# Group 4 Report

## Temple Switch

Jenna, Tyler, Lukas and Yuxin

# Contents

# Introduction

This game is set in the remains of a mysterious ancient temple, where players will embark on a journey full of puzzles and exploration. The core mechanic of the game revolves around the real-time switching of 2D and 3D perspectives. Players must alternate between these dimensions to solve puzzles and spatial challenges.

In 2D mode, players experience the classic 2D platformer gameplay with rooms in sequential order. When switching to 3D mode, the real structure of the space, hidden space, and three-dimensional structures that were originally unnoticeable in the 2D perspective is revealed. Players need to carefully observe and analyze the differences between the two dimensions, understand the scene layout through dimensional switching, and reveal the solution to the puzzle.

The game draws inspiration from the ideas of visual illusion and dimensional dislocation, as seen in Super Paper Mario. The 2D perspective often presents a limited view of the space, while the 3D perspective reveals its full complexity. This contrast between visual perception and spatial reality forms the core of the gameplay, offering players a novel and engaging exploration experience.

By combining the mechanics of both 2D and 3D perspectives, the game encourages players to think critically, shift viewpoints, and reconstruct their understanding of space. The temple becomes a maze built around perspective, and players can only pass it through trial and error and observation.

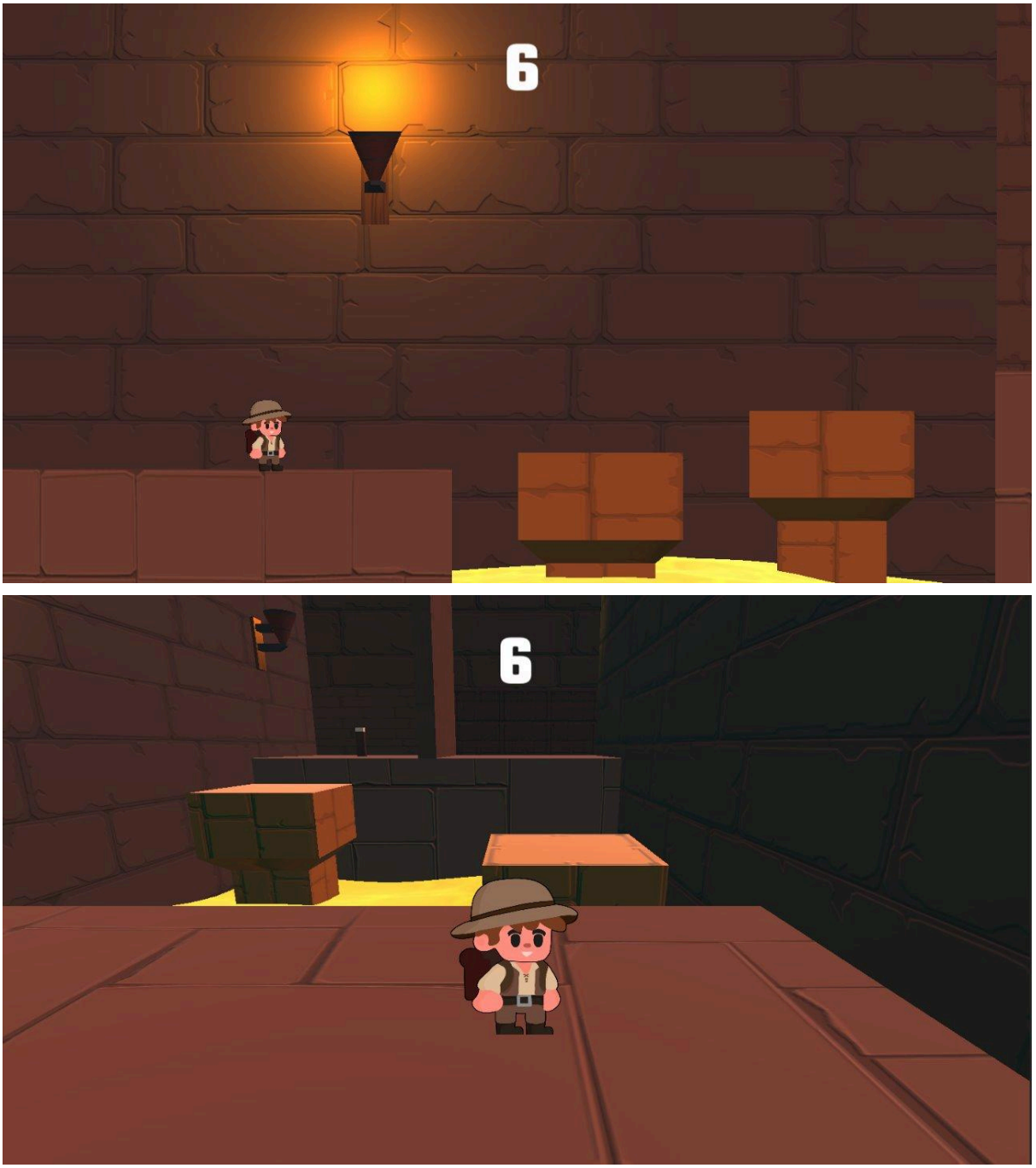
# Key Features

This project builds an engaging temple-style puzzle exploration game experience by integrating multidimensional perspective switching, immersive audio-visual design, and visual illusion mechanisms. The following are the main features of this digital experience:

## Real-time 2D-3D perspective switching mechanism

The core innovation of this game is that players can switch between 2D and 3D perspectives in real-time by pressing the "C" key. In 2D perspective, players can use "A" and "D" to move horizontally, and experience the linear level design similar to traditional platform jumping games; in 3D perspective, "W", "S", "A" and "D" are used to explore the space in all directions, and the space bar can control jumping, to discover the hidden information in the 2D perspective in the unknown space.

This mechanism is fully utilized at specific levels. For example, in Room 5, players can only see a wall with different textures in 2D mode, but when switching to 3D perspective, they can find a mechanism behind the wall that opens the next door. This type of design effectively utilizes the cognitive error caused by "perspective difference" and enhances the fun of solving puzzles.




## Immersive audio-visual presentation

The project focused on designing and optimizing the integration of visual and auditory elements to create a more immersive ancient temple atmosphere. In terms of visuals, we used textures that match the temple, with reddish-brown brick walls and gray slate floors as the main tones and added flowing lava effects to some levels. At the same time, the overall lighting of the level is set to be dim, combined with the Point Light lighting system to simulate the effects of torches and searchlights, making the overall light and shadow environment closer to the atmosphere of ruins exploration.

The game’s sound design is divided into three categories: background music, ambient sounds, and interactive effects. The background music features atmospheric tracks designed to enhance immersion and support seamless scene transitions. Additionally, responsive sound effects are triggered at key moments—such as activating mechanisms, clicking buttons, or when lava approaches—to improve player feedback and deepen the sense of interaction.

# Context and Literature Review

In the ideation phase of the project, we had found an article which our digital experience responds to. The article talks about the fall in popularity of 3D games, and the rise in 2D games (Padilla-Rodriguez, 2024). This article gave us the idea to pursue a new and unique game mechanic, combining the best of both worlds to achieve a fun and engaging player experience, one that fans of both types of games can enjoy. In recent years, immersive interactive experiences have become an important trend in independent game design. Especially in puzzle and platform jumping games, enhancing the fun and challenge of the game through perspective changes and spatial dislocation design has become an innovative means. For example, the game Fez (Polytron Corporation, 2013) introduced 2D and 3D perspective switching, constructing a non-linear puzzle experience in plane and three-dimensional space.

## Comparative analysis of related systems

Among existing perspective-switching games, Fez and Super Paper Mario (Nintendo, 2007) are representative works. Fez breaks the spatial limitations of traditional platform jumping games through the rotation mechanism of 2D and 3D perspectives and builds a multi-level, non-linear puzzle experience. Super Paper Mario also focuses on perspective switching, but the environment in its 3D perspective still maintains the 2D "paper" style, focusing on the new path identification and space utilization brought by the perspective. In contrast, this project, based on realizing 2D-3D switching, further emphasizes the authenticity and immersion of the spatial structure under the 3D perspective. It not only adopts three-dimensional modeling and a low-poly style but also renders a real world with the atmosphere of an ancient temple through lighting and textures. Spelunky (Mossmouth, 2013), is also a source of inspiration for the aesthetic of the game. In our experience, the player plays an explorer attracted by mysterious ruins. He needs to gradually cross multiple interconnected rooms, solve puzzles through perspective changes and mechanisms, and finally escape from the temple. This structural design that combines puzzle, adventure, and narrative guidance allows our project to inherit the classic mechanism while also presenting a more complete immersive experience and sense of space exploration.

## Project positioning and differentiation

The uniqueness of this project lies in the deep integration of "2D-3D perspective switching" and "mechanism puzzle solving mechanism" to build an interactive experience based on visual illusion and dimension switching. Compared with existing platform jumping games, we not only introduce three-dimensional puzzle solving in the level structure but also create a sense of temple exploration atmosphere in the art style and lighting system, making the game more challenging and immersive.

# Technical Implementation

The digital experience uses the Unity game engine to deliver an interactive experience. Using the respective features in Unity and the C# programming language, a number of game components needed to be implemented to fulfill the requirements. These include:

- Camera switching
- Player movement and management
- Triggers and Colliders
- User Interface

## Camera Switching

Camera switching is the core mechanic of the game, and therefore it was the most important component to implement. The game allows the player to switch between perspective and orthographic cameras by pressing the ‘C’ button. This allows the player to navigate through the experience through solving puzzles and completing jumps.

The Unity project consists of 2 different cameras. One is a 2D orthographic camera that is built with the Unity framework, Cinemachine. There is also a 3D perspective camera that is attached to the player. The 3D camera is a simple Unity camera which will follow behind the player in 3D mode. The 2D camera is a little more complicated. Each room in the Unity scene contains a disabled virtual camera. This virtual camera allows the 2D camera to effectively switch focus between each room. Each room also consists of a 2D polygon collider, which acts as the virtual camera’s bounding shape. This stops the camera from following the player through rooms, and instead allows the camera to switch focus between rooms. The script below facilitates this ability. When the player collides with the room’s collider, it enables the virtual camera. When the player leaves the collider, then the virtual camera is disabled, allowing the 2D camera to switch.

```
1 reference
public PlayerManager playerManager;
2 references
public GameObject virtualCam;

0 references
void OnTriggerEnter2D(Collider2D other)
{
    if(other.CompareTag("Player")) {
        virtualCam.SetActive(true);
    }

    playerManager.UpdateRoomsList(gameObject.name);
}

0 references
void OnTriggerExit2D(Collider2D other)
{
    if(other.CompareTag("Player")) {
        virtualCam.SetActive(false);
    }
}
```

To enable the player to switch between 2D and 3D cameras, the following script is used. Unity's 'Update' function listens for when the player hits the C key. When this is pressed, the opposite camera is enabled and the current camera is disabled.

```
0 references
void Update()
{
    if (Input.GetKeyDown(KeyCode.C))
    {
        cam3D.enabled = !cam3D.enabled;
        cam2D.enabled = !cam2D.enabled;
    }
```

## Player Movement and Management

As equally important as the camera switching, player movement and management is critical to the digital experience. The player will need to move and jump through the game to solve puzzles and complete platforming challenges. Due to the uniqueness of the camera switching mechanic, there are certain aspects of the player character that need to be changed to facilitate this.

Firstly, the player game object consists of a 2D and 3D collider, along with 2D and 3D mesh renderers. This allows the player object to appear correctly according to the camera projection. The player contains a Player Manager script, which handles the movement and management of the player. In Unity's 'FixedUpdate' function, which is called 0.02 seconds, is where the player movement is handled. As seen in the script below, the player character is treated differently depending on the camera that is currently enabled. When the camera is in

2D mode, the player is scaled to 100 on the Z axis. This forces the player to collide with obstacles despite their position on the Z axis. In 3D mode, the player is scaled normally. This feeds into the gameplay, allowing the player to solve puzzles, such as reaching a button behind a wall in 3D mode, or making jumps in 2D that were impossible in 3D. In 2D mode, the player can only move on the X axis using the ‘A’ and ‘D’ keys. In 3D mode, the player can move on the X and Z axis, using the ‘WASD’ keys

.

```
0 references
public void FixedUpdate()
{
    if (!cameraManager.is3D)
    {
        t.localScale = new Vector3(1.0f, 1.0f, 100.0f);

        mr.enabled = false;

        Vector3 input = new Vector3(Input.GetAxis("Horizontal"), 0, 0);
        rb.MovePosition(t.position + input * Time.fixedDeltaTime * speed);
    }
    if (cameraManager.is3D)
    {
        t.localScale = new Vector3(1.0f, 1.0f, 1.0f);

        mr.enabled = true;

        if (!hasCollided)
        {
            Vector3 input = new Vector3(Input.GetAxis("Vertical"), 0, -Input.GetAxis("Horizontal"));
            rb.MovePosition(t.position + input * Time.fixedDeltaTime * speed);

        }
```

Along with movement, the player can also jump. To stop the player from infinitely jumping in the air, some sort of check is required to ensure the player is touching the ground. Usually, this can be done with the use of a raycast shooting below the player to check if there is a collider underneath. However, due to the constrictions around the switching mechanic, this was not possible, or at least would not work in the 2D mode due to the player scaling. Instead, using a custom function called ‘isGrounded,’ checks whether the player has any velocity in the Y axis. If it does not, the player can jump. This is used in the ‘FixedUpdate’ function, where a force is added to the player's rigidbody when the player presses the ‘Space’ key.

```
1 reference
bool isGrounded()
{
    return rb.velocity.y == 0;
}
```

```
if (Input.GetKey(KeyCode.Space) && isGrounded())
{
    rb.AddForce(new Vector3(0.0f, jumpSpeed, 0.0f), ForceMode.Impulse);
    audioSource.clip = jumpSound;
    audioSource.Play();
}
```

The player also has the ability to reset. This lets the player return back to the start of the level, just in case. A respawn point game object is positioned at the beginning of each level which is used as a reference for the position the player respawns at. In the ‘PlayerManager’ script, a list is used that contains all of the names of the rooms which have been entered by

the player. The last room in this list, being the current room entered by the player, is then used to find this respawn position. The player's transform position is then made equal to the respawn point. Using the following sequence of functions, this is possible. The ‘RespawnPlayer’ function is called when the ‘R’ key is pressed.

```
1 reference
public void UpdateRoomsList(string roomName)
{
    if (!roomList.Contains(roomName))
    {
        roomList.Add(roomName);
    }
}

5 references
public string lastRoomName()
{
    return roomList[roomList.Count - 1];
}

2 references
public void RespawnPlayer()
{
    GameObject respawnPos = GameObject.Find(lastRoomName() + "/Respawn Position");

    t.position = respawnPos.transform.position;
}
```

## Triggers and Colliders

Triggers and colliders play a fundamental role throughout the digital experience, and are positioned throughout the Unity scene. As already mentioned, each room is equipped with a 2D polygon collider, which allows the 2D camera to switch focus between each room. In addition to this, buttons are a key part of the level design, and are positioned throughout much of the game. These buttons are usually used to open doors and make platforms appear. Each button is equipped with a script that uses the ‘OnTriggerStay’ function to run code when the player is inside. When the ‘E’ key is pressed, a door will open, platforms will appear or a switch will occur.

Triggers are also used for respawning. Using the functions related to resetting as mentioned in the player manager section, they can be reused for the purpose of respawning. This is for when the player character dies, usually when they touch lava, or in the unlikely scenario that the player is out of bounds. Using the ‘OnTriggerEnter’ function allows the player to call the ‘RespawnPlayer’ function from the ‘PlayerManager’ script, allowing them to respawn.

## User Interface

The last major component is the user interface. The user interface consists of many elements that aid with the user experience. When the game first loads, the main menu appears. This menu shows the player three buttons: Play, Controls and Quit. The play button will begin the game, the controls button will show the player the buttons that control the player, and the quit button will exit the game. In game, the player can press the ‘Escape’ key to bring up the pause menu. The game will be paused, and they will be able to view the control menu or exit the game. In addition to these menus, there are text tutorials that are

displayed at the beginning of the game. This text shows the player the controls in-game. Using the list from earlier that allows the player to respawn, this will be used to check the room the player is currently in, and display the according text.

## Development Tools and Technologies

In addition to these core mechanics, a number of technologies and tools were used to facilitate the technical implementation and enable collaboration between group members. These include:

- Unity
- GitHub
- Discord
- Miro
- Google Drive

The Unity game engine was the platform for development, and was ideal for every group member due to past experience with this platform. Unity provided the required functionality to deliver a game which met the requirements that we initially set out. GitHub was also used to facilitate collaborative work on the Unity project, allowing all group members to simultaneously work and contribute to the project. Discord was used as the main communication platform, allowing group members to communicate the progress made, facilitate meetings, divide workloads and discuss ideas. Miro was used in the early stages of the project, which allowed for a visual collaborative document for group members to come up with ideas. Lastly, Google Drive was used to share assets and documents, allowing for collaboration between group members.

# Development Process

The development cycle of this project is gradually promoted in multiple stages from conception, and prototype development to final integration and testing, reflecting a development strategy that emphasizes both flexibility and practicality.

## Initial conception and team building

In the early stage of the project, team members discussed the game's initial concept and technical feasibility. By sorting out the sources of inspiration, the team established the game concept with "2D and 3D switching mechanism" as the core, and initially selected pixels and low-poly style as the visual tone. At the same time, the team clarified the roles and responsibilities of the members and established collaborative platforms such as Miro, Google Drive, and Discord to support subsequent development work. An agile based development life cycle was chosen to ensure

## Core Mechanics and Story Design

During several meetings in March, the team refined the core gameplay - centered on the 2D and 3D perspective switching mechanism, and designed preliminary levels and narrative

backgrounds around this mechanism. Team norms were also established at this stage, including codes of conduct and task-tracking mechanisms. At the same time, the game structure, characters, and worldview were initially developed and finalized.

## Technical prototype and direction adjustment

The team began implementing basic functional prototypes, including perspective switching, camera control, and basic scene conversion, and further refined the game theme. The game scene was transformed from the original cave to the Aztec temple style to better support puzzle solving and modular space design. The project entered the stage of resource sorting and division of labor, and graphic assets, level sketches, and program frameworks were developed.

## Content expansion and system integration

At this stage, the focus shifted to content expansion and functional improvement. The team integrated game levels, art resources, and sound resources, and began to create menu interfaces, main game interfaces, and multiple complete levels to make the game demonstrable. At the same time, testing was carried out to evaluate the performance of game mechanisms in actual play, identify problems, and optimize the experience.

## Improvement and Optimization

In the final stage, the focus was on overall optimization and documentation. The team further improved the playability of the game, refined sound effect details, improved the main menu interface, and tested all core mechanisms. Meanwhile, the technical report and demonstration materials were finalised.

# Reflection

Overall, our game met all minimum requirements as well as some additional requirements which were added to further supplement the player's experience in the game: adding more story elements and polishing the design aspect with an array of new thematic textures and models. The theme of the game stayed consistent throughout, which makes it aesthetically pleasing to our target audience of children and young adults. The 2D/3D core mechanic is easy to grasp and works as intended, with a variety of levels of increasing difficulty and merging puzzles. Known bugs were fixed and rigorous testing showed that none were present at the time of completion, making the game run smoothly as intended. If we had more time, we would improve the variety of levels and add dialogue sections in the game to enhance the story. Adding collectable relics and an inventory would be a next step for us in the future to enhance player experience, and giving these relics in-game descriptions could further strengthen the plot and allow the player to discover more about the temple as a reward for reaching certain room milestones. Different types of rooms such as treasure rooms could be added to give the user a change of pace.

# Bibliography

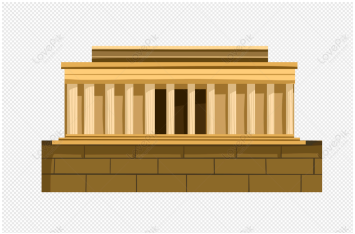
Padilla-Rodriguez, M. (2025). *3D Games Were Seen as the Future, Now 2D Games are Making a Comeback.* [Online]. How-To Geek. Last Updated: 23 November 2024. Available at: https://www.howtogeek.com/3d-games-were-seen-as-the-future-now-2d-games-are-making-a-comeback/ [Accessed 27 May 2025].

Polytron Corporation. (2013). *FEZ.* [Game]. Available at: https://store.steampowered.com/app/224760/FEZ/ [Accessed 27 May 2025].

Nintendo. (2007). *Super Paper Mario*. [Game].

Mossmouth. (2013). *Spelunky*. [Game]. Available at: https://www.spelunkyworld.com/ [Accessed 27 May 2025].

# Copyright Appendix

| Description of Asset | Source | Location | License / Permission |
|---|---|---|---|
|  | https://lovepik.com/image-40141815 5/paris-temple-of-the-nong.html | Start video | This image has a copyright license and is available for personal use. |
|   | Used Lemon Milk font: https://www.dafont.com/lemon-milk.font | Menu Logo | Font free for personal/educational use |
| Foggy Forest Song for background music | https://www.playonloop.com/2019-music-loops/foggy-forest/ | Background music in main scene audio source | Creative commons by Attribution 4.0 |
| Stone push sound | https://pixabay.com/sound-effects/stone-push-37412/ | Door sound | Pixabay Content License |
| Projector Button Push sound | https://pixabay.com/sound-effects/projector-button-push-6258/ | Button sound | Pixabay Content License |
| Nature - Essentials \| Nature Ambient | https://assetstore.unity.com/packages/audio/ambient/nature/nature-essentials-208227 | Lava sound | Used under terms of Unity EULA |
| RPG: Essentials Sound Effects - FREE! | https://assetstore.unity.com/packages/audio/sound-fx/rpg-essentials-sound-effects-free-227708 | Jump sound | Used under terms of Unity EULA |

| | | | |
|---|---|---|---|
| Fatality FPS Gaming Font | https://assetstore.unity.com/packages/2d/fonts/fatality-fps-gaming-font-216954 | UI font | Used under terms of Unity EULA |
| Simple Gems and Items Ultimate Animated Customizable Pack | https://assetstore.unity.com/packages/3d/props/simple-gems-and-items-ultimate-animated-customizable-pack-73764 | Diamond at final room | Used under terms of Unity EULA |
| Torch | https://www.turbosquid.com/3d-models/free-obj-model-torch-games-low-poly/334480 | Used for room lighting | TurboSquid 3D Model License |
| Walking sound effect for transition video | https://www.youtube.com/watch?v=rnZ7k6lv4_w&list=PLuE1RIpYgk-UEd_am_f5v4q0VQ1Eizjzb&index=8 | Appears when the character in the transition video is walking | Gfx Sounds Content License |