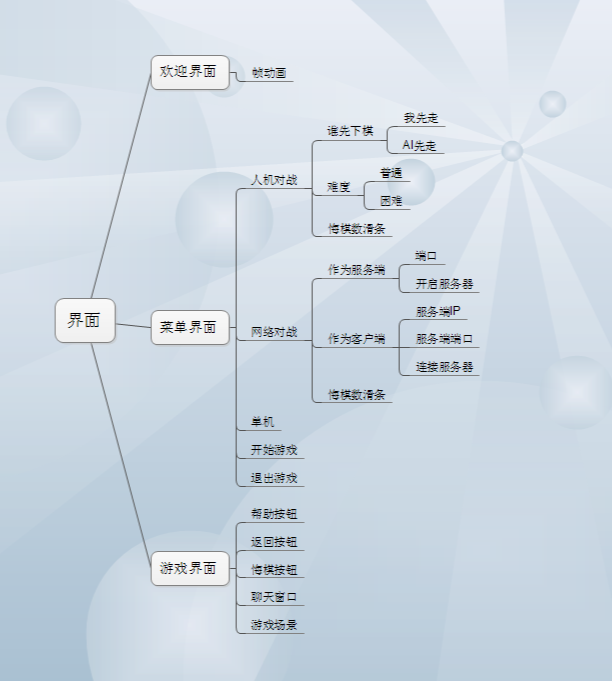
# 详细设计方案

## 界面



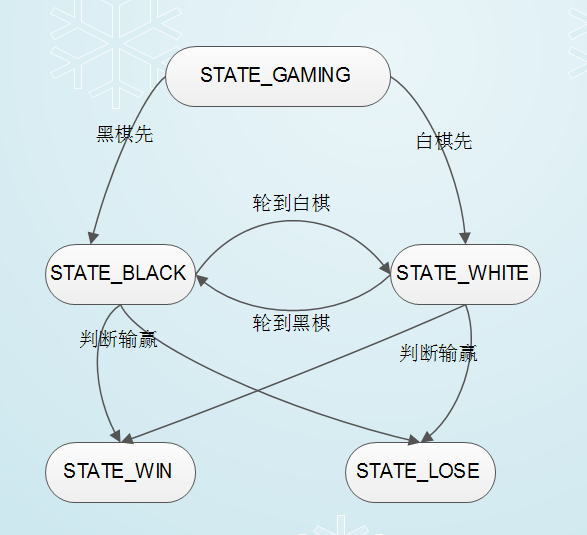
棋盘，棋子：3dsMax 制作。

天空盒的旋转：加一个摄像机，设置天空盒渲染，景深设的比主相机低。

## 游戏控制器

由状态机控制

*//游戏状态***public** **const** **int** STATE\_BLACK **=** 1**;**   *//黑棋走***public** **const** **int** STATE\_WHITE **=** 2**;**   *//白棋走***public** **const** **int** STATE\_WIN **=** 3**;**     *//胜利***public** **const** **int** STATE\_LOSE **=** 4**;**    *//输了***public** **const** **int** STATE\_GAMING **=** 5**;**  *//游戏中*



## AI

*\*成5:即构成五子连珠, 100分  
\*活4:即构成两边均不被拦截的四子连珠、双死4、死4活3， 90分  
\*双活3， 80分  
\*死3活3， 70分  
\*死4:一边被拦截的四子连珠， 60分  
\*活3:两边均不被拦截的三字连珠， 50分  
\*双活2， 40分  
\*死3:一边被拦截的三字连珠， 30分  
\*活2:两边均不被拦截的二子连珠， 20分  
\*死2:一边被拦截的二子连珠， 10分  
\*单子:四周无相连棋子, 0分*

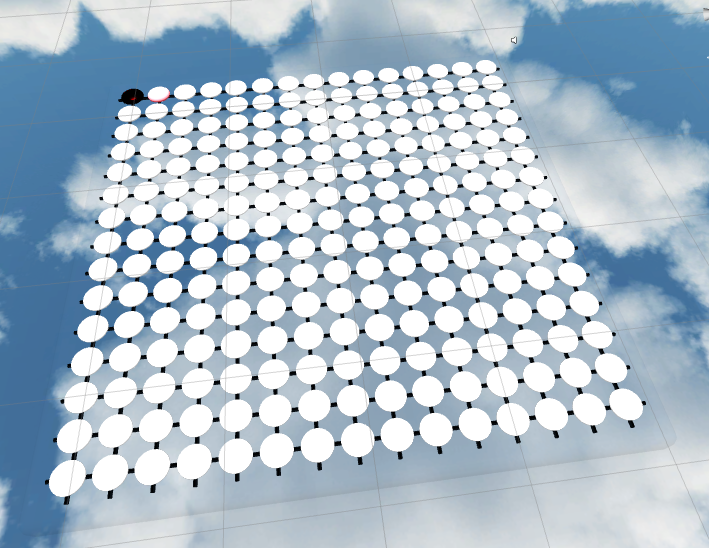
根据这个评分标准，写评分函数，获得每个位置的评分，取评分最高的作为AI落子位置。

## 网络

采用Unity3D的RPC机制进行聊天信息，网友下棋信息的发送。

## 下棋操作实现

设计时，所有对象都赋予刚体属性，都摆放好，如下图：



游戏开始时，每个棋子都会被隐藏。当鼠标左键点击抬起时，从主Camera朝光标所在位置发送Ray射线对象，当ray 碰到棋子对象时，根据当前的游戏状态，对棋子赋予相应的材质，并显示。

## 数据结构

**private** **char[,]** gameData **=** **new** **char[**ROW\_COUNT**,**COLUMN\_COUNT**];**    *//存储游戏棋盘信息,'b' 代表黑子，'w' 代表白子*

保存棋面。

**public** **class** Point  
**{**  
    **public** **int** x**;**  
    **public** **int** y**;**  
**}**

棋子位置信息。

**private** Stack record **=** **new** Stack **();**    *//记录每步的栈,用于悔棋*