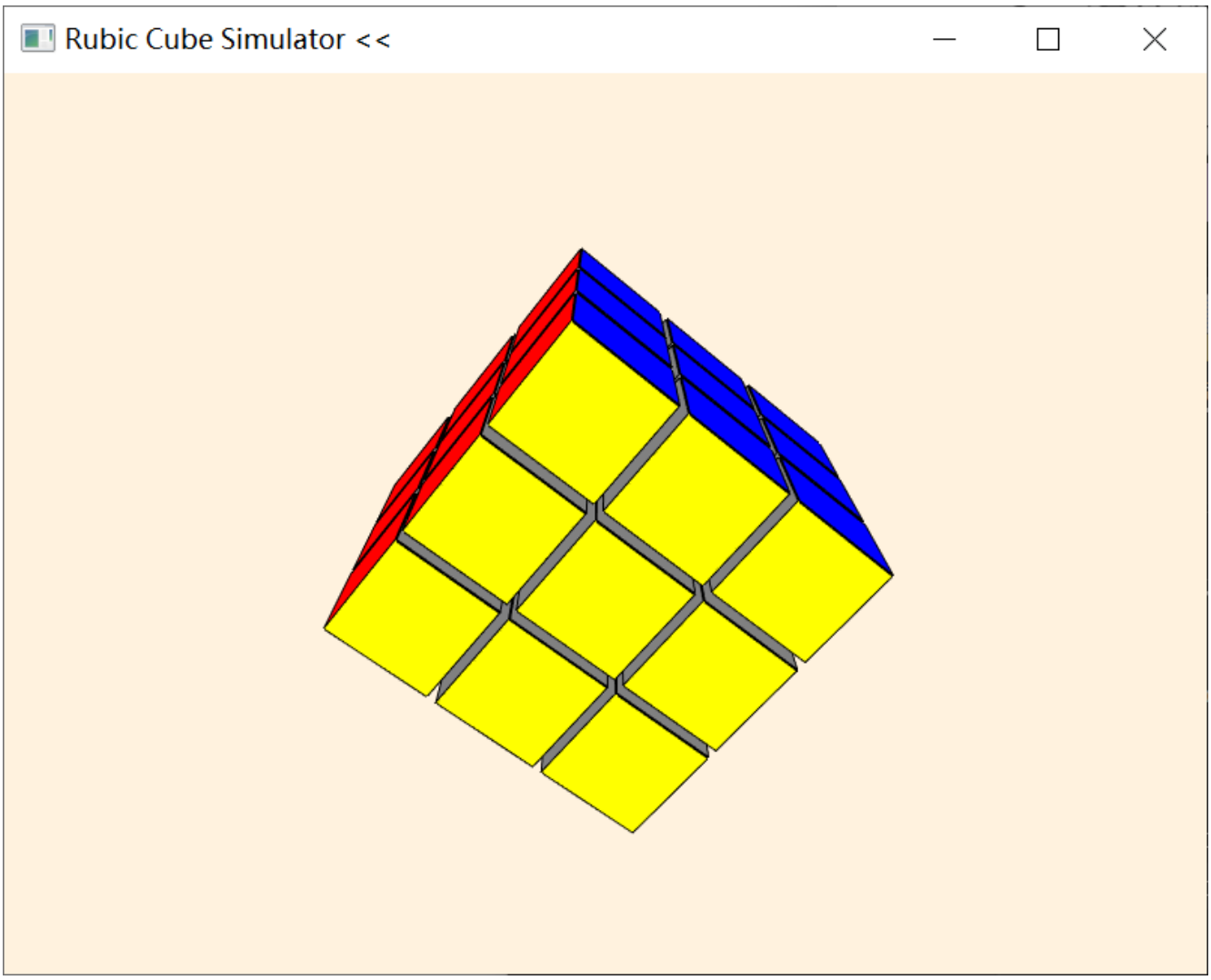
对于旋转和解魔方的指令，当处于渲染状态时，按照之前的方式进行连续绘制，以保证流畅的画面效果。在没有这些指令时，只需渲染单个帧即可，减少资源消耗。通过这样的优化，提升了渲染效率和用户体验。

3 运行效果图

旋转控制：



视角移动：



4　心得体会

之前我对项目管理和软件架构模式有一定的了解，但在实际的团队开发过程中还存在一些不足。通过参与这个课程的学习和项目实践，我对团队协作和软件架构有了更深入的认识和体验。

在第一轮迭代中，我负责了ViewModel层和Model层的任务。通过这个阶段的工作，我学会了如何合理地管理数据和处理业务逻辑，以及如何与其他层进行良好的交互。在第二轮迭代中，我转变了角色，负责了Common层和App层的开发工作。通过这个过程，我进一步提高了对C++编程语言的理解和应用能力。

参与这个项目让我更深入地了解了MVVM架构模式，并在实践中加以应用。通过模块化的开发和清晰的分工，我们能够更加高效地开发和测试各自的模块，大大提高了工作效率。同时，我也学到了如何撰写清晰而详细的文档，以便更好地与团队成员进行沟通和交流。