**北 京 邮 电 大 学**

**本科毕业设计（论文）开题报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 学院 | 计算机学院（国家示范性软件学院） | | 专业 | | | 计算机科学与技术 | |
| 学生姓名 | 张泽睿 | 学号 | 2022211397 | | | 班级 | 2022211312 |
| 指导教师姓名 | 谭剑 | 所在单位 | 北京邮电大学 | | | 职称 | 副教授 |
| 设计（论文）题目 | （中文）基于Unity引擎的多文明策略游戏设计与实现 | | | | | | |
| （英文） Design and Implementation of Multi-Civilization Strategy Game Based on Unity Engine | | | | | | |
| **毕业设计（论文）开题报告内容：**  **一、选题背景与意义**  本课题为《泰拉诺斯大陆：七境英雄（TH7）》本科毕业设计的“程序框架与系统整合”子课题，目标是在Unity引擎中实现一套可扩展的回合制策略/英雄养成核心框架：大地图探索—城镇建设—英雄招募与成长—回合制战斗，并能够承载7个文明（艺术资源与数值）在统一数据结构与目录规范下进行快速集成。该类型经过《英雄无敌III》长期验证，具有清晰的资源—建设—战斗—成长闭环[2]，且天然适合做“内容驱动型”扩展（新文明、新单位、新建筑、新法术）。  从产业与教学视角看，AIGC在概念生成、风格探索与批量资产生产方面显著降低成本，但只有在程序侧建立稳定的“资源与规则容器”（数据驱动、热插拔、可调试）后，美术与数值才能被有效复用与迭代[1][10][11]。因此，本课题的意义在于：用本科可控的工程规模，搭建一个可持续扩展的策略游戏骨架，为后续商业化或科研原型奠定基础。  **图1 TH7整体架构鸟瞰：数据驱动的‘大地图-城镇-战斗’三场景闭环**  **二、国内外现状与竞品评述**  竞品方面，《英雄无敌III》提供了“英雄+城镇+战棋”的经典交互范式：在高信息密度界面中同时表达资源、建筑树、兵种层级与战斗策略[2]；《文明VI》强调宏观科技与城市运营，其系统层级更复杂但战斗即时反馈较弱[3]；《奇迹时代4》在回合战斗与单位技能树方面更接近本项目的战斗设计空间[4]；《全面战争：战锤3》体现了多文明差异化与可扩展内容体系，但其制作规模远超本科项目[5]。综上，本课题采用“经典闭环+数据驱动扩展”的折中路线：用尽可能少的系统实现尽可能多的内容容器。  技术现状方面，Unity 2D生态成熟，Sprite Atlas、Addressables等可支撑中大型2D资源加载与管理[13][14]；AIGC侧，扩散模型在高分辨率概念图生成与局部可控编辑上表现突出[10][11]，配合ControlNet可实现“姿态/线稿/结构”条件控制[12]，但落地到游戏仍需在分辨率、透明通道、像素对齐、帧动画拆分、版权合规等方面做工程化约束。  **图2 竞品对比矩阵：HoMM3 / Civ6 / AoW4 / TW:WH3 的系统层级与资源规模**    **三、研究目标、研究点与关键词**  （1）总体目标：实现可运行的“最小可发布版本（MVP）”程序框架：支持至少1张大地图、1座城镇、2名英雄、每文明至少2种单位与1种法术，且能够在不改动核心代码的前提下，通过配置与资源目录新增文明内容。  （2）研究点：①数据驱动的规则表达与校验（单位/建筑/法术/英雄成长）[19]；②可扩展的战斗系统与AI行为（基于网格寻路、威胁评估、技能释放优先级）[20][21]；③资源加载与性能（Addressables、对象池、批处理、序列化）[14]；④调试可视化与可测试性（战斗回放、日志、配置检查、关卡验证）。  （3）关键词：回合制策略；数据驱动；Unity；ScriptableObject/JSON；可扩展架构；战棋网格；路径搜索；游戏AI；资源管线；可视化调试。  **图3 数据结构草案：文明/城镇/建筑/单位/法术/英雄的配置关系**    **四、关键问题与拟解决方案**  问题1：系统规模与可完成性。策略游戏系统易‘爆炸’，本科周期内必须锁定“可运行闭环”的核心：回合推进、移动与交互、建造与招募、战斗胜负、成长与资源循环。解决：采用MVP优先策略，第一阶段仅实现单城镇+单地图闭环；其余扩展通过配置迭代。  问题2：数据驱动与内容协作。7个文明并行制作需要稳定接口，否则合并成本会在后期集中爆发。解决：制定统一命名、目录、资源规格与配置字段；提供校验工具与示例模板；将“文明包”视为可加载的内容模块。  问题3：战斗AI与可解释性。仅有‘能打’不足以调参；需要可解释的决策记录与可视化。解决：分层AI（战略选择/战术执行），每一步输出理由（目标选择、技能评分、路径评分），并在调试面板可回放。  **图4 战斗AI分层示意：评估-决策-执行-记录（含可视化）**    **五、技术路线与实现方案**  1）场景与状态机：Overworld（大地图）/Town（城镇）/Battle（战斗）三场景，外层由游戏状态机统一管理，保证存档/读档与跨场景数据一致。  2）数据表示：优先使用ScriptableObject作为编辑期配置容器（可视化、可校验），运行期可导出JSON以便版本控制；关键结构（单位属性、技能、成本、成长曲线）保持字段稳定，避免后期破坏性修改。  3）资源加载：所有美术资源按文明目录组织，Sprite Atlas打包；运行期使用Addressables异步加载并缓存，城镇与战斗都只依赖‘配置+地址’而非硬引用，降低合并冲突。  4）战斗系统：采用方格战场（hex或square二选一，建议square降低复杂度），实现移动范围、攻击范围、障碍与地形修正；行为顺序采用速度+队列；技能系统采用“效果组件化”（伤害、控制、召唤、位移等）以复用。  5）工程规范：依照面向对象设计原则与常用设计模式（工厂、策略、命令、观察者等）组织代码[19]；核心逻辑与表现解耦，保证替换UI与特效不影响规则。  **图5 目录规范与模块边界：Runtime/Editor、Config/Art/Audio、Civilizations/\***    **六、AIGC协同与资产落地约束**  尽管程序组不直接产出美术，但需要为AIGC资产落地制定‘可用约束’，否则会出现：透明通道缺失、边缘抖动、分辨率不一致、帧动画不可拆、版权来源不清等问题。方案：①规定输出分辨率与像素对齐规则（例如单位立绘512×512或1024×1024，图标256×256，UI九宫格切片）；②规定命名与元数据（文明、单位ID、朝向、帧序号）；③提供一键导入工具（自动切片、自动入Atlas、自动生成Addressable标签）。AIGC工具建议以Midjourney做概念探索[18]，以本地扩散模型做可控修订与批量一致化[11][12][17]。  **图6 AIGC到Unity的落地流水线：生成-筛选-修订-切片-入库-绑定配置**    **七、研究基础与条件**  个人基础：具备C#与Unity开发经验，熟悉2D游戏对象、动画、UI系统与基础数据结构；具备阅读英文文档与调试能力。团队基础：7个文明美术由智能交互设计专业同学完成，能够提供稳定的资源与交互需求输入；项目已有统一命名与目录规范文档、文明设定与数值草案，可作为配置字段与接口的依据。平台条件：Unity 2021/2022 LTS，版本控制（Git），协作通信工具，AIGC侧可使用Midjourney/ComfyUI/Stable Diffusion等。  **八、进度计划（16周）**  第1-2周：梳理需求与系统边界，冻结配置字段与目录规范；完成最小大地图回合推进与角色移动。  第3-5周：城镇界面MVP（资源、建筑树、招募）；完成配置驱动加载与Addressables管线。  第6-9周：战斗MVP（网格、移动、攻击、胜负、战斗结算）；接入基础AI（就近攻击、威胁评估）。  第10-12周：扩展法术/技能系统、英雄成长、存档读档；完善调试面板与日志。  第13-15周：对接至少2个文明资源与数值，完成整体验收用关卡；性能与稳定性优化。  第16周：论文/文档整理、演示视频与可运行包提交。  **图7 16周甘特图（里程碑、风险点与验收标准）**  **九、预期成果与创新点**  预期成果：①Unity可运行项目（含大地图、城镇、战斗三场景闭环）；②文明内容模块化加载示例（至少2个文明可替换）；③配置校验与导入工具（Editor扩展）；④论文与技术文档（数据结构、接口、目录规范、扩展指南）。  创新点（本科层面）：①将“文明包”视为可独立交付的内容模块，降低多人并行协作的集成成本；②强调AI决策可解释性与回放工具，便于调参与教学展示；③面向AIGC资产落地的工程化约束与自动化导入，提高内容生产效率。  **图8 战斗回放与调试面板原型（可解释AI日志）**    **十九、附录：程序实现规范与编码约束**  （1）命名与目录：类名PascalCase、私有字段\_camelCase；资源按Civilizations/<CivId>/...组织；配置ID一旦发布不随意更改。  （2）解耦原则：规则层不直接依赖UI与特效；表现层通过事件/接口订阅规则变化；避免在MonoBehaviour中堆叠业务逻辑。  （3）可测试与可回滚：关键计算（伤害、命中、AI评分）写成纯函数；每场战斗生成摘要日志与随机种子，保证可复现。  （4）性能与资源：对象池复用；避免频繁Instantiate/Destroy；Addressables异步加载，防止主线程卡顿；图集与批处理策略遵循Unity文档[13][14]。  **图9 配置校验器与错误提示示例（Editor工具）**    **图10 扩展指南示意：新增文明的最小步骤与自动生成内容**    **十、参考文献**  1. [1] 谭剑.《人工智能与艺术》[M]. 北京：北京邮电大学（教材/讲义），近年版。  2. [2] New World Computing. Heroes of Might and Magic III（《魔法门之英雄无敌III》）[PC Game]. 1999.  3. [3] Firaxis Games. Sid Meier’s Civilization VI（《文明VI》）[PC/Console Game]. 2016.  4. [4] Triumph Studios. Age of Wonders 4（《奇迹时代4》）[PC/Console Game]. 2023.  5. [5] Creative Assembly. Total War: WARHAMMER III（《全面战争：战锤3》）[PC Game]. 2022.  6. [6] Schell, J. The Art of Game Design: A Book of Lenses (3rd ed.)[M]. CRC Press, 2019.  7. [7] Nielsen, J. Enhancing the explanatory power of usability heuristics (or “10 Usability Heuristics for User Interface Design”相关启发)[EB/OL]. 1994/2020.  8. [8] Norman, D. The Design of Everyday Things (Revised and Expanded)[M]. Basic Books, 2013.  9. [9] Isola, P., Zhu, J.-Y., Zhou, T., Efros, A. A. Image-to-Image Translation with Conditional Adversarial Networks[C]. CVPR, 2017.  10. [10] Ho, J., Jain, A., Abbeel, P. Denoising Diffusion Probabilistic Models[C]. NeurIPS, 2020.  11. [11] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B. High-Resolution Image Synthesis with Latent Diffusion Models[C]. CVPR, 2022.  12. [12] Zhang, L., Rao, A., Agrawala, M. Adding Conditional Control to Text-to-Image Diffusion Models (ControlNet)[EB/OL]. arXiv:2302.05543, 2023.  13. [13] Unity Technologies. Unity Manual: 2D Game Development & Sprite Atlas[EB/OL].  14. [14] Unity Technologies. Addressables System (Asset Bundles & Async Loading)[EB/OL].  15. [15] Adobe. Substance 3D Painter/Designer Documentation (PBR Materials Workflow)[EB/OL].  16. [16] Blender Foundation. Blender Manual (Modeling / UV / Rendering)[EB/OL].  17. [17] ComfyUI Community. ComfyUI Documentation (Node-based Diffusion Workflow)[EB/OL].  18. [18] Midjourney. Midjourney User Guide / Documentation[EB/OL].  19. [19] Gamma, E., Helm, R., Johnson, R., Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software[M]. Addison-Wesley, 1994.  20. [20] Millington, I., Funge, J. Artificial Intelligence for Games (2nd ed.)[M]. CRC Press, 2009.  21. [21] Buckland, M. Programming Game AI by Example[M]. Jones & Bartlett, 2005.  22. [22] Unity Technologies. DOTS / ECS Documentation[EB/OL]. | | | | | | | |
| 允许进入毕业设计（论文）下一阶段：是 □ 否 □ | | | | 指导教师签字 |  | | |
| 日期 | 2025年 12月 23日 | | |

注：可根据开题报告的长度加页