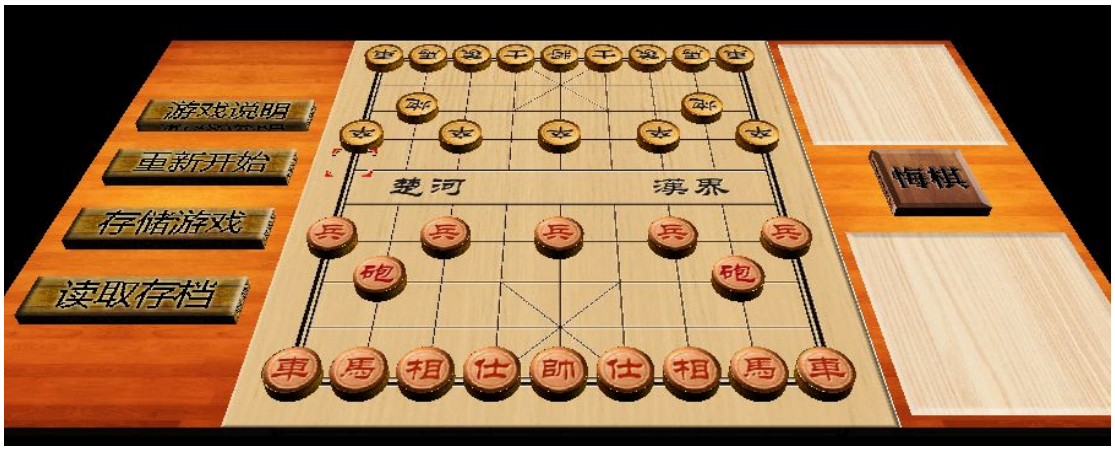
3dChineseChess 说明文档

——程序设计思想与方法 Final Project

5130309751 彭乾旸
5130309784 朱鸿儒



一 程序开发运行环境

1. 操作系统：Windows 8.1

2. Python 版本：2.7.8

3. 依赖库：

   i. pygame-1.9.1

   ii. PyOpenGL-3.1.0

   iii. numpy-1.6.2

   iv. OpenGLLibrary

二 设计思路

2.1 算法

界面部分：

本游戏采取了分层的界面，如图所示：

所有界面都以 x=0 ， z=0 为轴心，而 y 坐标不同

在主界面中，界面部分承担着两个主要任务：

1、将输入量（主要为鼠标输入量）转化为具体指令，并将指令传给后台；

2、接收后台的返回信息，并将返回信息在界面上反映出来。

通俗一点来说，在整个象棋流程中，界面唯一负责的部分就是画图，即将输入量交给后台，并将后台的返回值反映到屏幕上即可。至于点击的位置上有没有棋子，行走的方式是否合法，这些信息在界面部分都是未知的，都只能在后台的返回值之中得到。

在整个游戏中，我们能看到的变化分为两种：

1、摄像机视角的变化；

2、物体位置的变化。

摄像机视角变化是独立于象棋流程之外的，它的写法比较简单，OpenGLLibrary 中为我们封装了几个函数，set_target_pos()、set_target_center()、set_target_up_vector()、set_change_speed()等，对于只有一个窗口的本游戏，摄像机状态可以看做是一个状态机，程序在接收到特定的指令后，即会将摄像机切换到特定的状态，例如：

1、方向键：观察角度改变；

2、点击游戏说明键：摄像头 y 值小幅减小，并对准游戏说明所在的层；

3、点击存档/读档键：摄像头 y 值大幅减小，并对准存取存档所在的层。

另一种变化即为物体位置的变化。本游戏物体位置只存在瞬间变化，即没有速度的概念，位置的改变都在一瞬间内完成。游戏内物体位置的变化分为三种：

1、象棋的位置变化

i. 提起
ii. 移动
iii. 落下
iv. 被吃掉，消失
v. 还原

2、按键的位置变化

i. 按下
ii. 弹起

3、胜利信息的位置变化

i. 比赛结束，一方胜利的文字升起
ii. 重新开始，一方胜利的文字消失

要实现这一点，本游戏采用了一种通用的方法，即是用一个全局变量列表保存所有物体所在的空间位置（列表名：Objects_pos），每一帧绘图时，所有物体的位置都从这个全局列表中读取，即需要改变任何物体的位置时，只需要修改列表中对应于该物体的坐标，即可在下一帧绘图的时候使对应物体的位置完成改变。

算法部分：

本游戏算法部分分为基本构成和功能添加两部分。

1、 基本构成：

1） 棋盘：将每一个可走点都封装成 Unit 类，并带有国界和大本营标记，以方便相，士，兵，帅等特殊棋子约束限制或不同区域的不同走法。同时还有标记棋盘上该点是否有子的状态量。用 global 的 list 存储，在每次开始或读档时重置更新。

Unit 类成员变量：

self.__px,self.__py：位置坐标
self.__occupied：队伍和空位标记
self.__base：“中央九宫格”标记
self.__land：“国界”标记

2） 棋子：全部封装成基类 Chessman，之后再根据不同种类做出不同的派生类，其中定义主要的 move 函数，包括各个棋子的独特走法，传入 unit 类变量，更加方便的进行判断，对每个类实例给予独特编号，作为接口和界面交互。以 global 的 dic 字典存储当前棋盘上所有棋子的类实例，开始新游戏或读档时以此刷新界面。

基类 Chessman 成员变量：

self.__px,self.__py：棋子位置

self.__team：棋子队伍

self.__num：棋子专有序号

派生类新增成员变量：

self.__type：棋子种类

派生类新增方法：

move()：根据各个棋子种类，将其独有走法封装在派生类中，例如炮的走法判断，兵在过界之后的走法变化等等。

3） 计算走法：根据传入的一对坐标值，返回对应的移动或选中是否成功。这是界面和算法部分交互的关键，因为界面传入的实际是一系列点坐标，需要判断前后相邻坐标值是否能构成一步走法，或是在不断地更新选中待走的棋子。在这个过程中，尝试调用选中棋子的 move()函数，如果返回值 True 则成功移动。

4） 移动函数：在确定两对点坐标后进行对应棋子移动和消灭的函数，其中涉及到之后的日志记录和字典刷新等等，但本质上还是在调用特定类的 move()函数。

5） 更新棋盘：在有棋子被吃掉或有棋子位置变换后及时覆盖之前的 dic 记录棋盘的状态，由于 python 类没有指针，所以每次采用复制操作，如果在 C++中可以直接改变指针，但在运行效率上并无明显影响。

2、 功能添加

1） 悔棋功能：在每一次成功的移动时都会修改游戏日志，添加操作步骤，由于游戏简单，所以只有两种日志条目：

第一种：记录一个棋子从原位到另一空位；

第二种：记录一棋子从原位到另一棋子，而被吃棋子进入 trash can。

在悔棋时，考虑到要回到上一次玩家的 turn，因此设定一次悔棋，每个人各悔一子。而且还可添加悔棋次数限制，在每一局中规定每位玩家最多的悔棋次数。

为了完成和界面的交互，每次将悔棋结果以 list 形式传出，list 中包括待悔棋的棋子编号(由实例的 getNum()获得)和目标坐标。

2） 存档功能：提供复盘功能，可以存储游戏至本地，也可从本地文件中读档并继续之前的游戏，之前游戏的所有状态都将被保留还原，包括游戏日志，悔棋步数，红方或黑方走子等。

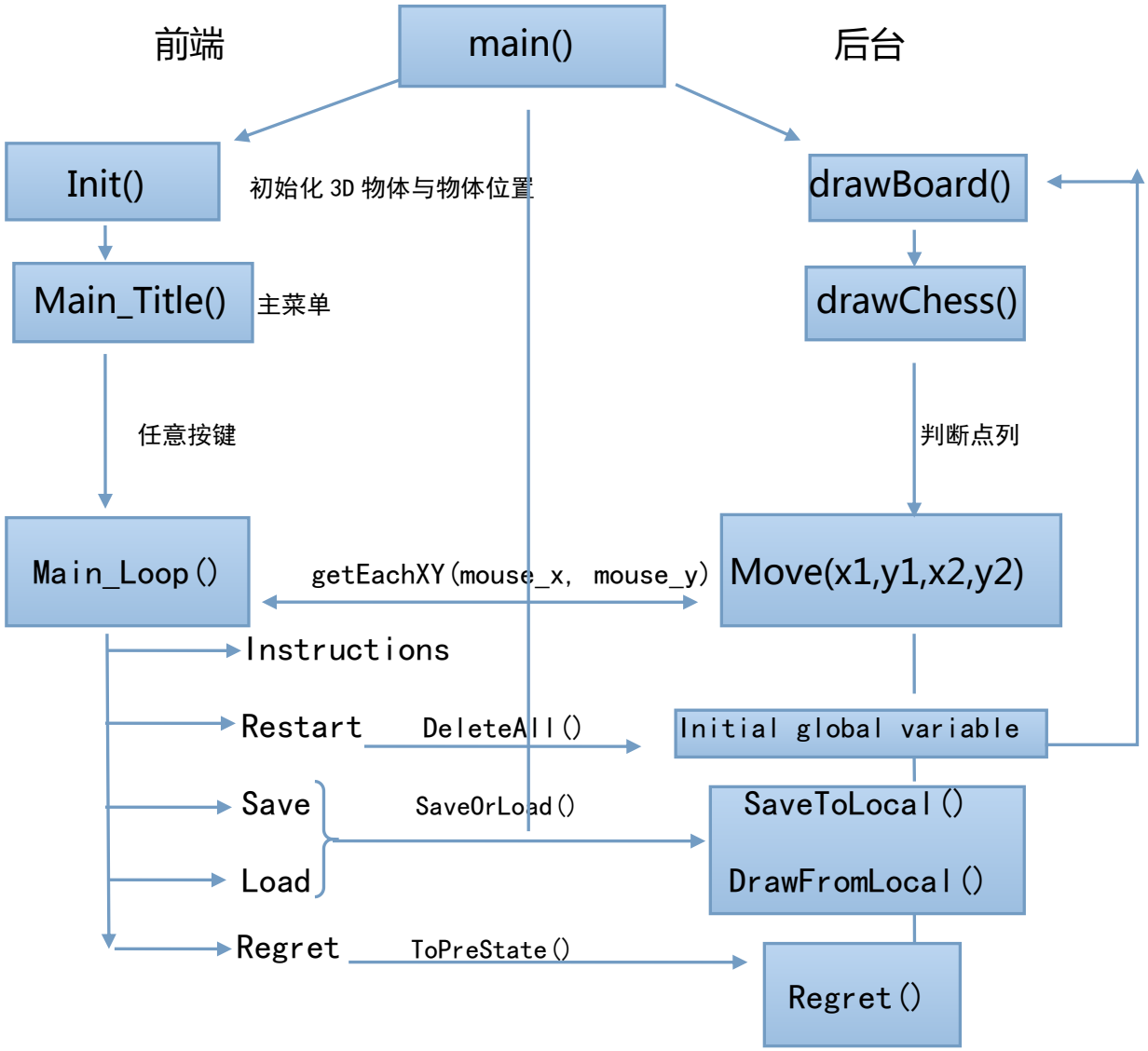
存盘功能：通过向本地文件中写入 dic、log 等状态量的全部内容实现存档，为了合理的和界面交互，暂定有八个可选存档区域，而文档命名暂不提供输入方式，而以选定的系统时间为准。

清零函数：保证每次读档前所有相关全局变量在内存中清零，不会和当前游戏冲突。

读档函数：以界面传来的文件编号读取指定文件中的 dic、log 等信息，重构全局变量。同时提供清除所有存档的功能，方便操作。

为了方便界面传入变量，将 save 和 load 函数判断放在一起，以此来进行选择。同时为方便拷入游戏，每个存档所在文件夹都会自动创建，不会引发 IOexceptions。

2.2 程序框架



2.3 GUI 设计

2.1 中对 GUI 设计已有一个基本的描述，精炼成八个字，就是“分层设计，异步更新”。

“分层设计”是基于本游戏 3D 特性的特有的切换界面的方式，也是保证“异步更新”的前提。由于空间中 3D 不透明物体对其下方物体的遮挡特性，我们将多个界面在同一条纵轴上依次排列，并通过切换摄像机的方式在不同界面上进行切换。这么做一是使特效上看上去更加酷炫（个人感觉，因人而异），二是可以在每次绘图时都绘制所有物体，即物体

的绘制与否与当前所在的界面没有关系，这就保证了接下来的一个大大减少工作量的技巧——“异步更新”能够正常工作。

“异步更新”的精髓是在于用一个全局变量的列表存储所有物体的位置，并在每一帧绘图的时候按照列表物体的位置进行绘图。这样一来，我们只需要在需要改变任何物体的位置的时候，修改一下全局变量列表的空间坐标，无需进行任何其他操作，下一帧绘图的时候物体的空间位置即可改变，而对其他任何物体的位置没有任何影响。只需要这一简单的操作，即可实现本程序的所有物体移动：

按钮按下——列表对应按钮位置的坐标 y 坐标下降

按钮弹起——列表对应按钮位置的坐标 y 坐标上升

棋子提起——列表对应棋子位置的坐标 y 坐标上升

棋子移动——列表对应棋子位置的 x、z 坐标移动

棋子落下——列表对应棋子位置的坐标 y 坐标下降

棋子死亡——列表对应棋子位置的坐标 y 坐标下降直至沉入棋盘

即只通过最为简单的操作，就可以实现界面中所有物体的相对移动。而且由于本程序“分层设计”的特性，坐标的改变不需要考虑当前该物体是否需要绘制——因为所有的物体都被绘制。这种结构使得代码的复杂度大大减小。

象棋游戏大家都玩过，做象棋游戏的 GUI 也不用对界面友好性做太多顾虑，只要按着平时的象棋游戏的风格来，不太会出现用户不会玩游戏的情况。本游戏的图片素材中，除了棋盘和棋子是找的网络资源，其他的素材都是自己找简单的素材用 PS 处理制作，没有花太大心思，也谈不上多美观，只能说让人能够接受而已。在这方面也没有太多好说。

指令的接收与处理方面，由于涉及到 3D 处理这一块，总体上比 2D 游戏还是要复杂不少，但好在 OpenGLLibrary 给我们封装了 glLibUnProject()这一函数，这一函数可以将 pygame 的 pygame.mouse.get_pos()的值转换为鼠标当前指向的最表面的物体的 OpenGL 中物体的三维坐标。象棋游戏对 3D 操作的精确度并不高，因而有了这个简单的转换函数，处理象棋游戏里的种种情况已经绰绰有余。

三 代码说明

※3dChineseChess.py

glLibWindow()：初始化 OpenGL-pygame 交互窗口
glLibView3D()：初始化 3D 视点
glLibCamera()：初始化摄像机
glLibLighting(True)：启用光源
glLibLight()：初始化光源位置
Texture = []：初始化素材列表
Texture.append(glLibTexture())：添加素材
Font = pygame.font.Font()：初始化字体
def GetCoordX()、def GetCoordY()：将投影坐标转化为棋盘坐标
def GetMessX()、def GetMessY()：将棋盘坐标转化为投影坐标

def Init()：初始化物体列表
- ├glLibObjTexCube()：创建有素材贴图的立方体
- ├glLibObjTexCylinder()：创建有素材贴图的圆柱体
- ├glLibObjTexDisk()：创建有素材贴图的圆盘
- ├glLibObjText()：创建文字
- └Objects=[]/Objects_pos[]：物体列表与物体位置列表，通过 append 操作添加元素

def GetInput()：得到输入量
- ├pygame.mouse.get_pressed()：鼠标按键当前状态
- ├pygame.mouse.get_rel()：鼠标指针移动方向
- ├pygame.key.get_pressed()：键盘按键当前状态
- ├pygame.mouse.get_pos()：鼠标指针当前位置
- ├glLibUnProject()：将鼠标指针位置转化为指向 3D 物体表面点的空间坐标
- └Camera.set_target_pos()：更改摄像机位置（视点不变）

def Update()：更新摄像机
- └Camera.update()：更新摄像机

def DrawChessboard()：绘制物体（所有物体，不只是棋盘）
- ├glBindTexture()：贴贴图
- └Objects[i].draw()：画物体

def Draw()：绘制界面
- ├Window.clear()：清空当前窗口
- ├Camera.set_camera()：设定摄像机
- ├Sunx.enable()：启用光源

┗Sunx.draw()：绘制光源

def Instruction()：“游戏说明”界面
┗Camera.set_target_center()更改摄像机视点

def Restart()：重开游戏

def Save()：“保存游戏”界面
┗SaveOrLoad()：与后台程序接口，保存游戏到文件

def Load():
┗SaveOrLoad()：与后台程序接口，读取文件到游戏

def Regret():
┗toPreState()：与后台程序接口，悔一步棋

def Main_Title()：欢迎界面

def Main_Loop()：主循环
┣getwinner()：与后台程序接口，得到当前胜利者
┗getEachXY()：与后台程序重要接口，得到走棋操作的反馈

※support.py

```
class Unit:#棋盘类
  def __init__(self,px=-1,py=-1,occupied=-1):#构造函数
  def getLoc(self):#得到棋盘坐标
  def getState(self):#得到所属队伍
  def changeState(self,val):#改变所属队伍
  def markBase(self):#九宫格标记
  def getBase(self):#返回九宫格
  def markLand(self,team):#国界标记
  def getLand(self):#返回国界标记
class Chessman:#棋子基类
  def __init__(self,px=-1,py=-1,team=-1,num=-1):#构造函数
  def getType(self):#棋子类型
  def getLoc(self):#棋子位置
  def changeLoc(self,px,py):#改变位置
  def getTeam(self):#返回队伍
  def getNum(self):#返回特有序号
  def move(self,absx,absy):#棋子移动
class Bing(Chessman)#兵类
class Shi(Chessman)#士类
class Xiang(Chessman)#象类
```

```
class Ma(Chessman)#马类
class Che(Chessman)#车类
class Pao(Chessman)#炮类
class Shuai(Chessman)#将类
def check_Notblocked(x1,y1,x2,y2):#检查棋子间是否有阻隔
def check_OneBetween(x1,y1,x2,y2):#检查棋子间是否有且仅有一子
def refreshChess(ori_px,ori_py,absx,absy):刷新棋盘
def addLogst(stx,sty,edx,edy,exist):#添加日志
def Move(stx,sty,edx,edy):#棋子移动
def getEachXY(px,py):#处理界面传入点坐标序列
def drawBoard():#初始化棋盘
def drawChess():#初始化棋子
def CheckChess(ex,ey):#得到棋子编号
def SaveToLocal(nm):#保存当前状态到文件
def DeleteAll():#清空当前状态
def DeleteSaves():#清空存储记录
def getLocalfiles():#得到本地文件列表
def drawChessFromLocal(nm):#从文件恢复已存储的棋局
def SaveOrLoad(ap):#选择读档或存档
def regret():#一步悔棋
def toPreState():#一次（二步）悔棋
def getwinner():#判断赢家
```

四 项目分工及贡献

彭乾旸：前端部分，即 3dChineseChess.py 中的全部内容。
贡献比例：50%

朱鸿儒：后台部分，即 support.py 中的全部内容
贡献比例：50%

五 感想

象棋游戏在我们看来是一款比较基础的游戏，由于其规则简单，玩法固定，并没有太大的编程难度,但是这样一个程序在完成内核和界面接口对接时还是出现了许多待解决的 bug，其中有许多是在单独运行算法程序时就存在的，却未被检测出来，还有一些是交互时产生的新问题，这些都说明做出一个完整的游戏有多么不容易，在这里向做出了 Warcraft 和 DotA 的程序猿致敬。同时我们的程序即使现在也并非完美的,如果有更多的时间可以做出更多的功能,比如联网对战、人机对战等。不过这些都是有着更高难度要求的,也会在设计算法方面迎来挑战，比如人机对战可能设计 A*算法等启发式算法、同时也需要将各种棋谱程序化，看似简单但工程量会很大。不过如果有时间、有机会再来完善我们的象棋程序的话，一定可以使它日臻完美，同时也可以期待像哈利波特与魔法石中的立体棋盘和棋子（其实这是我们的最终完整版的理想状态），配上虚拟音效，啧啧。当然，这一切都要拜托彭乾旸同学来一起实现了。象棋游戏虽小，但却由此体会到了工程中合作的重要性和调试的不可忽略性，也算是在 CS 的道路上又迈出了一步。最后感谢老师课上对 python 系统的讲述，以及助教们在上机课上给我们带来的实践体会，感谢大家的付出，最最感谢同组彭乾旸同学的辛苦的码农工作，和在 python 合作编程上给我带来的宝贵经验和体会。

朱鸿儒

在这门课大作业选题上面犹豫了很久，最后还是决定做象棋。作为电院的学生,我们希望能在这门面向全校的课程中,能在完成基本要求的基础上有所创新。想了很多创新的方法，比如做四人象棋、做带 AI 的象棋等等。这些想法都因为可行性或者难度方面的原因被 pass 了，最后我们还是决定做一个 3D 象棋 1.

目前在百度上面，似乎还没有关于用 pygame 做 3D 游戏的介绍。至少我找遍百度，没有一个现成的告诉我们怎么做一个 3D 游戏的教程。我本身也在这方面没有什么经验，在啃了几天 pygame 和 OpenGL 的资料之后还是感觉没什么头绪，还在错误的道路上试验许久，进行了大量的计算，浪费了不少时间，当时一时间还有着想要放弃 3D 的念头。最后还是在我翻找 pygame 官网上海量的说明文档的时候，在一个角落里面看到了 OpenGLLibrary 这个库。这个库挺冷门的，用的人也很少，但是对于我们这个象棋游戏，这个库却刚刚合适，有了这个库，几乎做

这个象棋游戏所有的困难都迎刃而解。

在错误的道路上滚打摸爬、从 0 基础开始学习 OpenGL，在参考资料奇缺的情况下搜集资料，最后能做成这个 3D 的界面，对我来说是一次不小的挑战，在这一过程中我也学到了很多。不过还好 ，我们的大作业最终达到了令人满意的效果，我一切的努力也都是值得的。

感谢组里的另一名成员朱鸿儒。他的工作效率非常高，在我对界面还不得要领、一筹莫展的时候，他就已经把后台算法部分写完了，这种工作效率是我所望尘莫及的。也多亏了他高超的编程能力与效率，除了实现一个象棋游戏后台的所有功能以外，程序界面与后台之间的所有接口函数也几乎都是由他实现，并且完成调试与修改。在这个象棋游戏里面，我给他的几乎都是最基本最基本的参数，而他则会把我可以直接拿过来就用的参数返回给我。就好像使用自动售货机的时候，我只用投币，就可以拿到直接能喝拿饮料，而中间的所有过程都是由他完成的（好像不怎么恰当，大概就是这么个意思- -）。这个的实现难度并不小，而他基本上都能在一天以内实现，最后我们能够按时完成这个象棋游戏，他起到了至关重要的作用。

然后说说对这门课的感想。作为电院第一学期就学了 python 的学生，我选这门课的目的是以复习 python 为主，本没有想过会学什么新的东西，但是事实却出乎我的预料。这门课虽然从零基础开始教起，上课的内容较为基础，但是课后作业的广度很大，我们从中能够了解到 python 的很多非常有用库与知识。例如操作 excel 表格的 xlrd 库、绘制图表的 matplotlib 库、PEP8 编程规范等等，这些库能够解决我们在今后的学习乃至生活中遇到的很多问题。课程的大作业涉猎范围更是广泛，可以说做完了任何一个大作业，我们都会对 python 的某一个领域的应用有一个较为深入的了解。这种深入浅出的课程让我感觉收获良多。当然，这是与老师高超的教学艺术与助教们的辛苦付出分不开的。

最后希望期末能考个好成绩%>_<%

彭乾旸