Computer Science and Application 计算机科学与应用, 2024, 14(11), 149-160
Published Online November 2024 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2024.1411224

Hans汉斯

基于Unity3D的解谜游戏设计与实现

张　瑜，王一寒，王华蕊，丁立新，陈建龙

北方工业大学信息学院，北京

收稿日期：2024年10月20日；录用日期：2024年11月18日；发布日期：2024年11月27日

摘　要

文章针对解谜游戏开发中常见的背包系统缺乏专门实现方案、对话系统内容加载不够灵活等问题，提出了一种基于Unity3D引擎的背包系统和对话系统优化方案。该方案通过改进背包管理逻辑和对话内容加载机制，旨在提升游戏开发效率和用户体验。基于此优化方案，文章设计并实现了一款解谜游戏，验证了所提出的系统框架能够有效减少开发中的逻辑混乱，缩短内容加载时间，为开发者提供了具有实际应用价值的参考。

关键词

游戏开发，软件工程，解谜游戏，系统设计

Design and Implementation of Puzzle Game Based on Unity3D

Yu Zhang, Yihan Wang, Huarui Wang, Lixin Ding, Jianlong Chen

School of Information Science and Technology, North China University of Technology, Beijing

Received: Oct. 20th, 2024; accepted: Nov. 18th, 2024; published: Nov. 27th, 2024

Abstract

Aiming at the problems such as lack of specific implementation scheme and inflexible loading of dialog system in common knapsack system in puzzle game development, this paper proposes an optimization scheme of knapsack system and dialog system based on Unity3D engine. The solution aims to improve the efficiency of game development and user experience by improving the backpack management logic and dialog content loading mechanism. Based on this optimization scheme, this paper designs and implements a puzzle game, and verifies that the proposed system framework can effectively reduce the logic chaos in development, shorten the content loading time, and provide a practical reference for developers.

文章引用: 张瑜, 王一寒, 王华蕊, 丁立新, 陈建龙. 基于 Unity3D 的解谜游戏设计与实现[J]. 计算机科学与应用, 2024, 14(11): 149-160. DOI: 10.12677/csa.2024.1411224

Keywords

Game Development, Software Engineering, Puzzle Games, System Design

1. 引言

解谜类游戏以其独特的思维挑战和逻辑推理吸引了大量玩家，成为游戏市场中一类重要的游戏类型。开发此类游戏的核心在于设计合理的游戏机制，尤其是在背包系统和对话系统的实现上，这些系统直接影响到游戏的交互性和玩家的体验。然而，现有的游戏开发方案在这些方面仍存在诸多不足。

在解谜游戏的开发过程中，背包系统用于管理玩家所获得的道具，常见的实现方式往往缺乏灵活性，容易导致代码冗杂和逻辑混乱。此外，解谜游戏的对话系统通常涉及复杂的剧情和分支选择，常用的静态文本加载方式不仅增加了开发的难度，也使得游戏内容的动态更新变得更加困难。这些问题直接影响了游戏的开发效率和用户体验。

随着 Unity3D 引擎在游戏开发中的广泛应用，开发者们获得了更多开发工具和资源。然而，Unity 引擎并未提供专门的背包系统和灵活的对话系统实现方案，开发者需要自行解决这些问题。为了简化开发流程，提升游戏的性能和灵活性，本文提出了一种基于 Unity3D 的优化框架，通过使用 ScriptableObject 优化背包系统，并设计了一个动态加载的对话系统。在对话系统的实现中，常见的方法依赖预先设定的文本框和静态加载方式，难以动态调整剧情内容，导致剧情发展缺乏灵活性和互动性，尤其是在面对多分支、多选择的复杂剧情时，现有方法显得繁琐且效率低下。

为了应对这些挑战，本文提出了一种基于 Unity3D 的背包系统和对话系统优化方案。该方法主要包含三大步骤：首先，基于 ScriptableObject 对背包系统进行优化，提供了一种高效的物品管理机制，能够灵活处理物品的添加、删除和查询操作，减少了程序的耦合性并提升了系统的扩展性。其次，本文设计了一种动态加载的对话系统，利用协程实现对话的灵活加载与管理，避免了静态文本加载方式的局限性，使得游戏剧情的变化更加自然流畅。最后，本文通过改进的系统框架，整合了背包系统与对话系统，保证了两者之间的逻辑一致性，进一步优化了游戏的整体性能和用户体验。

本文的贡献在于提出了一种针对解谜游戏开发中背包系统与对话系统的全新优化方案。通过引入 ScriptableObject 和动态加载技术，本文的方法有效地解决了现有开发方法中灵活性差、效率低的问题，并且通过实际游戏开发验证了其可行性和有效性。本文的研究为开发者提供了一个新的思路，能够有效简化解谜游戏的开发流程，提高游戏系统的可扩展性和稳定性，进而为游戏开发者提供了具有实际应用价值的参考。

2. 相关工作

2.1. 开发工具

本文所提出的框架建立在 Unity3D 游戏引擎之上。Unity3D 游戏引擎是一个多平台综合型游戏开发工具，使用 C#作为用户脚本语言，运行 Unity 脚本会按预定顺序执行事件函数，在脚本的生命周期内事件函数的执行顺序会对游戏效果产生影响[1]，且其具有多平台支持特性，其具有易上手，功能多，跨平

台支持的优点，其本身支持多达 28 个平台游戏的一次编译多平台发布的特性远远超过了同领域的开发平台，这使游戏开发过程明显更快，其需要的编程量远远少于自己独立开发[2]。

在搭建本文框架的实例游戏过程中，绘制美术素材使用的是 CLIP STUDIO PAINT，它搭载了绘制漫画、插画和动画所需的功能，拥有极其丰富的素材库，用户可使用模型辅助创作或使用库内所下载的素材。

2.2. 解谜游戏开发

解谜游戏，是与侦探小说故事类似的，以发掘线索、探索未知、解决谜题为目标，需要玩家进行细致观察、深入思考以及严密逻辑推理的一种电子游戏类型[3]。这类游戏通常结合丰富的故事情节和复杂的人物关系，使玩家能够在虚拟世界中感受到每一个线索背后隐藏的深意与可能性。由于人类本身具有想要解决问题的天性，所以解谜类游戏成为了大众喜爱的经典游戏类型之一[4]。这些游戏不仅能够提供智力上的挑战，还能激发玩家的好奇心和探索欲望，使他们沉浸在不断解锁新线索的过程中，并不断追求真相。

同时，游戏交互设计不同于以往软件、网站的交互。游戏更注重沉浸、心流、情感等体验，接近于一种来于书籍电影中的人、景、事、物的互动交融碰撞，其交互方式与交互通道更是多而深入[5]。这种独特的交互设计不仅提升了玩家的参与感与投入度，还增强了他们与游戏环境之间的情感联系，使得解谜游戏成为一种具有高度艺术性和娱乐性的体验，能够引导玩家在解谜的过程中，体验到成就感和愉悦感。

2.3. 游戏内容呈现

且近年来，计算机软硬件技术的进步赋予游戏设计师构筑更加丰富叙事内容与多元叙事模式的可能性[6]，这种技术的发展不仅提升了游戏的视觉和音效效果，还使得叙事更加复杂和多样，能够让玩家在探索中获得更深层次的体验。

电子游戏毋庸置疑地拥有着可与传统叙事媒介相媲美的叙事属性，而更自由、更开放、更具互动性也将成为未来游戏发展的终极趋势[7]，这种趋势不仅反映了玩家对于个性化体验的需求，也标志着游戏设计向更高层次的叙事艺术迈进。游戏，至少电子游戏，可以算是一种堪与文学、戏剧、影视相提并论的艺术呈现形式，游戏具有“讲故事”的能力，只是呈现方式很是新颖而已[8]，而解谜类游戏以多选择触发为特色，玩家选择不同的对话结果后触发的事件也将不同，能够使得玩家在游玩过程中获得主动角色的体验感，使得玩家不仅是故事的观众，更是参与者和塑造者，因此本文以一款解谜游戏的实现为例子以实现所提出方法的论证。

3. 方法设计

本文所提出的关键技术实现集中于对背包系统的框架设计以及对话系统的优化。通过深入分析现有方法的不足，本文提出了创新性的方法，分别在模块化、分层架构和数据管理等方面进行了改进，并通过优化用户界面设计增强了系统的交互性和用户体验。

3.1. 背包系统

3.1.1. 模块化设计

为了提高背包系统的可扩展性和灵活性，本文采用了模块化设计，将背包系统划分为独立的功能模块。具体而言，系统功能被拆分为物品管理、物品显示、数据存储等多个独立模块。通过模块化设计，开发者可以根据游戏需求轻松添加新的物品类型或功能，而无需修改现有系统。这种设计减少了系统更新或拓展时的代码冲突风险，提升了系统的可维护性。此外，模块化设计还使得系统具备了良好的复用性，

不同项目中可以直接移植或调整这些模块来满足具体需求。

3.1.2. 分层架构

本文提出的背包系统采用了三层架构，包括数据层、逻辑层和 UI 层，如图 1 所示。这种分层设计有助于更好地管理和组织系统功能，提升了系统的可维护性和数据操作的效率。

数据层

通过 PackageLocalData 和 PackageTable 实现物品数据的本地存储和管理。PackageLocalData 负责对物品信息的持久化存储，保证了玩家在不同游戏会话中的数据一致性，特别是在保存和加载游戏时，能够快速准确地恢复玩家的物品状态。这种数据持久化方案有效减少了复杂数据处理带来的错误风险，同时也为物品数据的扩展性提供了基础。

逻辑层

由 GameManager 类提供核心 API，支持物品的增删改查、排序、筛选等操作。逻辑层通过集中化管理物品操作，减少了重复代码的产生，并且为开发者提供了统一的接口，便于在游戏开发过程中快速调用背包系统的核心功能。这种设计不仅简化了开发工作，还增强了系统的可读性和扩展性。

UI 层

通过 PackageCell 和 PackageDetail 构建用户界面，PackageCell 负责每个物品的显示与交互，而 PackageDetail 用于呈现物品的详细信息和描述。UI 层设计注重玩家体验，采用响应式设计确保玩家能够流畅地查看和操作背包物品，增强了游戏的互动性和沉浸感。

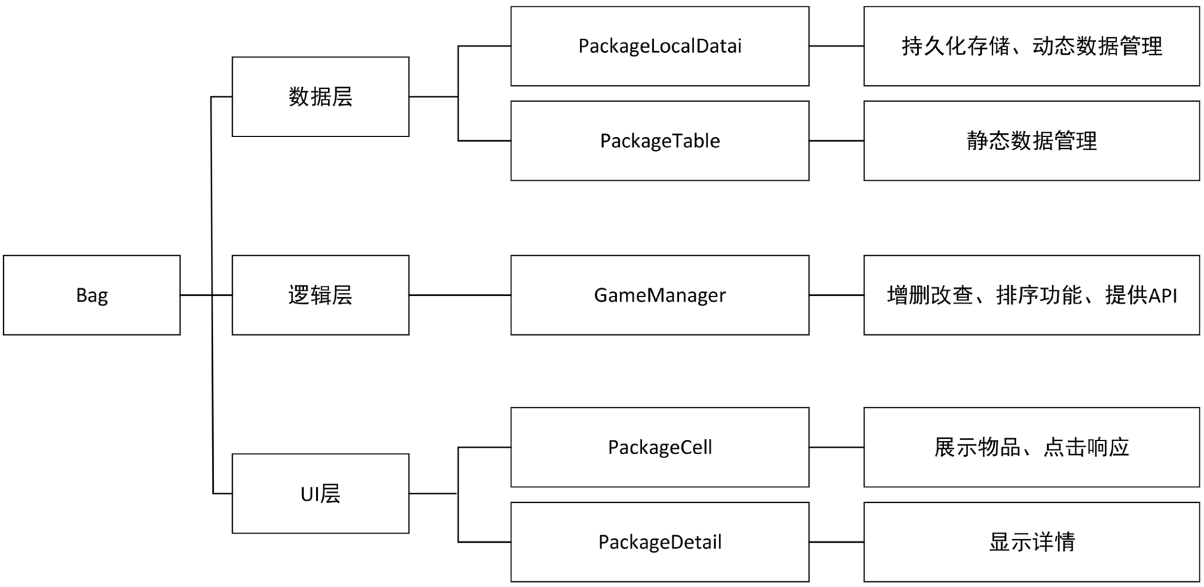

Figure 1. Backpack system framework
图 1. 背包系统框架

3.1.3. 系统优势

通过模块化和分层架构的结合，本文设计的背包系统不仅具备高效的物品管理功能，还极大地提升了系统的可维护性和可扩展性。开发者可以在不影响其他模块的前提下对特定功能进行独立更新或优化。此外，背包系统的 UI 设计通过精心布局和动态响应的用户界面，提升了玩家与物品系统的交互体验，使得物品管理更加直观易用，进而提高了游戏的整体可玩性和用户粘性。

3.2. 对话系统

对话系统在游戏中扮演着关键角色，它不仅为玩家提供沉浸式的叙事体验，还能影响游戏的进程和结果。本文提出了一种高度模块化和可扩展性强的对话系统实现方法，不仅提高了对话管理的灵活性，还增强了玩家的参与感和互动性。

3.2.1. 分层架构

本文提出的对话系统采用了三层架构设计，如图 2 所示，包含数据层、逻辑层与 UI 层。通过这种分层架构，系统不仅具备良好的可维护性和可扩展性，还能为玩家提供流畅而沉浸的对话体验，进一步增强游戏的趣味性和深度。

数据层

采用 ScriptableObject 管理数据，并通过 DialogDataListSO 和 DialogData 类实现了对话数据的高效存储和管理。DialogDataListSO 类作为数据容器，负责存储多个对话数据对象的列表。它不仅支持对话数据的批量操作，还提供了对话数据的快速检索和组织功能，确保游戏在运行时能够高效访问所需的对话信息。而 DialogData 类则负责定义单个对话的结构，包括对话文本、角色信息、选项内容及对应的后续情节。通过封装这些数据，DialogData 类使得对话内容更加模块化，便于扩展和维护。同时，该类还提供方法以便于对话内容的动态修改与更新，确保游戏的灵活性和互动性。

逻辑层

使用 DialogManager 类负责整个对话系统的流程控制、状态管理和核心功能实现，主要处理对话的进度控制、选项逻辑、打字机效果等关键功能，逻辑层通过解析并执行相应的逻辑，确保玩家体验的连贯性，根据玩家的选择动态调整对话流向，并在必要时触发相关事件或状态变化。并简化了对话系统的设计，减少了冗余代码的产生，还为开发者提供了统一的接口，便于在游戏开发过程中快速调用对话系统的核心功能，提高了系统的可读性和可维护性。

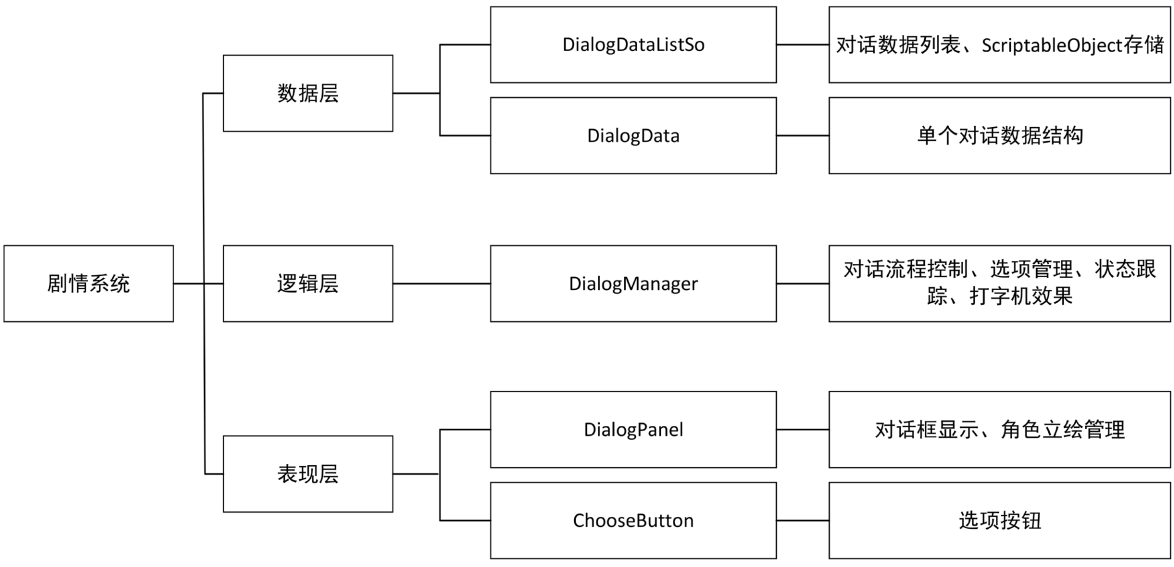

Figure 2. Dialog system framework
图 2. 对话系统框架

表现层

由 DialogPanel 和 ChooseButton 等 UI 组件组成，负责对话内容的可视化呈现和用户交互。DialogPanel

负责管理对话框的显示，确保对话文本清晰可读，并且能够灵活切换角色立绘，以增强角色的表现力和玩家的沉浸感。ChooseButton 组件则专注于处理分支对话中的选项展示及用户选择。通过将这些 UI 组件有效结合，不仅提升了对话的可视化效果，还增强了用户的互动性，为玩家提供了更为丰富和沉浸的游戏体验。整体设计理念注重玩家的需求，致力于创造流畅且引人入胜的交互过程。

3.2.2. 实现意义

分层架构的实现方式确保了对话系统具有清晰的结构和完善的功能，具有良好的可扩展性和维护性。这种设计为游戏中的叙事和角色互动提供了强大而灵活的技术支持，能够有效地处理复杂的对话结构及丰富的剧情展开。通过模块化的设计，开发者能够独立地更新和优化特定功能，而不会影响到系统的其他部分，从而提升了系统的整体效率和可操作性。此外，该对话系统的架构能够适应多变的游戏需求，支持多条对话线索和分支选择，进一步增强了玩家的沉浸感和参与度。

4. 游戏实现

4.1. 游戏简介与玩法

《古数绘缘》是一款以中国古代数学知识为核心主题的古风横版解谜类角色扮演游戏。游戏旨在通过寓教于乐的方式，帮助玩家体验数学之美，并学习基本的数学知识。玩家在游戏中扮演不自信的女主人公小茗，在好友小芸的协助下，通过一系列数学大赛的挑战，逐步提升自信，最终让更多人了解数学的魅力。游戏中的美术风格独具古典韵味，结合了中国传统文化元素，如图 3 所示。

游戏的核心玩法围绕数学解谜展开。每个关卡中，玩家将面对一道或多道数学题目，需要根据题目要求进行计算或逻辑推理，并得出正确答案。题目涵盖了多种数学领域，包括算术、代数、几何、组合数学等，随着关卡的推进，题目难度逐渐增加，要求玩家掌握更高阶的数学知识。游戏在数学解谜的基础上，巧妙地融合了场景互动和物品收集等要素，玩家可以通过与环境的互动获得相应的数学提示或道具，帮助他们解决更复杂的谜题。

Figure 3. Game start screen
图 3. 游戏开始界面

4.2. 游戏功能模块设计

游戏的功能模块设计遵循高内聚、低耦合的原则，确保各个系统独立开发和维护。图 4 展示了游戏的整体功能架构。以下是主要功能模块的详细描述：

(1) 场景管理：场景管理模块负责管理游戏中不同场景的切换和状态维护。它包括场景的加载、存储、切换逻辑，以及游戏状态的处理(如暂停、进行中和结束)。通过该模块，游戏能够在玩家完成特定任务后动态切换场景，同时保持玩家当前的游戏进度和位置。

(2) 角色控制：角色控制模块负责处理玩家输入，包括角色的移动、跳跃和与场景的互动。此模块采用基于物理引擎的控制方式，确保角色的运动与游戏物理规则保持一致。通过对输入的实时监控和反馈，该模块使玩家能够灵活地操控游戏角色，提升互动的沉浸感。

(3) 背包系统：背包系统用于管理玩家在游戏中获得的各种道具，玩家可以查看道具的详细信息并使用道具来解锁新场景或触发隐藏任务。背包系统采用模块化设计，便于扩展和维护，支持道具的存储、删除、查询和持久化管理。

(4) 对话系统：对话系统管理游戏中的剧情对话，包括文本的动态显示、选项选择和剧情分支。通过该系统，玩家能够与游戏中的角色进行互动，并根据选择影响剧情的发展。对话系统支持多分支、多结局的复杂结构，为玩家提供丰富的互动体验。

(5) 关卡设计：关卡设计模块负责设计和管理游戏中的不同关卡，并保存与关卡相关的数据。每个关卡都包含特定的数学谜题和奖励，玩家需要通过解谜来完成关卡任务。该模块支持关卡难度的动态调整以及玩家进度的追踪和存储。

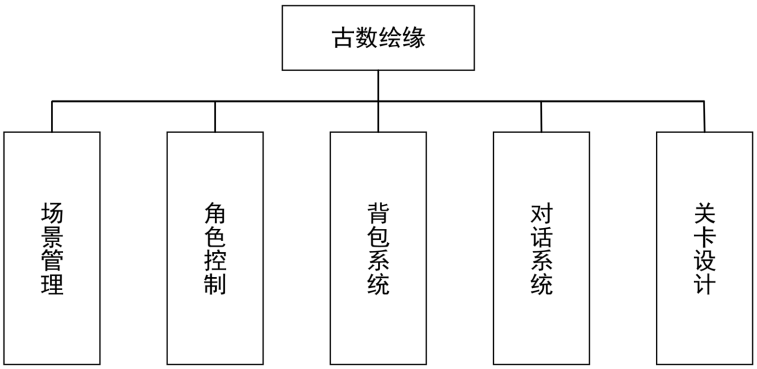

Figure 4. Overall game framework
图 4. 游戏整体框架

4.3. 游戏流程

游戏的整体流程如图 5 所示，遵循典型的解谜类游戏结构。玩家首先进入游戏主界面，可以选择“开始游戏”、“设置”或“退出游戏”选项。选择“开始游戏”后，游戏将自动进入剧情介绍阶段，玩家通过剧情了解游戏的背景故事，并逐步熟悉操作。在游戏进行中，玩家控制角色在场景中移动，并通过与场景物体的互动获取道具或解锁新关卡。

游戏中共设有三处包含解谜类小游戏的场景，玩家需要逐一与这些场景进行互动，或完成其中的解谜小游戏，以获得必要的道具。在破解了这三处场景中的谜题之后，游戏将触发结束动画并退出游戏。有关游戏的详细流程请参见图 5。

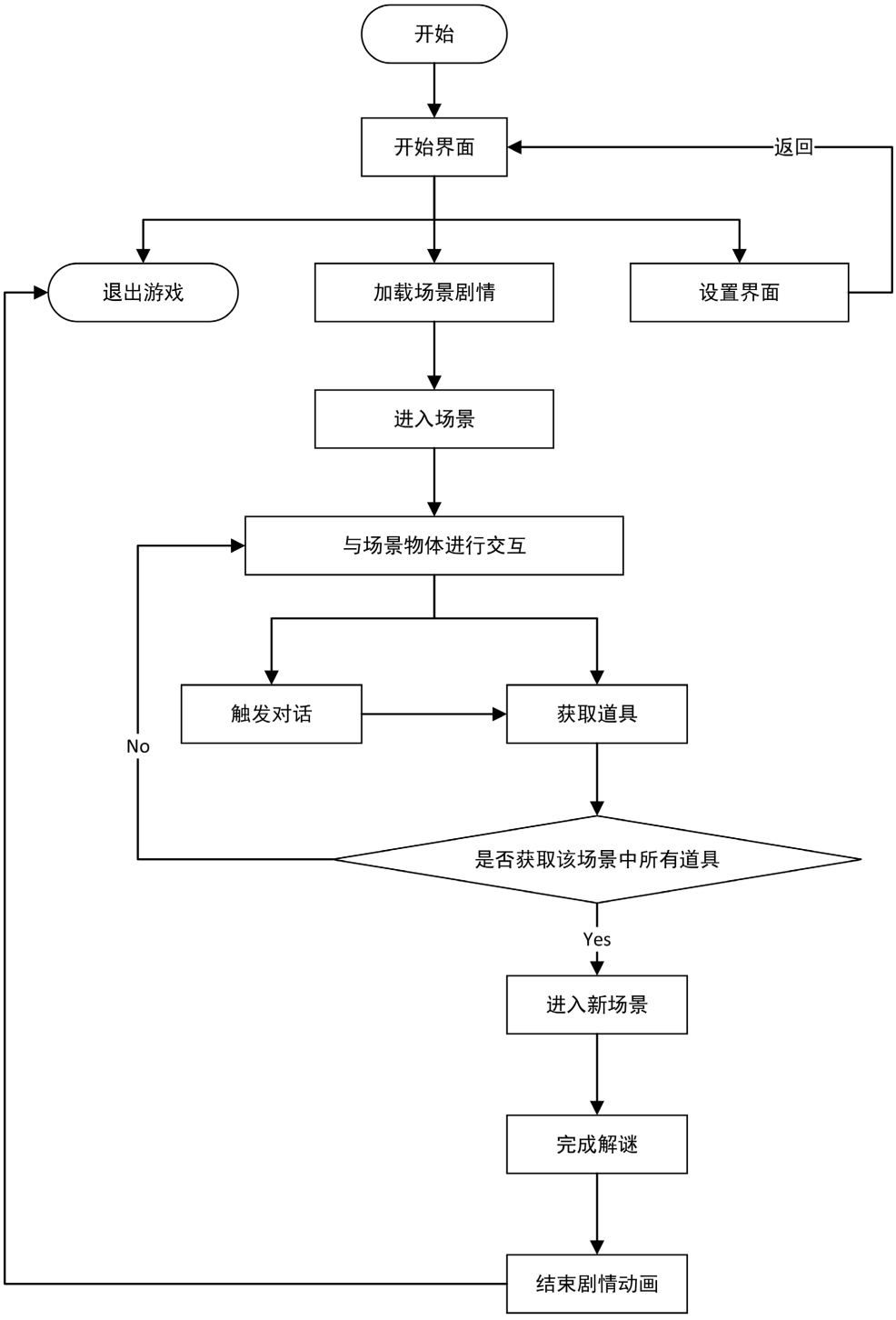

Figure 5. Game flow chart
图 5. 游戏流程图

4.4. 游戏实现

4.4.1. 场景管理的实现

在游戏开发过程中，场景管理是至关重要的一环，它确保了玩家在不同游戏关卡中的顺利过渡。在

《古数绘缘》中，场景切换的实现主要依赖于对玩家行为的实时监控。我们通过在场景中放置传送点，当玩家接近这些传送点时，系统会检测到玩家的输入行为。一旦玩家按下触发键，系统立即调用场景切换功能，将玩家传送到新场景。在实现过程中，我们使用了 Unity 的 SceneManager.LoadScene ()函数来加载新的场景，确保玩家在不同场景间的过渡自然流畅。同时，摄像机的位置也会根据新场景的布局进行调整，确保玩家角色始终位于屏幕的中心位置，保证视角的舒适性和一致性。

除此之外，游戏的状态管理也是通过全局控制系统实现的。如图 6 所示，游戏状态分为“进行中”、“暂停”、“结束”三种主要模式，这些状态由一个专门的枚举类进行定义和管理。当玩家需要暂停游戏或进入设置界面时，系统会通过状态机判断当前的游戏状态，并做出相应的操作，比如暂停动画、显示菜单等操作。这一设计的灵活性使得玩家能够在游戏过程中随时调整状态，且不会破坏游戏的整体流程。动态 UI 面板的切换通过 CanvasGroup 组件来实现，该组件可以让 UI 元素在不同的游戏状态下动态显示或隐藏。

Figure 6. Setting interface
图 6. 设置界面

4.4.2. 角色控制的实现

角色控制是游戏中玩家与世界进行互动的基础。《古数绘缘》中的角色控制系统依赖于物理引擎的实现，确保玩家的操作能够与游戏中的物理规则相匹配。在实现过程中，我们使用了 Rigidbody2D 组件来为角色提供物理属性。玩家的每一次移动都是通过读取输入值来实现的：当玩家按下方向键，游戏通过 Input. GetAxis (“Horizontal”)来获取输入的水平值，然后根据这个值调整角色的移动速度。这个速度值会被应用到 rigidbody. velocity 上，进而驱动角色在场景中左右移动。

如图 7 所示，角色的方向变化也与玩家的输入紧密相关。当玩家改变移动方向时，系统会通过调整角色的局部缩放(transform. localScale)来翻转角色，使其面向正确的方向。此外，角色的动画效果是通过动画状态机(Animator)来控制的。状态机中的“速度”参数会根据角色的运动状态动态更新：如果角色的水平速度大于零，游戏将播放跑步动画；当角色静止时，动画会切换为待机状态。整个过程流畅且自然，确保了角色行为与玩家操作之间的同步。

玩家与环境的交互则通过物体之间的碰撞检测实现。我们使用了 OnTriggerEnter2D 等方法来检测角色是否接触到了可交互对象(例如 NPC 或道具)。当角色进入这些交互区域时，系统会在界面上提示玩家按下特定键来触发交互。这种交互方式使得玩家能够直观地感受到与游戏世界的联系，进一步增强了游戏的沉浸感。

Figure 7. Characters and scenes interact
图 7. 人物与场景互动

4.4.3. 背包系统的实现

图 8 展示了游戏中的背包系统。背包系统在《古数绘缘》中扮演着管理和展示玩家所获得道具的重要角色。为了确保系统的可扩展性和易用性，我们采用了 Scriptable Object 来管理道具数据。每个道具被定义为一个独立的对象，包含道具的名称、描述、图标以及其他相关信息，这样的设计可以使道具数据独立于游戏场景，便于存储和调用。

Figure 8. Backpack system
图 8. 背包系统

在实际运行过程中，背包界面通过网格布局(GridLayoutGroup)进行展示，每一个道具格子都与一个数据对象绑定。当玩家点击某个道具时，系统会捕获该点击事件，并通过更新背包的详细信息面板来显示道具的具体内容(如名称、描述和用途)。这种交互方式不仅提供了清晰直观的物品管理体验，还确保了玩家能够随时掌握自己拥有的道具信息。

为了让背包数据在玩家多次进入游戏时保持一致，我们实现了数据的持久化存储。当玩家获得新道具或使用道具时，背包数据会自动更新，并通过 Json Utility 将道具数据序列化为 JSON 格式，存储在本地文件中。在下一次启动游戏时，系统会通过反序列化过程恢复之前的道具数据，确保玩家可以继续之前的游戏进度。

4.4.4. 对话系统的实现

对话系统是游戏中剧情呈现和角色互动的核心部分。为了实现高效的对话管理，我们通过ScriptableObject 存储了所有对话内容。每段对话都被分配了一个唯一的 ID，游戏可以通过这个 ID 快速查找并加载相应的对话条目。在游戏启动时，系统会提前构建一个对话字典，确保在玩家触发对话时能够立即加载相应内容，从而提高对话系统的响应速度。

对话的展示通过打字机效果来增强互动体验。具体实现方式是利用协程(Coroutine)逐字显示对话文本。协程会根据设定的时间间隔逐一显示文本的每个字符，营造出字符逐步显现的效果。这一效果不仅提升了玩家的沉浸感，还增加了游戏的节奏感。此外，对话面板还支持角色的动态立绘展示。根据对话情节的发展，角色的表情和立绘会随着剧情变化自动切换，给玩家带来更加生动的视觉体验。图 9 展示了人物对话时的文本效果。

Figure 9. Character dialogue
图 9. 人物对话

对话系统还实现了多分支对话和选项选择功能。每当玩家进入某段具有分支选择的对话时，系统会在面板上动态生成选项按钮。玩家点击不同的选项后，系统会根据选择更新对话内容或引导剧情进入不同的方向。这一设计不仅增强了玩家的参与感，还提供了更多样化的剧情发展可能性。

4.4.5. 关卡设计的实现

关卡设计是解谜类游戏的核心之一，《古数绘缘》通过对中国古代数学知识的巧妙运用，设计了一系列富有教育意义的关卡。每个关卡中的数学题目都来源于历史悠久的数学典籍，如杨辉三角、幻方、算盘、七巧板和华容道。为了将这些古代数学知识与现代游戏设计相结合，我们对每个题目进行了游戏化的改造，确保玩家能够通过解谜的方式学习和理解这些知识。

Figure 10. Game effect picture
图 10. 小游戏效果图

在算盘关卡中，玩家需要通过拖动珠子来完成计算。每颗珠子的状态会影响总金额的计算，系统通过实时更新珠子的位置信息来计算当前的总金额，并在屏幕上显示最终结果。华容道关卡则通过物理引擎实现拖动块的移动，玩家需要将特定的块移到目标位置，当所有块移动到正确的位置后，系统会触发关卡完成的逻辑，进入下一个挑战。七巧板关卡的实现依赖于物体的拖放机制，玩家需要将几何图形拖动到指定区域，当物体成功放置时，系统会自动更新当前的拼图进度，直至玩家完成整个拼图。效果图见图 10。

5. 总结

本文针对解谜类游戏中关键技术，包括对于如何在 Unity3D 中更好的制作背包系统给出了实现方案，并在游戏中进行了验证，验证了剧情系统的实现方法，通过运用在解谜类游戏开发中，证明了技术方案的可行性。课题中提供的设计和实现背包系统与剧情系统的方法具有创新性，为游戏中的物品管理提供了一个高效而灵活的解决方案，并通过动态加载剧情方法，给出的剧情读取解决方案。课题没有对在背包系统中添加搜索功能、物品分类以及多语言支持、可视化调试等进行方案的验证，在下一步的工作中，将对该项内容进行探索，优化技术方案。

参考文献

[1] 杨淮敏, 邱树伟. 基于 Unity3D 的冒险游戏的设计与实现[J]. 现代计算机, 2023, 29(13): 73-78.

[2] 吴悠, 陈亦欣, 胡曦. 基于 Unity 平台的多操作系统共同开发技术研究及实现[J]. 计算机应用研究, 2020, 37(S1): 237-239.

[3] 李玮, 高玉涛. 双重情节、场景嵌入、内在探索: 解谜游戏的互动叙事模式[J]. 新闻知识, 2023(5): 24-32+93.

[4] 王一博. VR 解谜游戏的交互设计研究[D]: [硕士学位论文]. 沈阳: 鲁迅美术学院, 2021.

[5] 黄志辉. 基于心流理论的游戏交互研究[D]: [硕士学位论文]. 广州: 华南理工大学, 2017.

[6] 徐杰. 空间叙事学视域下的解谜冒险类电子游戏设计研究[D]: [硕士学位论文]. 无锡: 江南大学, 2022.

[7] 闫郡虎. 电子游戏的叙事模式研究[D]: [硕士学位论文]. 重庆: 重庆大学, 2014.

[8] 李璐. 电子游戏的叙事美学研究[D]: [硕士学位论文]. 重庆: 西南大学, 2017.