Project design

Overview

This project is based on the ‘Physics-based game’ template, which requires a short, but compelling game using physics modelling as a key element of gameplay.

My proposed design will be based on simulating a tabletop board game. Board gaming is an area that holds especial interest for me, as a structure around which to base social engagements, and for the mental – and occasionally physical – challenges such games provide.

While board games are inherently physical items, a subset - known as *dexterity games* - make explicit use of their physicality. Such games may challenge players to maneuver components using manual dexterity with varying objectives, such as building towers, accurately flicking pieces around a game space, or carefully maneuvering components using tools. Well-known examples of games in this genre include *Jenga, Operation*, and *Subuteo*. This focus on the physical interaction of game components may make a dexterity board game a suitable influence for a project conforming to the provided template of a ‘physics-based computer game’.

The project template specifies that the end product be easy to pick up and play. It is therefore assumed that the users (players) will not necessarily be ‘hardcore’ gamers, familiar with the conventions and control schemes of modern PC gaming. The users may, or may not, have familiarity with the type of physical game upon which the software will be based. Both the gameplay, and the methods of control and interaction must therefore be reasonably intuitive, or quickly learnable. The users may not have access to advanced computer hardware, and may not have use of control pads or other gaming-specific input methods. It will be important that the software can run on a reasonably specified machine, and can be used with mouse-and-keyboard inputs.

Design

The product will be a video game playable on a windows PC. The core gameplay will be based on the commercial dexterity game *Junk Art* [1], a game revolving around stacking objects with unusual shapes, with the goal of creating the tallest structure, or incorporating the most pieces, without the structure collapsing.

The core gameplay loop will involve each player in turn being randomly assigned an unused game piece to be added to their structure. The player will have the ability to move and rotate the active piece, before placing onto their structure, ideally without any pieces falling off. As each player’s structure grows, adding more pieces will become more challenging, until a collapse occurs. This asynchronous, turn-based approach will allow multiple users to play on the same machine, allowing a multiplayer experience without the need to incorporate online play (which would greatly increase the required development and testing resources).

The game will manifest as a representation of a tabletop holding the game pieces. The player will not have an avatar within this space, but rather will use keyboard and mouse to manoeuvre the camera around the environment, similar to the sandbox environment implemented in *Tabletop Simulator* [2]. These camera controls will by default use WASD keys for translation and mouse XY for rotation. This interface will be familiar to anyone having familiarity with modern video games, and should be quick to pick up for those without that experience.

When the player selects a game piece to add to their structure, they must be able to manoeuvre it with six degrees of freedom - three dimensions of translation and three axes of rotation. The system employed by *Garry's Mod* [3] will be a basis for this, with additional control to add the final rotational axis missing from that implementation. Translation will follow camera movement: as the player moves and rotates the camera, the held piece will stay at a fixed distance from the camera, held at its centre. The mouse scroll wheel will be used to adjust the distance between piece and camera. By holding an additional command key (e.g. 'E'), the player can switch to rotation mode; the mouse XY movement and scroll wheel will adjust the piece's rotation within world space, with translation of piece and movement of camera disabled while in this mode. This combined control aspect will need careful testing and iterative development to ensure players can easily manoeuvre pieces as they require, in order to place then onto their structures.

Aside from the 3D world containing the game pieces, the game window will contain several interface elements which are not part of the physics environment. An initial plan for the layout of an in-progress game is illustrated in figure 1. Elements will include information on the game status of the active player, and each other player, taking inspiration from other stand-alone digital boardgame adaptations such as Catan Universe [4] and Carcassonne Digital [5]. This will include points scored in each round of the game, and progress in the current round (height of structure etc.). In a physical game, each player would be randomly assigned a differently shaped component to add to their structure by drawing from a shuffled deck of cards, showing representations of each possible piece. This will be replicated by an interface element, rather than by creating card objects within the 3D environment, which would be needlessly fiddly.

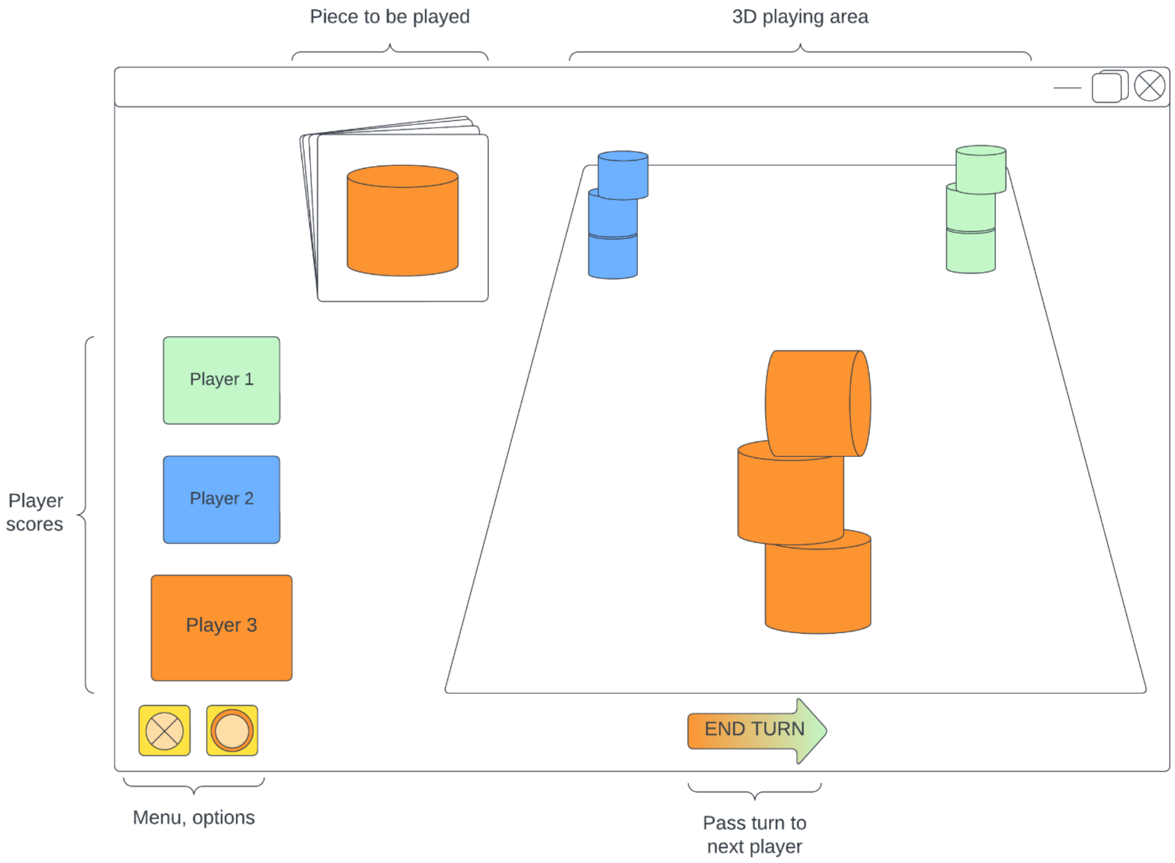


Figure 1 - game layout

A button will allow the player to indicate their turn has ended and pass control to the next player. It is possible that this element may not be required, if the game can be configured to detect when the active player has completed all required actions on their turn and automatically advance to the next player; this would not form part of a minimum viable product, but it is likely to create a smoother playing experience and so would be a desirable feature.

The game will need to detect when one or more game pieces fall from any player's structure. Collision detection and object velocity calculations will trigger functions to indicate whether a player has been eliminated due to a collapsed structure. This will also be linked to audio routines to trigger sounds of components falling and striking the tabletop.

The software will be primarily build using the Unity game engine. This development platform allows for creating and configuring the 3D game environment, physics simulation, and 2D interface overlays. It also allows building of the software for multiple platforms, potentially increasing the range of users for both testing and as users of the final product.

Unity includes by default only primitive 3D components. This will be useful for the prototype phase, but to create the unusual shapes which are key for this game, separate 3D modelling software is required. These custom components will be created in Blender. It will be important to create both the render models and also reasonably high-fidelity mesh colliders. Low-vertex box colliders are often used in 3D games to reduce CPU load, however this game revolves around how specific unusual shapes interact, and generic box colliders would remove these different interactions. Examples of game pieces from the physical game *Junk Art* are shown in figure 2. The digital adaptation for this project will not necessarily replicate these pieces, but these will provide inspiration for components created within the Blender software.

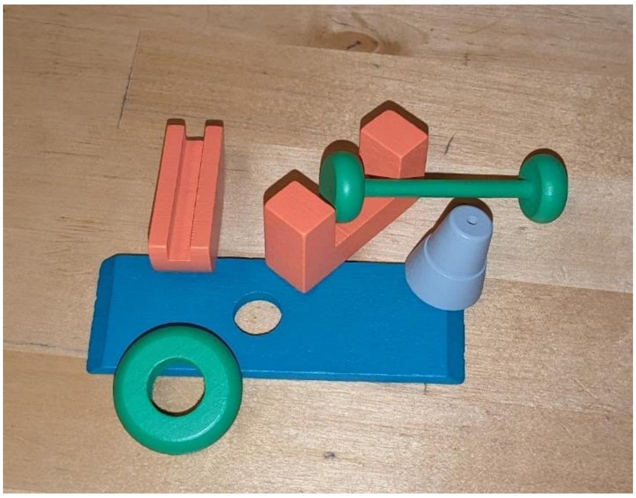


Figure 2 - physical game pieces

2D interface elements will utilise premade assets acquired from the Unity asset store. This resource includes many collections of game UI assets, including free and paid bundles. A free bundle such as *MiMU Studio 2D Casual UI* [6] will be used for initial prototyping. Depending on the outcome of user testing and feedback, an alternative paid asset set may be required.

Work plan

The work plan for this project is illustrated in the Gantt chart in appendix 1. This chart shows the high-level objectives for the planning and development phases of the project, with allowances for break times over the Christmas and New Year periods as well as a pre-arranged holiday.

During the development phase, there will be several build points to allow for user testing of the features implemented to that point. It is intended that this testing will continue while the next stage of development continues. A large amount of time has been allocated to this testing, as it will rely on volunteers who may only be able to dedicate a small amount of their time during the allocated testing periods. Changes and enhancements resulting from user feedback will be incorporated during the entire development phase, with a final week (labelled 'refinement') dedicated to any last changes indicated from the final round of user testing.

Evaluation plan

The aims for this project are derived from a combination of the provided template and the specific design chosen. The final software should be a game that is:

- Stable and bug free
- Easy to pick up and play
- Fun to play
- A good adaptation of a dexterity-type board game

Each of these aims will be evaluated by testing, but the techniques and tester groups required will vary to ensure all aims are covered.

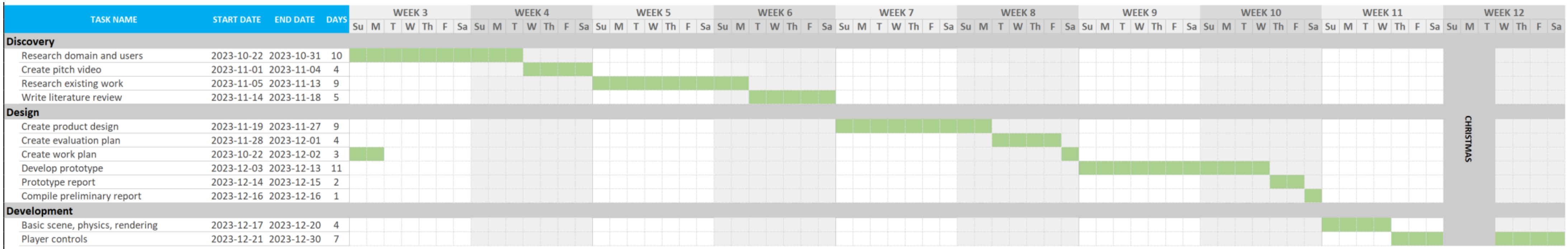
Stability of the software will be evaluated by the developer, through a series of pre-written test cases. High-level cases are included in appendix 2 to this report. These will be refined with more specific cases as each phase of development starts. In each internal testing cycle, all previous tests will be repeated to ensure previously working functionality has not been disturbed. It is unlikely that an automated testing regime will be possible due to the nature of the project and resources available. This internal testing has not been separately marked on the Gantt chart in appendix 2, as this work will be conducted continuously throughout the development phase.

The other three aims detailed above can not be reliably evaluated by the developer alone. This will rely on recruiting volunteers from the developer's friends and family. As mentioned in the work plan section above, there will be several build points for the software, where testers will be able to evaluate the functionality ready at that point. This will form part of the ongoing development and bug-testing process. The later user testing periods will be more informative regarding evaluation against the project aims.

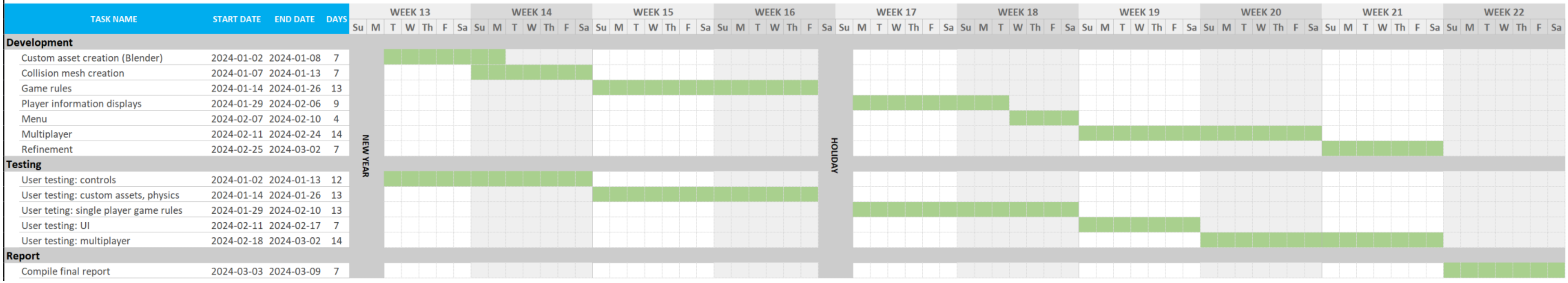
It will be important to recruit testers with a variety of previous experience, both with computer games, and with the board games on which the software will be based. The 'easy to pick up and play' criterion will be best evaluated by testers with limited experience of 3D computer gaming, and who have not been part of earlier testing rounds. The final criterion above will require testers with some familiarity with the type of physical games serving as an inspiration for this project. All testers of the final product should be able to comment on the 'fun' criterion, however as this is the most subjective point it will be important to have a reasonable number of testers.

The testing methodology involving users will also change throughout the project. Initially users will only have a very limited build to test, which will require supervision from the developer to instruct the user what functionality is present, and what feedback is required. As the software is built out, less supervision should be required. For the final product, the software should be entirely self-explanatory (including user help where necessary), and effective feedback can only be achieved by unsupervised user testing.

Appendix 1- work plan



TASK NAME	START DATE	END DATE	DAYS
Discovery			
Research domain and users	2023-10-22	2023-10-31	10
Create pitch video	2023-11-01	2023-11-04	4
Research existing work	2023-11-05	2023-11-13	9
Write literature review	2023-11-14	2023-11-18	5
Design			
Create product design	2023-11-19	2023-11-27	9
Create evaluation plan	2023-11-28	2023-12-01	4
Create work plan	2023-10-22	2023-12-02	3
Develop prototype	2023-12-03	2023-12-13	11
Prototype report	2023-12-14	2023-12-15	2
Compile preliminary report	2023-12-16	2023-12-16	1
Development			
Basic scene, physics, rendering	2023-12-17	2023-12-20	4
Player controls	2023-12-21	2023-12-30	7



TASK NAME	START DATE	END DATE	DAYS
Development			
Custom asset creation (Blender)	2024-01-02	2024-01-08	7
Collision mesh creation	2024-01-07	2024-01-13	7
Game rules	2024-01-14	2024-01-26	13
Player information displays	2024-01-29	2024-02-06	9
Menu	2024-02-07	2024-02-10	4
Multiplayer	2024-02-11	2024-02-24	14
Refinement	2024-02-25	2024-03-02	7
Testing			
User testing: controls	2024-01-02	2024-01-13	12
User testing: custom assets, physics	2024-01-14	2024-01-26	13
User teting: single player game rules	2024-01-29	2024-02-10	13
User testing: UI	2024-02-11	2024-02-17	7
User testing: multiplayer	2024-02-18	2024-03-02	14
Report			
Compile final report	2024-03-03	2024-03-09	7

Appendix 2- example test cases

Test Name	Game loads
Preconditions	None
Test Steps	Open software
Expected Results	Main menu displayed

Test Name	Main menu functions
Preconditions	Main menu displayed
Test Steps	Adjust player count Start game
Expected Results	Game starts with selected player count

Test Name	Camera movement
Preconditions	Game started
Test Steps	Use mouse and WASD keys to move camera around game world
Expected Results	Camera responds to input

Test Name	Game piece translation
Preconditions	Game started
Test Steps	Click and hold to pick up piece Use camera movement controls Use scroll wheel
Expected Results	Game piece is held at camera focus Game piece moves with camera Game piece moves to/from camera with scrolling

Test Name	Game piece rotation
Preconditions	Game piece held
Test Steps	Hold rotate button Move mouse, scroll wheel
Expected Results	Game piece rotates around three axes

Test Name	Gravity and collisions
Preconditions	Game piece held
Test Steps	Move game piece around world Release game piece
Expected Results	Held piece collides with other pieces, moving them Released piece falls under gravity, colliding with other pieces

Test Name	Add to structure
Preconditions	Game started
Test Steps	Use controls to add piece to player’s structure
Expected Results	Piece detected as part of structure Structure height updated

Test Name	Structure collapse
Preconditions	Game started Player structure exists
Test Steps	Place piece such that at least one piece falls from structure
Expected Results	Fallen piece(s) detected Structure height updated Player score reduced

Test Name	Pass turn
Preconditions	Game started Active player has placed piece in their structure
Test Steps	Click ‘pass turn’ button
Expected Results	Next player camera set as active Active player indicator in interface updates

Test Name	Player wins
Preconditions	Game started
Test Steps	Active player reaches target structure height
Expected Results	Game end screen displayed Active player declared winner Scores displayed

Test Name	Return to menu
Preconditions	Game started
Test Steps	Click menu button
Expected Results	Main menu displayed Return to game in progress button displayed Start new game button displayed

Test Name	Return to game in progress
Preconditions	Game started Menu button clicked
Test Steps	Click return to game in progress
Expected Results	Game resumed in status as left

Test Name	Start new game
Preconditions	Game started Menu button clicked
Test Steps	Click start new game
Expected Results	New game starts with selected player count

References

[1] Asmodee, “Junk Art 3.0,” [Online]. Available: https://www.asmodee.co.uk/products/pbgpzg20030_junk-art-3-0. [Accessed 25 10 2023].

[2] Berserk Games LLC, “Tabletop Simulator,” [Online]. Available: https://www.tabletopsimulator.com/. [Accessed 26 10 2023].

[3] Facepunch Studios, “Garry's Mod,” 29 11 2006. [Online]. Available: https://gmod.facepunch.com/. [Accessed 17 11 2023].

[4] KOSMOS Publishing, “Catan Universe,” [Online]. Available: https://catanuniverse.com/en/. [Accessed 16 11 2023].

[5] Asmedee Digital, “Carcassonne,” 2019. [Online]. Available: https://www.asmodee-digital.com/en/carcassonne/. [Accessed 25 11 2023].

[6] “2D Casual UI HD,” MiMU Studio, 23 02 2017. [Online]. Available: https://assetstore.unity.com/packages/2d/gui/icons/2d-casual-ui-hd-82080. [Accessed 26 11 2023].