《数字图像处理基础》课程实践

结题报告

项 目 名 称 ： 基于视觉的俄罗斯方块游戏机器人

姓　　名 ： 刘启明

学　　号 ： 516021910585

联 系 电 话 ： 13262921489

电 子 信 箱 ： liuqiming666@sjtu.edu.cn

合 作 人 姓 名 ： 张沛东、陈思哲

2019 年 6 月

1 研究内容综述

本次课程大作业中，我们意图通过一个普通 RGB 摄像头拍摄平板上的游戏界面，通过图像处理、游戏状态分析等方法给出游戏策略，并根据策略控制机械臂进行相应动作。经过讨论，我们小组决定选择俄罗斯方块游戏。根据任务需求，我们将整个任务分为了三个模块：图像处理与状态提取、游戏策略计算、机械臂控制。我（刘启明）负责图像处理与状态提取的工作，张沛东和陈思哲完成游戏策略计算和机械臂控制的内容。

对比开题报告中确定的研究目标，我将在本节中简要介绍各目标和性能指标的完成情况，并对我们小组的项目效果作简要概括。

首先，在开题报告中，我们希望将摄像头采集的图像处理后得到游戏的状态信息。对于俄罗斯方块来说，所谓“游戏状态”，就是判断哪些地方存在方块、哪些地方是空白。我们首先提取出了游戏界面左上角白色边界框的位置，然后依据边界框与方格之间的确定性位置关系，依次扫描所有方格，并根据颜色深浅的判断该位置是否填充了方块。经过实际测试发现，摄像头能够实时地将图像转化为当前游戏状态，并存储在一个字符串中。在多次、长时间的测试下，如果标定准确，我们没有发现状态识别错误的情况，这说明我们图像识别与信息获取部分的实现效果稳定优秀，指标达到了预期。

其次，得到游戏状态信息之后，我们还需要利用某种算法，利用当前信息进行决策。我们参考了 Pierre Dellacherie 算法，并根据我们的游戏环境进行了调整和改进。我们将游戏状态传入算法，希望程序输出一个合理的控制策略。通过在电脑上的虚拟测试，以及在平板电脑上进行实际测试，我们发现决策算法给出的策略都非常合理，往往与人的决策不谋而合。因此在此部分中我们完成了预期目标的要求。

最后，我们还希望算法给出控制策略后，机械臂能够根据策略做出正确的反应，这一部分涉及到很多与硬件的交互。虽然对机械臂的控制可以调用函数实现，但是在实际测试中，机械臂的动作经常会与程序输入的期望动作不一致。例如，执行点击动作时，触控笔下落高度不一致（这直接导致了点击不到或点击多次的问题）。我们通过调整硬件与软件延时解决了大部分问题，游戏得以进行。但是在少数情况下，还是会出现误触、未接触、接触多次的问题，这已经成为了影响游戏成绩的最大因素。我们通过调整，将这种影响降到了最低，但是还有进一步改进的空间。

总的来看，项目各项指标符合预期。通过最终测试，整个系统能够消除方块 200 行以上，这已经超出了我们组三位成员的个人成绩上限。项目进行顺利，一方面是因为前期调研比较充分，技术路线务实；另一方面是因为小组成员的全心投入和助教的全力协助。在接下来的章节，我将详细介绍项目实现的具体方案和一些细节的处理方法，并对项目过程中的一些问题进行讨论。

2 研究方案

在本节中，我将以图像处理与状态提取、游戏策略计算、机械臂控制三个模块分别进行介绍。出于行文条理考虑，我在本节仅对我们的最优方案进行介绍，以逻辑顺序为序，对各种问题的讨论将在 4.1 节中进行。

2.1 图像处理与状态提取

我们游戏界面的具体形态如图 1 所示。游戏环境高 20 个格子、宽 10 个格子，在上一个图形下落之后，上方会继续弹出下一个图形。当下方某一行被方块填满时，这一行就会被消去。如果方块达到了天花板处，游戏即结束。

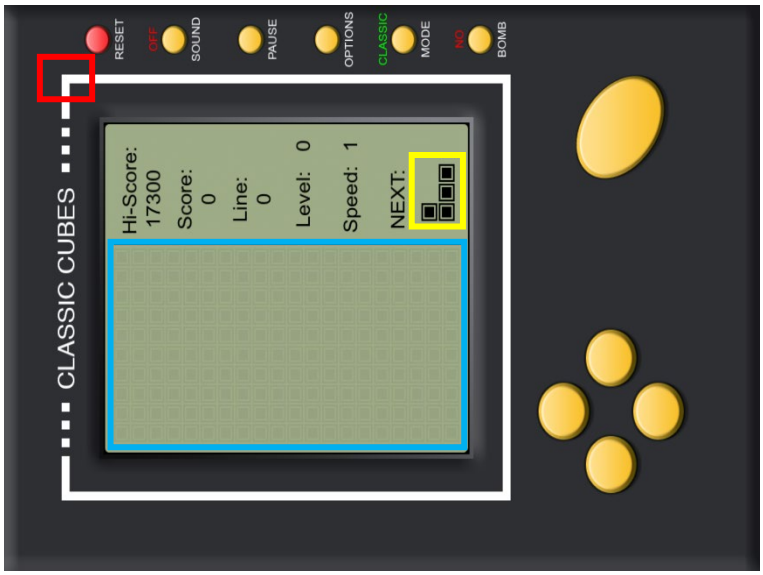

图 1 游戏界面形态

观察这张图片的特点，可以发现游戏界面的背景颜色是深灰色，游戏区域与操作区域被一个很粗的白色方框隔离开。我们最需要关注的就是图 1 中蓝色方框内部每一个方格中是否有方块，如果没有方块，相应的格子里就呈现出浅灰色；如果有方块，相应的格子中便会是黑色（如图 1 中黄色方框所示）。

我们采用了基于标定物的信息提取方法。从图 1 中可以观察到，在整张图片的左上方有一个明显的白色直角（如图 1 中红色方框所示），与背景黑色对比十分明显，找到它很容易。同时，这个白色直角与蓝色方框的相对位置是固定的，找到了白色直角也就能够知道每个方格在图像中的位置。依据前文所说，相应位置的深浅意味着方块的存在与否，我们只需要扫描前面找到位置上的灰度值，即可知道该位置是否存在方块。具体操作介绍如下。

首先我们需要将 RGB 摄像头采集的彩色三通道图像转化为灰度单通道图像，因为颜色信息对我们来说并不重要，可以简化。观察图 1 黄色方框中方块的具体形态，可以发现方块并不是一个完全的黑色矩形，游戏中为了美化处理，将方块做成了黑色框嵌套黑色块的形状，如图 2 左侧所示。当我们扫描的像素点恰好位于两者之间的白色区域时，本来存在方块的格子就会判定为不存在方块，这会对状态判断带来极大干扰。为了解决这一问题，我们采用了

膨胀图像的方法，通过膨胀两侧的黑色区域去掉中间的白色条纹，膨胀后的结果如图 2 右侧所示，可以看到效果很好，避免了误识别的问题。

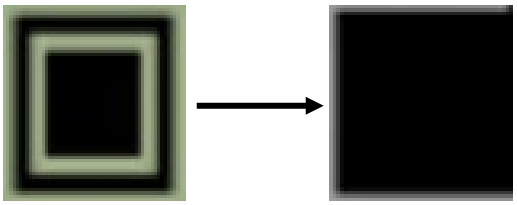

图 2 原始方块形状以及膨胀之后的图像

随后我们对整幅图像进行二值化。选取合适的阈值之后，我们能够将黑色背景、存在方块的方格等区域全部变为黑色；同时将白色粗边框、不存在方块的方格等区域全部变为白色。计算机此时能够看到如图 3 所示的图像，可见二值化和膨胀效果很好，达到了处理目的。

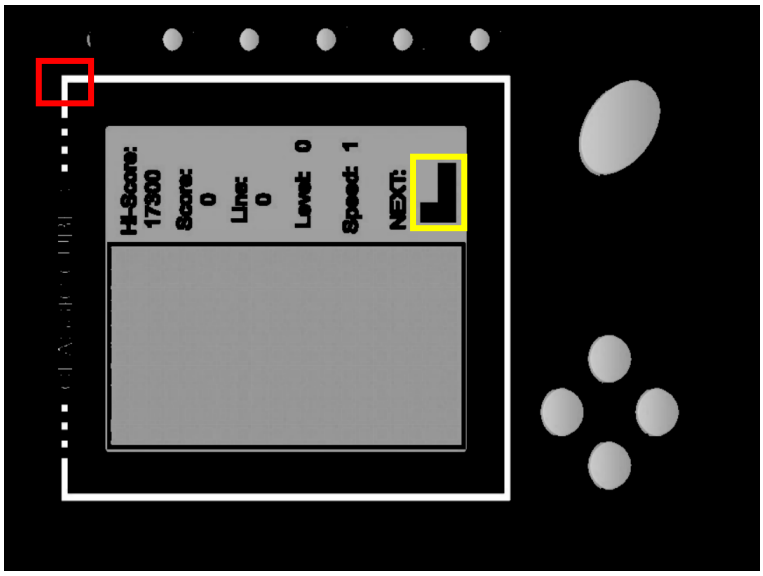

图 3 经过膨胀和二值化处理之后的图像

接下来进行信息提取的工作。根据之前的思路，我们首先需要查找到图 3 红色方框中标注的白色边角。由于此白色角在最左上方，所以我们采用了小区域搜索的方法，仅仅需要从小到大遍历左上角若干像素的灰度值，第一次出现白色的位置即为标定点。随后根据调试找到 200 个方格的具体像素坐标位置，设置一个长度为 201 的字符串。依次扫描这些位置的灰度值，当相应位置为黑色时（有方块），向字符串内写入 1，否则写入 0。扫描完毕后的字符串即储存了该时刻游戏的棋盘状态。例如在图 3 中游戏区域没有方块，返回得到的状态即为存有 200 个 0 的字符串。

图像处理与状态提取的完整过程即介绍完毕，但是还有一个非常重要的问题需要解决，即摄像头位置标定。我们在摄像头位置固定时，可以确定图像中标定点到方格之间的像素距离，一旦摄像头被移动，这个距离就会改变，对我们的重复实验造成麻烦。为解决这个问题，我们采用了矩阵参考点标定法，即在原始图像上叠加预期参考点的位置。开始游戏前，我们可以根据输出的红色参考点调整摄像头，直到参考点对准图像上的方格。标定参考点的图像

如图 4 所示。

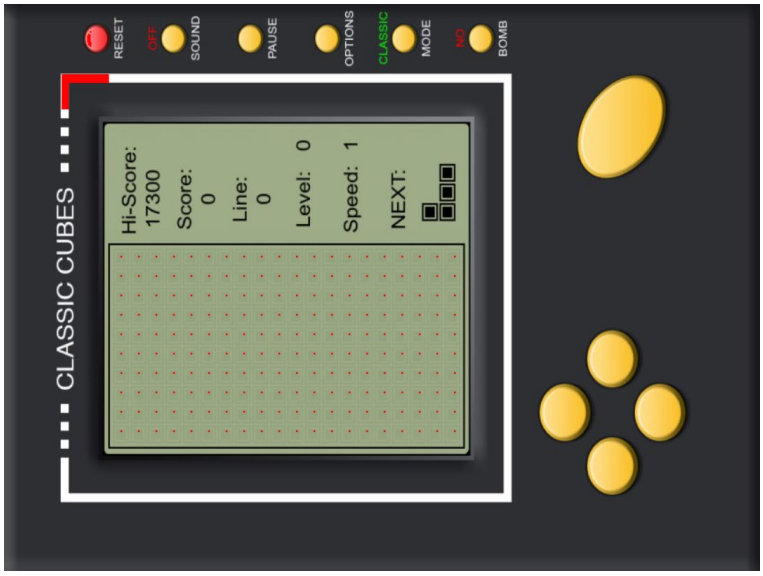

图 4 输出用于辅助标定的红色点，红点对准方格时即标定成功

以上即介绍了图像处理、状态提取和摄像头标定的全部处理细节。在整个项目过程中，我们还尝试了很多不同的图像处理和标定方法，具体讨论将在 4.1 节中介绍。

2.2 游戏策略计算

在 2.1 节中，我们已经得到了以二进制字符串表示的游戏状态信息，策略算法需要做的，就是根据游戏状态信息返回左移、右移、旋转等指令序列。

首先我们应该知道下落方块的形状。由于方块都是从上方落下的，所以我们通过扫描顶部三行的信息就能够得到方块的形状。在俄罗斯方块的游戏中，方块总共只有七种组成形态，每一种组成形态都可以进行旋转。它们在计算机中的存储形式具体如图 5 所示，其中#代表方块的旋转中心。

图 5 所有可能的方块形态

我们采用的算法是经典的 Pierre Dellacherie 算法。该算法的基本原理非常简单，就是基于当前的状态穷举所有可能的落点，并根据落下之后的情形进行评分，选取评分最高的情形进行控制。由于我们的游戏区域高为 20，宽为 10，所以需要考虑的情形实际上并不太多。基于这一原理，可以发现算法的核心部分实际上是一个评估函数，我们需要对评估函数进行合理设计，使之能够真实地反映一个状态的优劣。对于评估函数，我们有 6 个状态参数值可以使用，下面将逐一介绍。

Landing height（LH）表示方块下落后的堆积高度，显然方块下落后堆积高度越低越安全。Eroded piece cells（EPC）表示当前方块下落后，消除的行数和此方块中参与到行消除的方块数目的乘积，体现的是方块对消除的贡献程度。Row transitions（RT）表示每一

行从左到右扫描，方块有或无的变化次数之和。从有/无方块的格子到相邻的无/有方块格子计一次变换；其中，边框算作有方块。Colum transitions（CT）表示每一列从上到下扫描，方块有或无的变化次数之和。从有/无方块的格子到相邻的无/有方块格子计一次变换；其中，边框算作有方块。Number of Holes（NH）表示空洞的数量。空洞指的是每列中某个方块下面没有方块的空白位置，该空白可能由 1 个单位或多个单位组成，需要说明的是，空洞会对后续游戏造成严重影响，所以在方块落下后产生的空洞越少越好。Well sum（WS）表示“井”的大小之和。“井”是指某一列中两边都有方块的连续空格，边框算作有方块。例如，若一个井由 1 个方块组成，则为 1；若由连续 3 个组成，则和为 1+2+3=6。如果一个井的顶部有方块的存在，则该“井”同时也是空洞。

在调节算法参数的过程中，实际上就是在调节上述六个参数的权重，最后通过各分量相加实现评估函数的计算，对每一种可能的情形都进行计算，选取最好情况，输出到后续的机械臂控制环节。

在代码实现上，我们很希望利用 Python 完成决策部分的编写，因为 Python 书写简单清晰，而且功能更加丰富，对类似优化功能的支持也较好。但是我们的程序框架语言是 C++，需要实现在 C++中调用 Python 脚本的功能。我们最终选择使用 Python.h 库，它能够将 Python 文件中的函数封装成对象，给定输入即能够传出输出。在 C++中调用 Python 极大地拓展了我们决策算法的功能，我们能够用熟悉简洁的方式实现非常复杂的功能，这为我们俄罗斯方块取得优秀成绩打下了很好的基础。在调试环节，我们将决策算法从虚拟环境的 Pygame 中剥离出来，接入到 C++中，并根据我们游戏的实际环境设置参数，最终的实测结果非常优秀，符合预期。

实践证明 Pierre Dellacherie 算法是一个相当优秀的贪婪算法，所提出的六个关键指标都非常成熟，能够准确反映游戏的具体状态。我们在 Pygame 模拟了上述算法，并根据实际情况进行了参数调整，最终确定评估函数的计算公式为：

$$50 \times \mathrm{LH} + 40 \times \mathrm{EPC} - 32 \times \mathrm{RT} - 98 \times \mathrm{CT} - 79 \times \mathrm{NH} - 34 \times \mathrm{WS}$$

采用此公式，在模拟游戏中能消除 5000 行左右，这个成绩足以让我们达到最初设定的目标。

通过评估函数选择最优的控制方式之后，我们需要传输到 C++中并控制机械臂，这涉及到控制方式的表达。游戏中我们仅仅需要做三个动作，即左移、右移、旋转。我们分别用 1、2、3 代表这三个动作，并写入到一个 8 位长度的字符串中，不足的用 0 补齐。例如，字符串 33100000 表示先让方块旋转 2 次，再使其向左平移一格。通过理论推导，我们发现一次控制序列最多会有 8 个动作，这也是我们控制字符串长度选取为 8 的原因。

在实际测试中，随着消除行数的增多，方块下落的速度也会逐渐加快，长条形和 L 形方块的出现概率会下降，游戏的总体难度也会上升。我们还考虑了动态优化评估函数的方法，在后期让算法倾向于消除更多函数。事实上，1988 年俄罗斯数学家已经证明，如果只出现 S 和 Z 型方块，游戏最终都会走向失败。

2.3 机械臂控制

机械臂控制涉及到软件与硬件的交互，因此存在诸多复杂的情况。当方块下落时，我们能够对其进行三种有效操作：按上方按钮将其顺时针旋转 90 度；按左侧按钮将其向左移动一列；按右侧按钮将其向右移动一列。于是我们可以将这三种动作用一个封闭的函数来表示。初始时将触控笔放置于四个按钮的中央。要点击某个按钮，就先移动到按钮的绝对坐标位置，点击后移动到下一个指定位置。

当接收到 AI 返回的策略字符串时，程序将从头到尾扫描。当扫描到“1”时，机械臂移动到左侧按钮的位置并点击一次，使下落方块左移；当扫描到“2”时，机械臂移动到右侧按钮的位置并点击一次，使下落方块右移；当扫描到“3”时，机械臂移动到上方按钮位置并点击一次，使下落方块顺时针旋转 90 度；当扫描到“0”时，说明本轮调整已经结束，机械臂静止不动，等候下一次控制策略的到来。

在控制机械臂时，由于硬件原因，我们需要进行适应性调整。下面一一介绍。

首先，由于机械臂完成某一动作是需要时间的，如果它接收到一个新的指令，就会立即中断当前动作转而完成最新的指令。因此，在调用机械臂控制函数之后，需要加入一个延时函数，让机械臂做完当前动作之后再执行下一步。由于系统内置的 sleep 函数会导致视频流的卡顿，我们通过 while 循环手动实现了延时的功能，避免了机械臂动作冲突的问题。

其次，我们在调用 comHitOnce 函数时，发现机械臂下落距离不一，导致触控笔有时点击不到屏幕，有时又多次点击。我们通过调节硬件解决了此问题，即在每次游戏开始之前将触控笔放置在恰好合适的高度，无论机械臂下落深或浅，触控笔都只能恰好点击屏幕一次。

最后，由于俄罗斯方块的游戏时间很长（一小时以上），我们发现机械臂在长时间工作后会偏离原来的标定点，游戏中需要时刻观察。当机械臂偏离时我们快速手动复位即可。

2.4 系统控制小结

本节中，我将对整个系统做简单的小结，明确各个环节的相互关系及实现方法。我们对整个项目的开发遵循如图 6 所示的步骤，在实际操作中虽然有所调整，但大体上顺序不变。

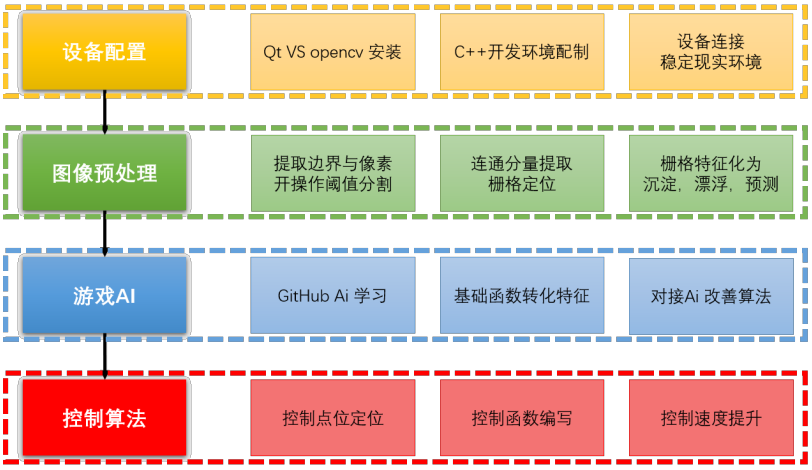

图 6 项目开发的大致逻辑和步骤

整个系统在工作时遵循一套完整的逻辑，简要的流程图如图 7 所示。每一个细节部分都在 2.1~2.3 节的介绍中详细提到。

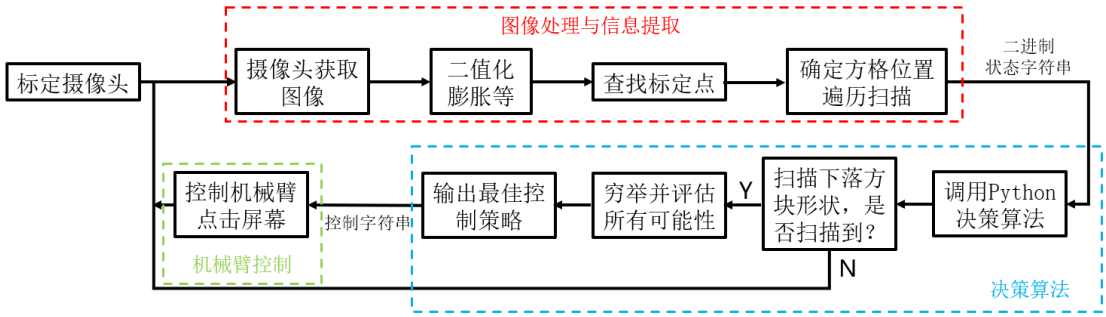

图 7 系统工作流程简图

系统工作时，iPad、机械臂、摄像头、电脑等设备的摆放位置如图 8 所示。注意到，摄像头应该放置于 iPad 的左侧，这样摄像头接收到的图像方向就跟图 1 中的一致。

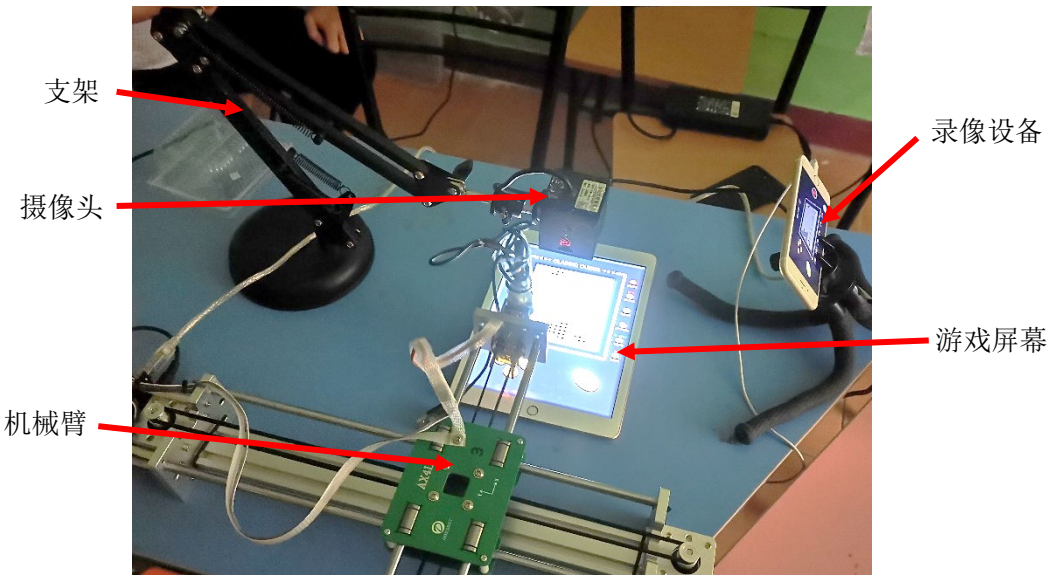

图 8 游戏中设备摆放概览

3 研究成果

在本节中，我将结合开题报告中的预期计划和任务目标，详细介绍我们小组的项目成果。本节主要分为时间进度和任务完成情况两个部分进行说明。

在项目进度方面，我们完成目标任务的速度非常快，感觉时间并不是特别紧迫，因此在后期得以有充分的时间调整系统以获得更好的游戏成绩。由于项目刚开始时，我们花费了大量的时间进行预研，对整个任务的流程有了清晰的了解，这让我们的开发过程比较顺利。项目从第 10 周开始，我们利用大约两个星期的时间查阅资料、编写代码的初级版本。从第 13 周开始我们进行全系统测试，并在这个过程中不断完善软件和硬件，使得俄罗斯方块游戏能够消除更多行数。第 14 周我们进行了第一次正式测试，能够消除 104 行。但是通过询问助教发现，这个成绩并没有打破前几届学长学姐的纪录，本着挑战最高纪录的目标，我们又进

行了一些软件参数的调整，并在 17 周进行了重新测试（考试周期间暂停约两个星期）。第二次正式测试时，已经能够连续消除 224 行，整个系统的性能得到了较大提升。

基于任务出发，我认为我们很好地完成了预期目标，某些性能表现甚至在我们的意料之外。图像处理方面，我们能够根据游戏界面的特点制定针对性的图像处理策略，整个过程中，我们使用了非常简单明了的方法即完成了看似复杂的信息提取工作，将图像中的状态信息转化为了每一个方格中的占用状态，并用二进制字符串加以表示。图像处理与信息处理过程的简洁性占用了极少的计算机算力资源，从而为我们的游戏提供了很好的实时性，这在俄罗斯方块游戏中是十分关键的。

在游戏策略方面，我们参考了 Pierre Dellacherie 算法，并根据实际游戏环境进行了改良，达到了非常好的决策效果。首先我们配置了 C++调用 Python 的环境，将决策算法封装在 Python 文件中并形成黑箱。图像处理得到状态字符串后，C++自动将状态字符串传入决策黑箱中（调用 Python 文件），算法就能够几乎实时传出决策（得到 Python 文件的返回值并继续 C++主程序）。将 AI 算法给出的控制决策与我们人自身的判断进行比较，我们发现 AI 都能给出与人一致的决策，有些时候甚至做出更加长远和聪明的判断，几乎没有失误。总的来说，决策算法是非常稳定可靠的，只不过由于实际游戏中的硬件限制，我们也必须对算法中的一些参数和功能进行改良，删去一些技巧性的功能，算法性能有所下降，这让我们非常遗憾。

在机械臂控制方面，由于软件参数与硬件执行动作不一致的影响，它成为了我们游戏难以消除更多行数的罪魁祸首。我们依据决策算法返回的策略对机械臂进行控制，但是机械臂经常会出现点击两次或点击不到屏幕的情况。通过调整触控笔的高度、机械臂动作延时时间等参数，最终能够较为稳定地执行预期的控制动作。

综合来看，在我们的第二次正式测试中，机械臂能够通过点击屏幕消除 224 行的俄罗斯方块，这已经远远超出了我们组三位成员的个人成绩上限。值得说明的是，如果经过更多参数调整及硬件调试，我们有信心让整个系统的成绩进一步上升。

4 研究总结

4.1 研究中的问题与讨论

课程项目进行的过程中，一些问题常常会给我们的任务实现带来困难，在尝试解决这些问题的过程中，我们想到了不少解决方案，并根据实际效果进行了取舍。第二部分介绍的方法仅仅是我们最后使用的、效果较好的方法，在本节中，我将较为简略地介绍整个项目过程中遇到的一些问题以及解决方案。

第一。环境因素的影响会对摄像头提取信息带来极大困难。在摄像头对准 iPad 时，由于 iPad 屏幕镜面的反光作用，摄像头得到的图像会因为天花板上灯的照射而产生大块白斑。毫无疑问，反光会对图像处理中的阈值分割、膨胀等操作带来巨大影响，所以我们必须加以

避免。我们首先将 iPad 屏幕亮度调到最高，尽可能抑制环境中散射光的影响；我们还使用了一个塑料凳子作为支架，在上面覆盖不透光的纸，以此将 iPad 和摄像头放置在一个黑暗的环境中；在测试时，我们也可以不遮光，而是关掉天花板上的灯来抑制反光现象。这些简单的解决方法效果都非常不错。另外，当摄像头俯视屏幕时，机械臂前端固定触控笔的螺丝旋钮会挡到游戏区域的中间。我们放弃了用螺丝固定触控笔的方法，将旋钮取出不用，同时在触控笔上缠绕若干层卫生纸，加宽笔的直径使得笔能够嵌入到圆筒中，同时也增大了笔的摩擦力，防止多次点击后笔出现松动。

第二。摄像头的标定是我们必须解决的问题。摄像头一开始是被固定在支架上面的，而支架与 iPad 屏幕的相对位置并不固定，并且摄像头在支架上也会轻微旋转和移动。在不同时间进行测试时，摄像头位置的移动就会造成观察区域相对位置和大小的改变，对我们代码的普适性和鲁棒性提出了很高的要求。在第二部分中已经介绍，我们是通过标定点找到游戏区域的，通过遍历查找获得图像左上角的标定点，这样可以免除摄像头平移移动产生的影响。同时，我们还在原图中叠加了标定矩阵。在游戏开始之前，打开摄像头之后仅需要两步即可完成所有标定工作：先保证摄像头与屏幕完全平行（消除旋转影响，可以通过观察屏幕轮廓长度是否相等进行调节）；再将标定矩阵中所有的关键点对准游戏区域的方格即可。除了上述方法，我们还想到了四点标定法，即在相应像素处固定屏幕的边角位置，每次将屏幕边角对准图像上的标定点。经过思考，我们发现这种方法与标定矩阵的方法在原理上并无区别，所以最后并没有进行修改。

第三。在状态信息传入控制算法的过程中，为了知道当前下落方块的形状，我们选择扫描顶端的两行。但是当方块下落速度越来越快时，由于帧率等原因，程序很有可能漏掉上方的方格，导致机械臂没有任何动作。为了避免这种情况，我们将扫描的行数由两行变为了三行，这样留给了程序一定的时间裕量，上述问题不再发生，但是同时又产生了一个新的问题。即程序很有可能重复观察同一个方块，将其理解为两次不同的下落，这样机械臂就会执行两次相同的动作，这当然与我们的期望不一致。为了解决这一新问题，我们设置了一个判断逻辑延时，即在一次判断之后的 2 秒内不再判断第二次，这样方块就有足够的时间退出前三行区域，从而解决了问题。

第四。我们对原始的决策算法进行了大量修改，最终才能应用到我们的游戏当中，这个过程存在着许多理想情况与现实情况的矛盾与权衡。我们知道，俄罗斯方块在下落的过程中，只要还没有触及到下方的方块，就能够进行平移和旋转操作，一些诀窍是，先让俄罗斯方块以某一位置或形状移动到“内部缺口”附近，然后再通过变形和平移快速“嵌入”到“内部缺口”中。但是通过第二部分的介绍发现，我们的控制方法不允许我们完成这样复杂而又快速的动作。我们只能在方块下落的过程中调整好其形状和位置，然后让其以自由落体的方式下落。为了避免决策策略通过这种方式对方块进行控制，我们采用了“模拟填平”的方法，即让程序假装认为方块是无法进入到“内部缺口”中的，选取其它次优但是容易控制的方法

进行输出。其次，我们调高了控制策略对堆积高度的敏感性，当高度超过警戒线时，即便当前方块能够更好嵌入到较高的区域，我们也让程序趋向于优先消除行数以降低高度。通过这种方式，我们能够生存更久。

第五。从实验结果的角度出发，即便我们消除的行数足够多，最终游戏都会走向失败，我们需要仔细分析游戏失败的原因，才能在后续测试中取得更好的成绩。由于图像处理与信息提取部分的工作是绝对的，提取得到的信息只有对错之分，只要信息无误，游戏失败肯定就不是这部分的问题，经过检查，排除信息出错的可能性。在控制策略中，我们经过虚拟Python环境中的游戏测试,发现在同等难度下决策算法能够保证消除5000行以上而不失败，这与我们实际测试的结果相差巨大，最终问题锁定在机械臂等硬件环节。我们着重观察了机械臂的动作及相应方块的反应，发现游戏最终的失败都是由于触控笔重复点击按钮导致的。比如方块仅需左移一格时，若触控笔点击两次，方块就不能到达指定位置，而是在提升堆积高度的同时留下了很多空隙，这会直接导致游戏失败。尽管这种情形极少出现，但是一旦出现一次,就会对局面造成毁灭性的伤害。触控笔点击两次是由于机械臂下落高度不一造成的，内部函数我们无法修改，只能通过调节硬件来解决。我们在每次游戏之前将触控笔到屏幕的距离调节到合适的范围，先进行若干次测试，测试无误后再启动游戏。要想从根本上解决此问题，我们还想到用闭环控制的方法，即在机械臂执行动作之后观察图像上是否有正确的反应，如果控制错误进行反向校正即可。但是由于游戏中方块在不断下落，闭环操作存在时间不足的问题，可能动作还没有执行结束，方块就已经落下了。这个问题是我们项目中反复思考但还没有实质性解决的问题。

4.2 感想与体会

在半个学期的课程实践中，我们小组成员齐心协力、各有分工。我主要负责图像处理与信息提取的工作。小组成员的工作组合起来，共同完成了各项任务要求。在整个过程中我们对课程学到的知识充分加以利用，并结合工程要求，广泛查阅资料，在解决实际问题的过程中锻炼了各项能力。虽然最后结果符合预期，但是过程不总是一帆风顺的，我在这个过程中收获了很多知识层面和经验层面的东西，在此略作总结。

在这门课程的实践部分，我们通过摄像头获取数据控制机械臂，从而模拟人类的游戏过程，这种实践让我不仅对图像处理和人工智能的基础知识有了更为全面深入的了解，更让我看到了相关知识的重要应用价值，提升了对它们的兴趣。机器视觉和人工智能是当今的热门研究领域，本门课程对相关知识的大量应用，为我打下了良好的实践基础。

实践过程中，我还体会到了理论知识与实践的差别。以图像处理为例，在课程中我们虽然学习了很多经典的图像处理方法，但是在我们实际调试时才发现环境中的反光、遮挡等干扰难以消除，一些形态学的处理也很难进行。我们在实际情况中，往往是在理论的基础上对各种方法进行加工改进，使得整个算法适用于我们的控制系统。在其它工程问题中，这种情况也是普遍存在的，数字图像处理课程实践让我对工程有了更加深入的认识。

在项目进行的过程中，有进展快速的时期，也有瓶颈期。探索的过程中存在着诸多未知的因素，对待工程任务的心态非常重要。我们小组在处理机械臂下落距离不一的问题时，多次尝试软件代码和硬件调试，但都以效果不明显告终。我们曾尝试更换 Apple Pencil 进行触控，但是发现它更加灵敏，难以控制；我们也曾尝试过修改内部的机械臂点击函数，但是代码一直存在问题。但当我们冷静下来思考一两个星期之后，就能跳出之前的死胡同。与其它同学沟通交流之后，问题得到了很快地解决。这说明工程难题是家常便饭，我们不能一味消极应付，用合适的方法、独特的眼光和积极的态度，才是解决问题的正道。这门课程为我之后学习和工作提供了一些启示。

最后，我还想感谢我的两名队友，在与他们合作的过程中，我不仅完成了复杂庞大的工程任务，还锻炼了团队协作能力和沟通能力，知识和经验的相互交流更是让我学到了课本上没有的东西。

4.3 建议与致谢

课程运行中，助教及时有效地发布各项通知，及时解答同学们的疑惑，因此我认为在课程秩序方面已经非常完善了。我主要有四点建议。一是建议老师更早布置大作业任务，以便我们能够有更加充分的时间来完成。二是建议项目任务上应该更加侧重图像处理方面的工作，毕竟本次实践是数字图像处理课程的大作业，实际情况却是，我和其它同学感觉重点在游戏策略的设计上，这有些偏离任务的主要目的。三是建议助教对课程用到的工程内部代码进行优化，我们发现在游戏过程中，程序所占用的内存容量是不断增长的，我们的俄罗斯方块在进行一个多小时的游戏之后，内存占用会达到 10G 以上；同时机械臂点击动作不一致的问题也给我们的工作带来很大困扰。四是建议同一小组共同完成一份结题报告，而不是一人提交一份。因为一个小组完成的工作是一样的，当小组成员共同完成一项任务时，报告的内容都是大致相同的，大家对技术问题交流很多，重复撰写多次没有意义；当小组成员分工非常明确时，大家又对其他成员的内容不太了解，写报告也会产生不完整的情况。除了以上四点之外，我认为本课程在其余所有方面都非常优秀，在保证趣味性的同时，也让我们学到了不少知识。

课程进行的过程中我们遇到了很多困难，咨询助教时总能得到助教的及时耐心的帮助和非常暖心的鼓励，助教也为我们的一些技术细节进行了提示，实实在在地帮助我们解决了一些棘手的难题。在此我想特别向课程的老师和助教表达感谢！另外，我们还与其它小组的同学进行了深入的交流，在此过程中也得到了一些难得的启发，在此也向本课程的其他同学表达感谢！作为小组的成员之一，我们小组在整个学期中互帮互助，高效地完成了所有任务，三位成员都很努力，在此我也向我的队友陈思哲和张沛东表达诚挚谢意！