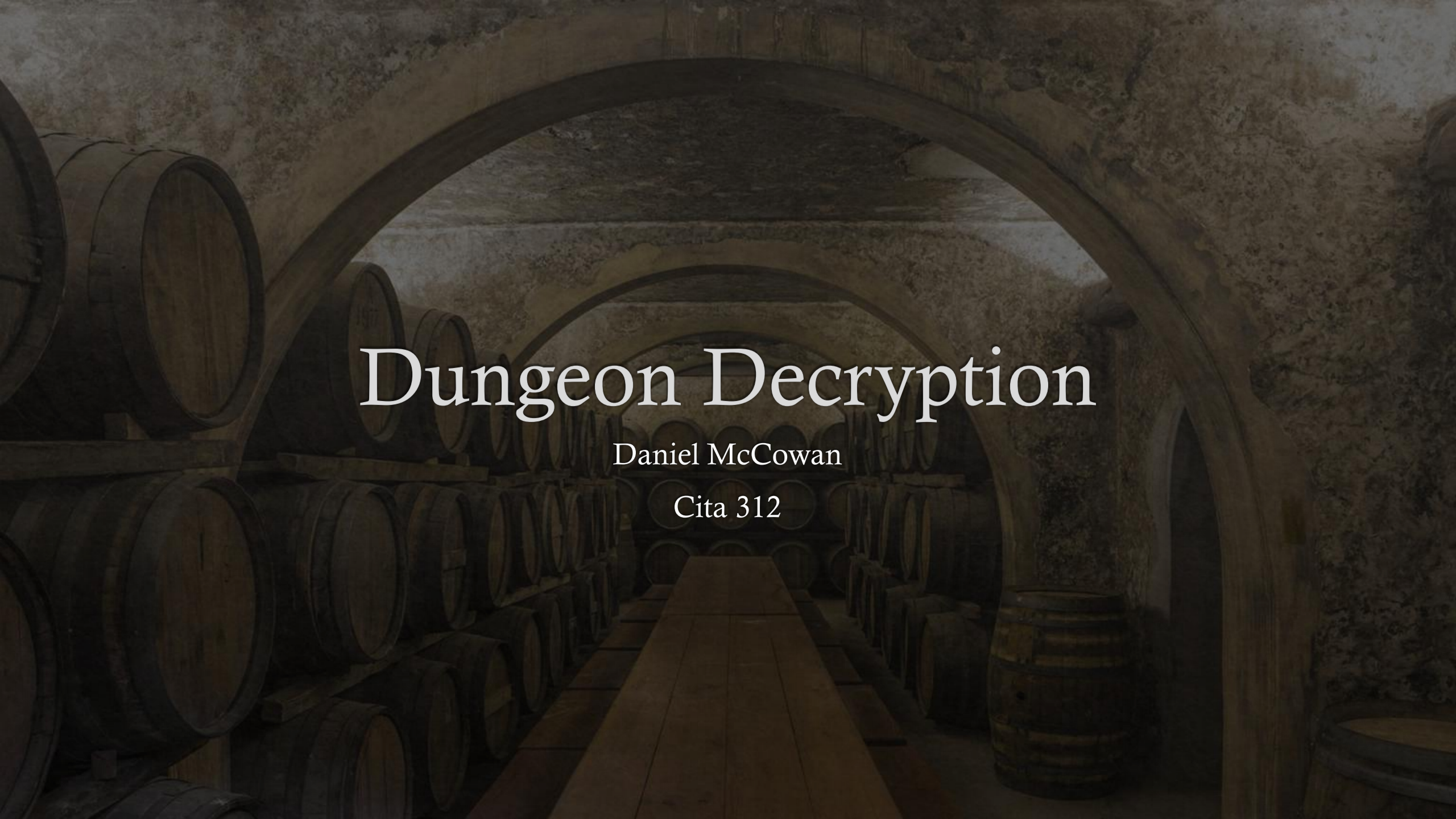


Dungeon Decryption

Daniel McCowan

Cita 312

Overview

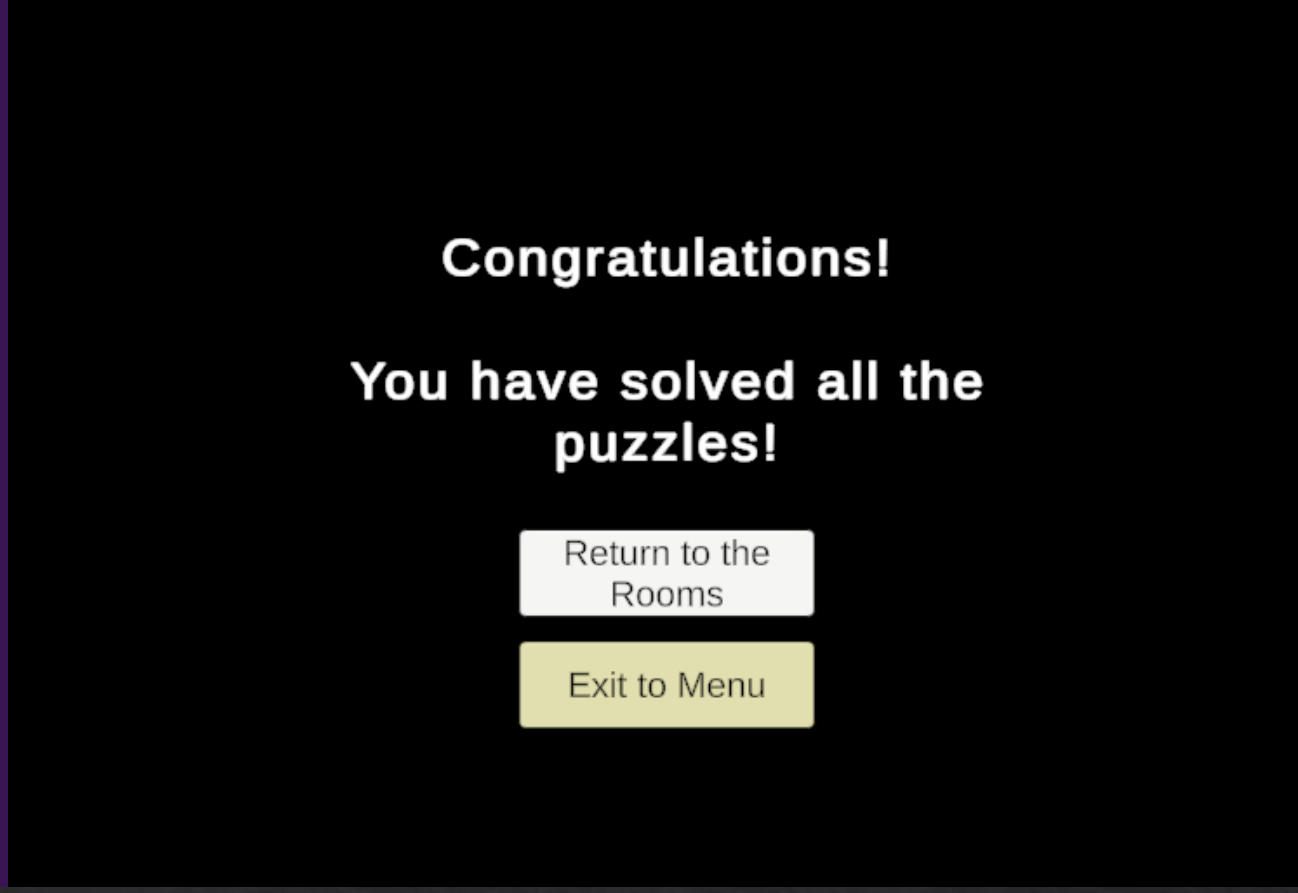
Play as a treasure hunter who has become trapped within a mysterious dungeon

Solve puzzles to escape

The game ends when all of the rooms' puzzles have been solved

Development Highlights


Highlights cont.



Menu & Victory Screens

Scripts

```
void CreateSlidePieces()
{
    for (int row = 0; row < size; row++)
    {
        for(int col = 0; col < size; col++)
        {
            Transform piece = Instantiate(piecePrefab, gameObject.transform);
            pieces.Add(piece);
            piece.localPosition = new Vector3((width * col) - width,
                0,
                -(width * row) + width);
            piece.name = $"{row * size + col}";
            if ((row == size -1) && (col == size -1))
            {
                emptyLocation = (size * size) - 1;
                piece.gameObject.SetActive(false);
            }
            else
            {
                float uvWidth = 1/(float)size;
                quad = piece.GetChild(0);
                Mesh mesh = quad.GetComponent<MeshFilter>().mesh;
                Vector2[] uv = new Vector2[4];
                uv[0] = new Vector2(uvWidth * col, 1 - (uvWidth *(row + 1)));
                uv[1] = new Vector2(uvWidth * (col + 1), 1 - (uvWidth *(row + 1)));
                uv[2] = new Vector2(uvWidth * col, 1 - (uvWidth * row));
                uv[3] = new Vector2(uvWidth * (col + 1), 1 - (uvWidth * row));
                mesh.uv = uv;
            }
        }
    }
}
```

Scripts – SlidePuzzle.cs

```
public void MoveEmpty(Transform selectedPiece)
{
    for (int i = 0; i < pieces.Count; i++)
    {
        if (selectedPiece == pieces[i])
        {
            if (SwapIfValid(i, -size, size)) { break; }
            if (SwapIfValid(i, +size, size)) { break; }
            if (SwapIfValid(i, -1, 0)) { break; }
            if (SwapIfValid(i, +1, size - 1)) { break; }
        }
    }
    if (CheckCompletion() && !hasFinished)
    {
        OnFinish();
    }
}
8 references
bool SwapIfValid(int i, int offset, int colCheck)
{
    if (((i % size) != colCheck) && ((i + offset) == emptyLocation))
    {
        (pieces[i], pieces[i + offset]) = (pieces[i + offset], pieces[i]);
        (pieces[i].localPosition, pieces[i + offset].localPosition) = (pieces[i + offset].localPosition, pieces[i].localPosition);
        emptyLocation = i;
        GetComponent<AudioSource>().Play();
        return true;
    }
    return false;
}
```

```
1 reference
void ProcessTilt()
{
    gameObject.transform.rotation = Quaternion.Lerp(gameObject.transform.rotation, targetRotation, tiltSpeed * Time.deltaTime);
}
1 reference
public void SetTilt(Vector3 direction)
{
    Vector3 prevEuler = gameObject.transform.localEulerAngles;
    deltaX = prevEuler.x + (direction.x * tiltRate);
    if (deltaX < 180)
    {
        deltaX = Mathf.Clamp(deltaX, -tiltMax, tiltMax);
    }
    else
    {
        deltaX = Mathf.Clamp(deltaX, 360f-tiltMax, 360f+tiltMax);
    }
    deltaZ = prevEuler.z + (direction.z * tiltRate);
    if (deltaZ < 180)
    {
        deltaZ = Mathf.Clamp(deltaZ, -tiltMax, tiltMax);
    }
    else
    {
        deltaZ = Mathf.Clamp(deltaZ, 360f-tiltMax, 360f+tiltMax);
    }
    targetRotation = Quaternion.Euler(deltaX, prevEuler.y, deltaZ);
}
```

Scripts – MazeTilt.cs

Scripts – GameManager.cs

```
void HandleMenu()
{
    if (!starterAssetsInputs.pause) return;

    starterAssetsInputs.PauseInput(false);

    uiOverlay.enabled = false;
    pauseMenu.enabled = true;
    starterAssetsInputs.EnableMouseLook(false);
    player.GetComponent<PlayerInput>().enabled = false;
}

5 references
public void ActivateChkpt(int index)
{

    Debug.Log($"Checkpoint Activated {index}");
    if (index > sh.GetChkPtIndex() || sh.GetRestarting())
    {
        sh.SetChkPtIndex(index);
        progressBarHUD.value = index;
        progressBarHUD.GetComponentInChildren<TextMeshProUGUI>().text = $"Progress {index/progressBarHUD.maxValue * 100 : 0}%";
        progressBarMenu.value = index;
        progressBarMenu.GetComponentInChildren<TextMeshProUGUI>().text = $"Progress {index/progressBarMenu.maxValue * 100 : 0}%";
    }
    if (index == progressBarMenu.maxValue)
    {
        OnVictory();
    }
}
```

```
public void Continue()
{
    int index = sh.GetChkPtIndex();
    ActivateChkpt(index);
    currentChkpt = Chkpts[index].transform;
    player.transform.SetPositionAndRotation(currentChkpt.position, currentChkpt.rotation);
    if (index >= 1)
    {
        slidePuzzle.GetComponentInChildren<SlidePuzzle>().OnFinish();
        if (index >= 2)
        {
            mazePuzzle.GetComponentInChildren<MazeFinish>().OnFinish();
            if (index >= 3)
            {
                mazePuzzle.GetComponentInChildren<MazeFinish>().OnSpecial();
            }
        }
    }
    sh.SetRestarting(false);
}
```

Scripts – GameManager.cs cont.

Project Completion

FINAL ROOM BLOCK STACKING PUZZLE

RAN OUT OF TIME DUE TO MULTIPLE OTHER PROJECTS

Challenges & Solutions

UV was not being divided across the slide puzzle pieces	Added a quad above the pieces Was missing the line which actually assigned the piece width correctly
Ball clipping into the maze when it rotated	Physics.ComputePenetration Added a rigidbody to the maze itself

Github and Portfolio

- https://github.com/dmccowa206/dMcCowan_CITA312_PuzzleProject
- https://sites.google.com/view/dmccowanportfolio/home

Thank you