Game Proposal: Raycast

CPSC 427 – Video Game Programming

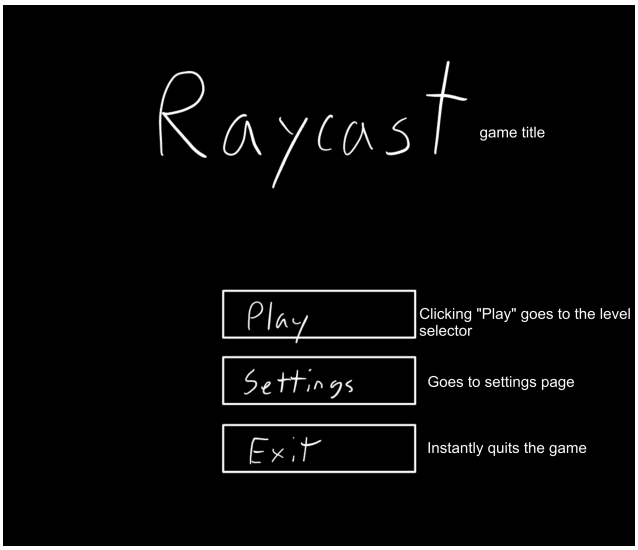
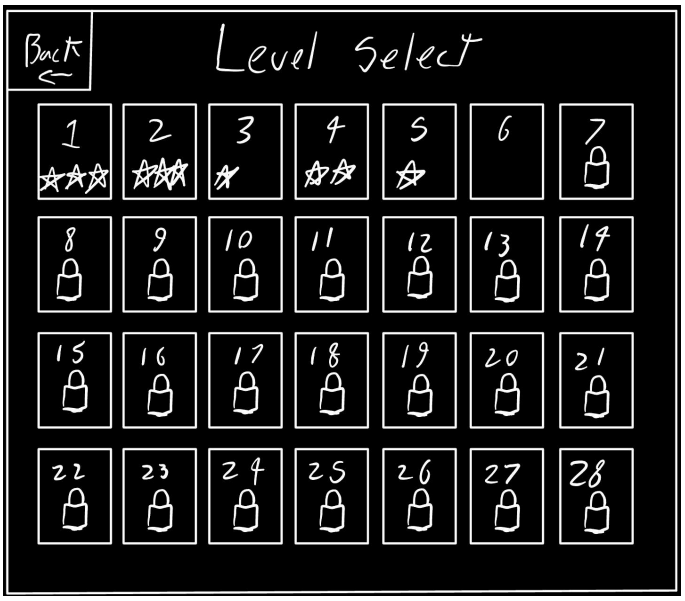
Team: Lightbox Studios

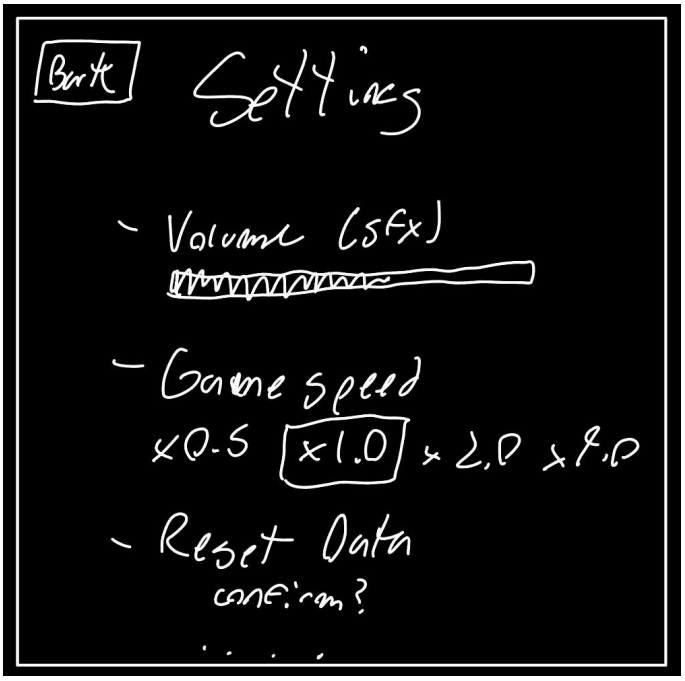
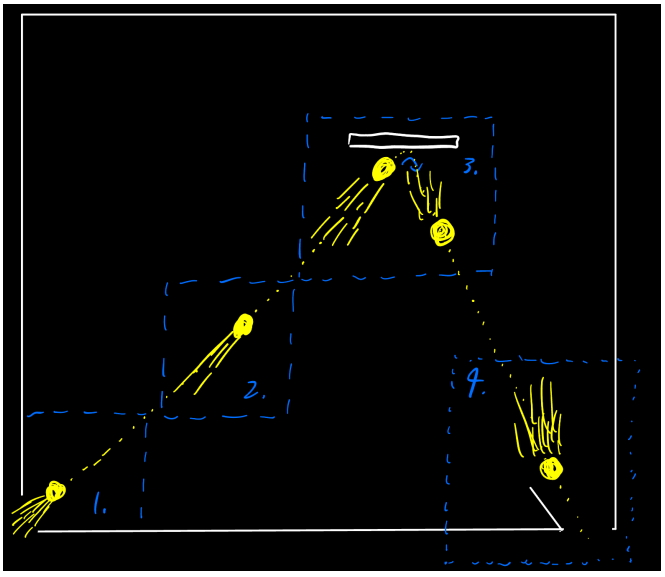
Name	Student Number
Jackson Li	60526746
Gwen Kornak	23685241
Suraj Emmattiparambil	34148627
Muhammad Assad	54398540
Arpit Kumar	54272380
Jocelyn Baker	74137134

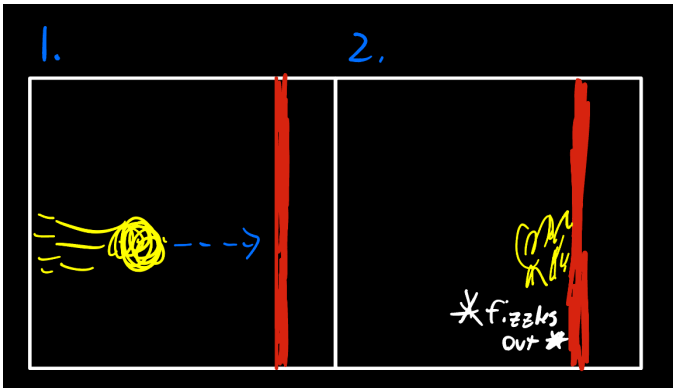
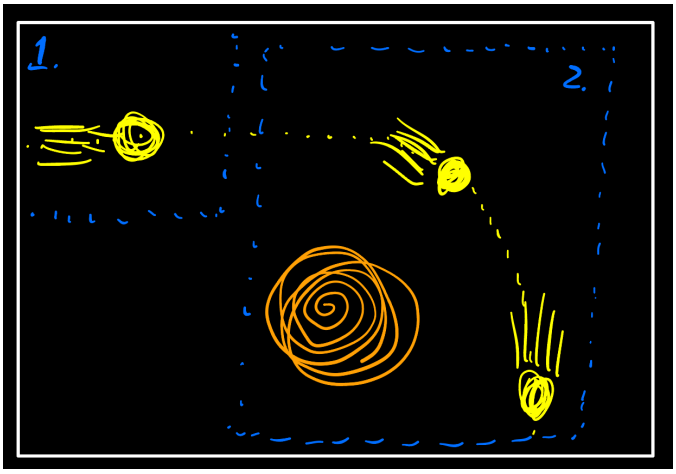
Story:

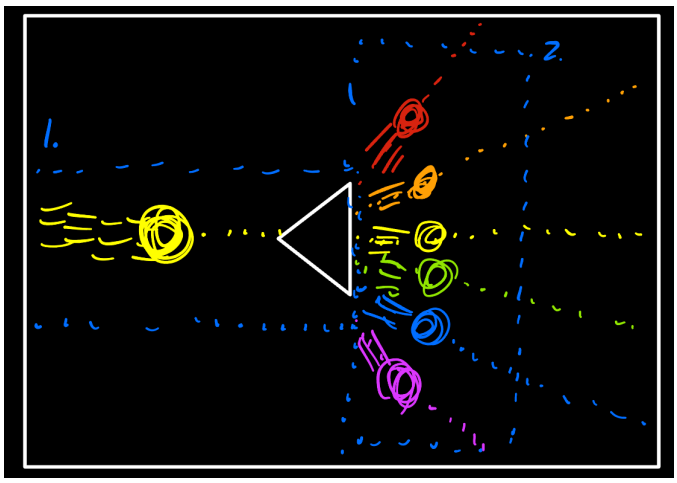
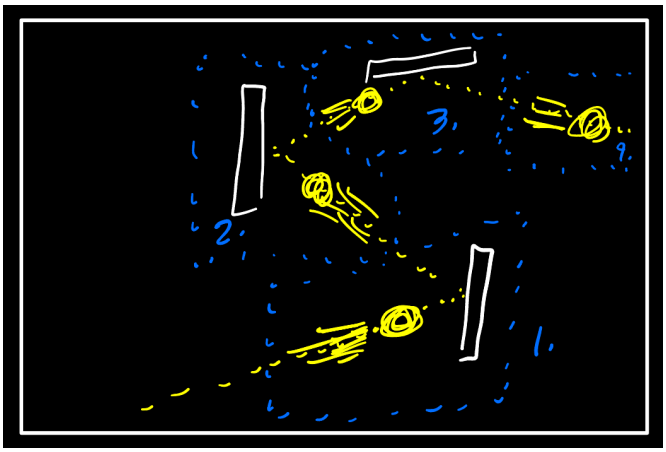
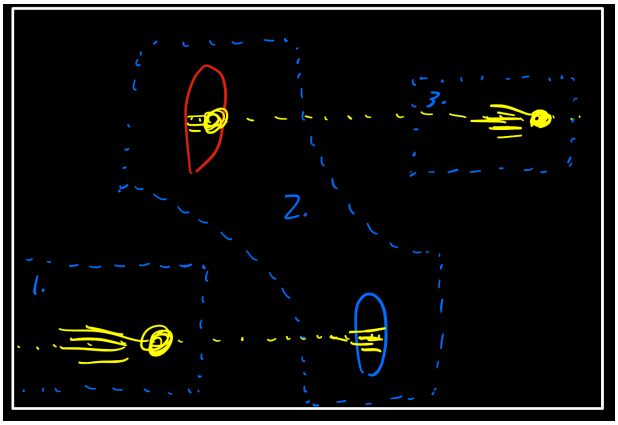
The world has fallen into darkness, and nearly all life has disappeared. The last surviving plant needs light to photosynthesize. As the player, you guide the final source of light through a series of levels by moving mirrors, navigating around blocks and avoiding getting trapped in black holes. Each level presents new challenges in guiding the light, with the ultimate goal of reviving the last plant on Earth.

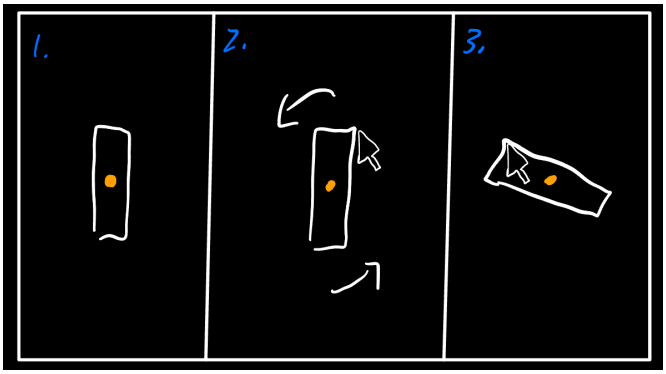
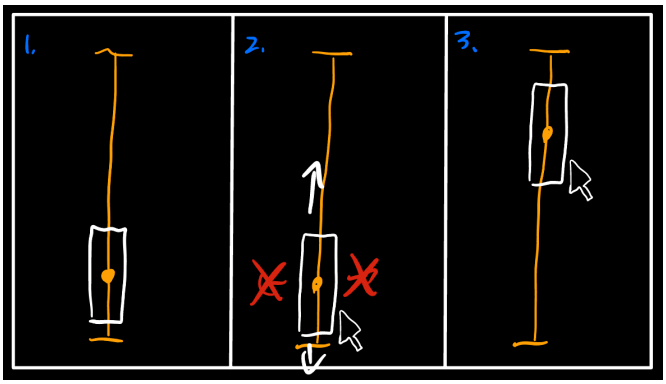
Scenes and Interactions:

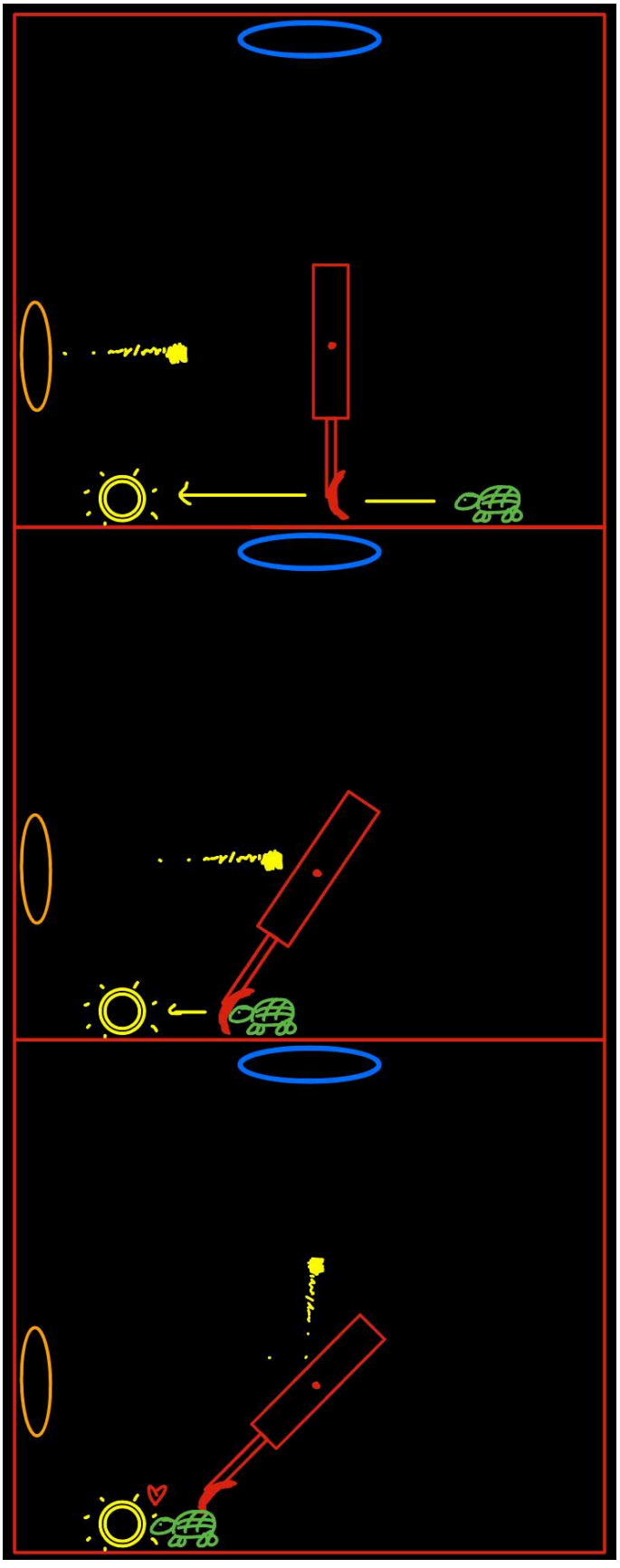
Scene	Picture	Description
Title Screen		None.
Level Selector		• Levels are either **completed, unlocked, or locked**

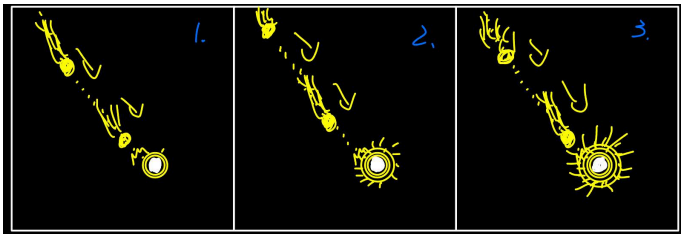
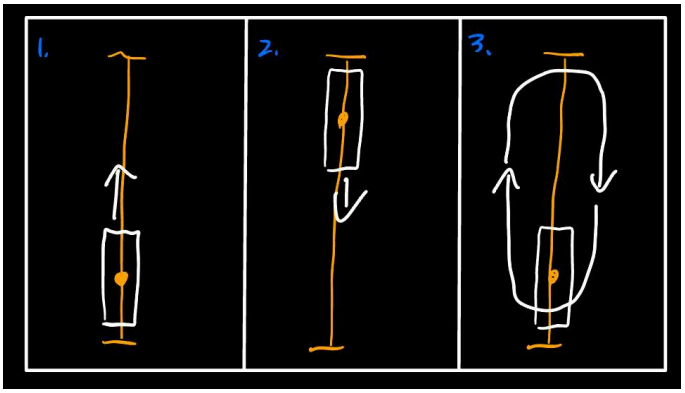
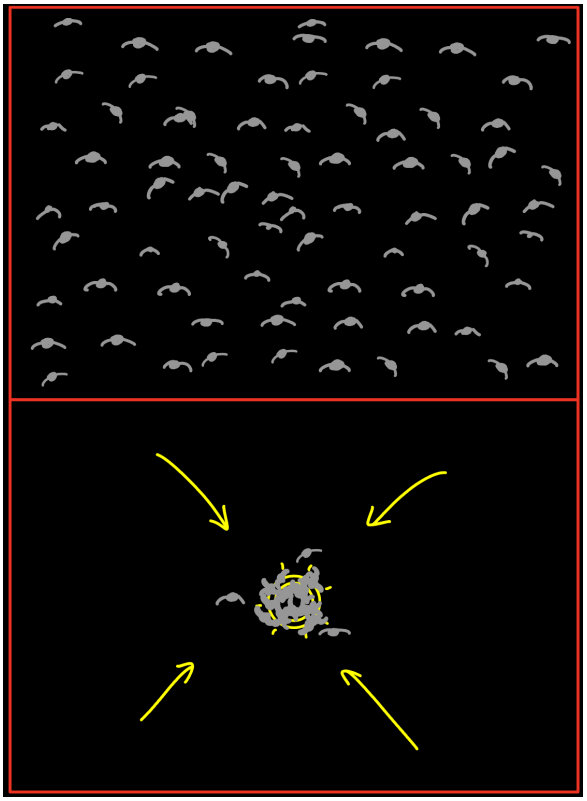
Settings Menu		- Multiple settings for the player to modify - Volume effects the sfx of the game - Gamespeed will adjust the rate the light (and other mechanisms) move at - Reset Data allows the player to clear all saved data
Light colliding with reflective surface into endzone		1. The light is shot out periodically (automatically) at a **fixed angle** (determined by the level design) from the **start zone** 2. The light travels in the direction it was shot at 3. The direction of the light changes when it collides with a **reflective surface (e.g. mirror)** 4. The level is completed when the light is guided through an **endzone**

Light colliding with non-reflective surface		1. Light heading towards a **non-reflective surfaces** 2. Will cause the light to fizzle out
Light traveling through black hole area		1. Light heading towards an area affected by a **blackhole** 2. Will adjust the trajectory of the light path using **orbital mechanics**
Light getting absorbed by a black hole		1. The blackhole pulls light towards it 2. Depending on how close the light is to the black hole, it can spiral inwards to the black hole, being destroyed

Light refracted upon colliding with a prism		1. Prism's split the light into multiple coloured lights 2. These coloured lights continue on their own paths, and can be curved and navigated like other light
Light reflecting multiple times consecutively		1 - 4. Light can interact with multiple objects such as reflecting off of multiple mirrors
Light travelling through a portal		1-3. Portals move the light from one area in the scene to another (possible extension feature)

User interacting with environment objects via Rotation		1-3. Interactive objects fixed in place can be rotated by click + drag
User interacting with environment objects on a fixed path		1-3. Objects fixed to rails, can only be moved along its fixed path

AI turtle affecting objects in the level		1. Charged mini suns can attract AI creatures 2. AI creatures are able to interact with objects in the environment 3. Will stop once they reach the charged mini sun

Charging a light source		If light keeps hitting a light source, it will slowly get brighter and brighter. Until it is charged (indicated visually) This can be used to attract AI creatures
Powered moving mirror		Some rails are powered, moving back and forth continuously This behaviour can also be dynamically controlled by a connected mini sun — if the sun is charged and unobstructed, the platform moves.
AI swarm covers map until a mini sun is charged		The swarm affects gameplay in a number of ways for mini sun levels. First, it stops the first lit mini sun from distributing its power, and second, it stops that mini sun from attracting turtles. Not shown here, swarm will also slightly concentrate around light rays.

Technical Elements:

Rendering: Building on the light-based gameplay, the levels will be dynamically lit by the light rays as they move through the level with 2D raycasting. To allow the player to see the level elements when they're not illuminated by the light ray, basic shaders will provide dim illumination to all level elements.

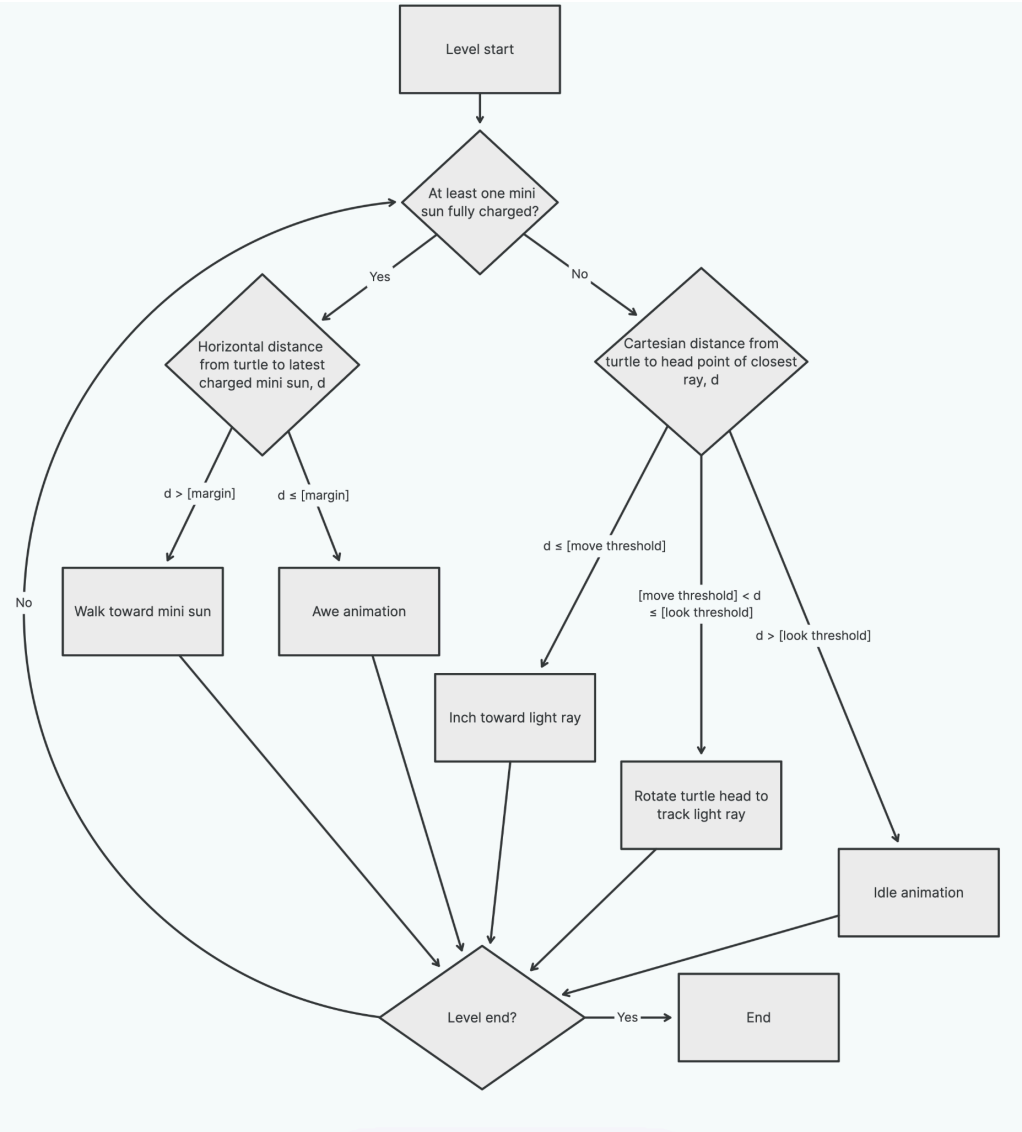
Moreover, the user-controlled movement of various objects throughout a level should be reactive.

Geometric/Sprite/Other Assets: The ray of light will be defined by a current position and a path of its previous positions, which will be used to create a glowing ray of light with a trail using a shader. The environment backgrounds and interactable level objects will be composed of assets from kenny.nl and custom drawn elements.

2D Geometry Manipulation (transformation, collisions, etc.): The light ray will move forward at a **constant speed** after being spawned. It will also be affected when it collides with a level object. For example, if the object is a mirror, the ray will be reflected, or if the object is a black hole, the ray will be bent toward the hole. Some level objects can also be moved or rotated by the player.

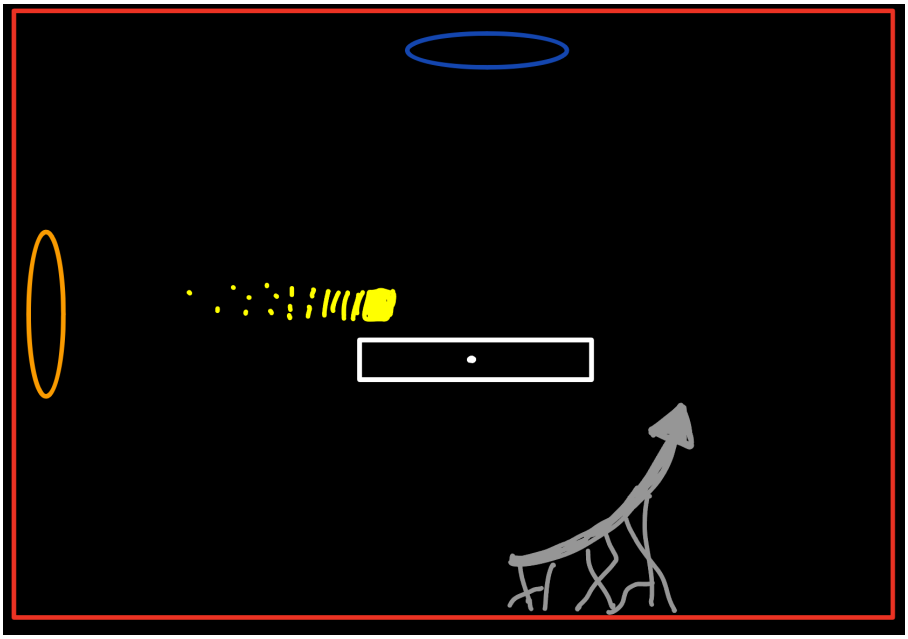
Gameplay Logic/AI:

- **AI**
    - In some levels, there is a turtle-like creature that is attracted to light. The player must strategically direct the light to the "mini-sun" (object that absorbs light) objects to charge them up. When these suns are fully charged, the turtle is drawn to them, pushing levers and objects as it walks toward the sun, manipulating the objects in ways the player cannot directly control. This mechanic introduces unique environmental puzzles where the player guides both the light and the turtle to solve challenges.
    - **Decision Tree:**



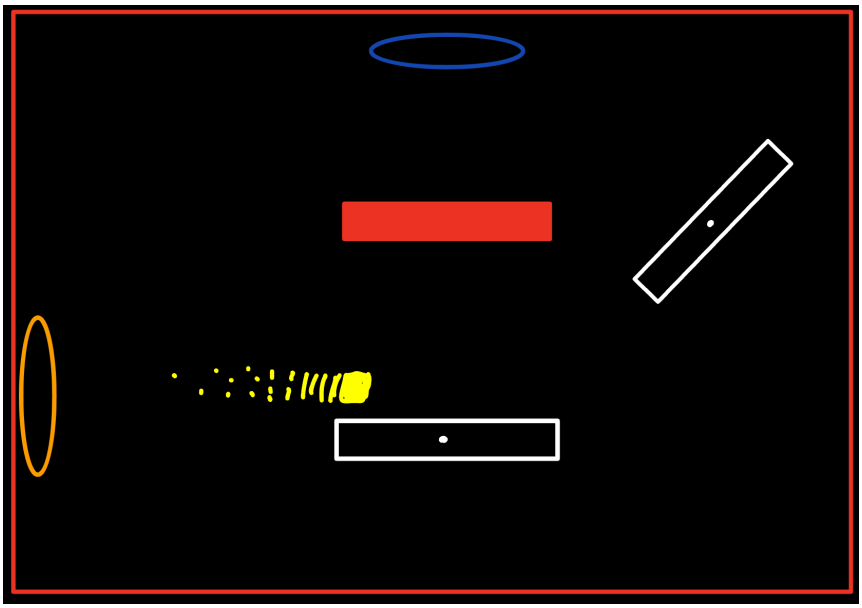
- **Level Design and Progression**
  - **Intro cutscene:**
    - **Shot 1:** The earth spins in space with the sun behind it. Strauss' Op. 30 plays (copyright free, of *2001: A Space Odyssey* fame), and right when the music is about to hit its triumphant crescendo, the sun is sucked into a black hole, leaving only a faint yellow speck of light where the sun was. The earth loses its colour as a foreboding drone plays. The speck moves in the direction of earth.
    - **Shot 2:** Follows the speck as a light ray zipping past stars in the void of space. The earth comes into view, and the camera locks in place. The ray enters the atmosphere, getting further and further from the camera as it falls toward the earth.
  - **Level 1: Introducing mirrors**
    - **Detailed description:**

- The light enters from a hole on the left side of the level and must exit through a hole on the top side of the level. At the level's centre is a mirror with a centre pivot, placed in a horizontal orientation. Light rays fly barely over the mirror at regular intervals and fizzle out on the right wall. The apocalyptic refuse forms an arrow curving from down and to the right to upward (suggesting a counterclockwise rotation).
- To complete the level, the player follows the arrow's indication to rotate the mirror 45 degrees and reflect the light ray into the top hole.
- **Diagram:**

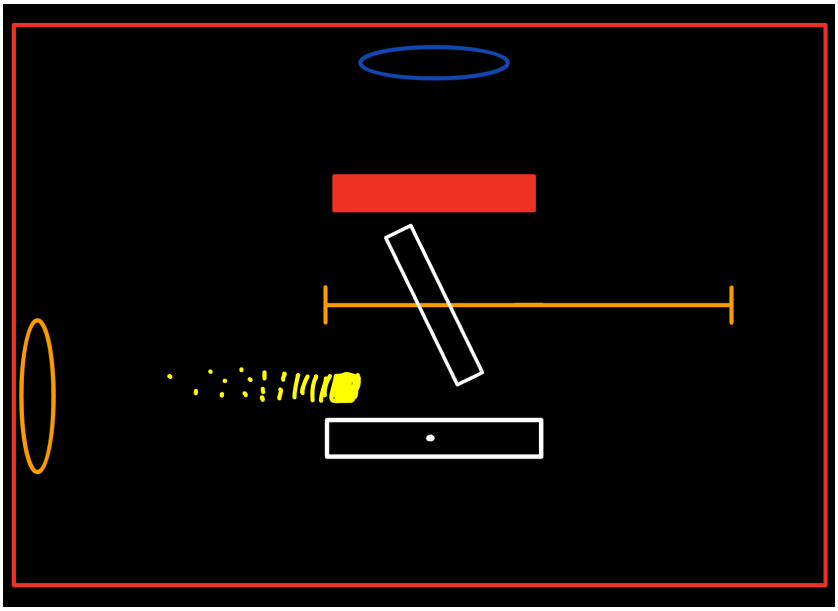


- **Level 2: Introducing barriers**
  - **Detailed description:**
  - The light enters from a hole on the left side of the level and must exit through a hole on the top side of the level. Just like the previous level, there is a mirror in the center, but above the mirror between it and the exit hole sits a barrier, blocking the light from taking that path. Another mirror to the right of the block with a centre pivot is also added.
  - To complete the level, the player realizes the light fizzles out when it hits the barrier, and discovers that they can reflect the light off the centre mirror into the added mirror, and angle the added mirror to send the light into the top hole.

- **Diagram:**

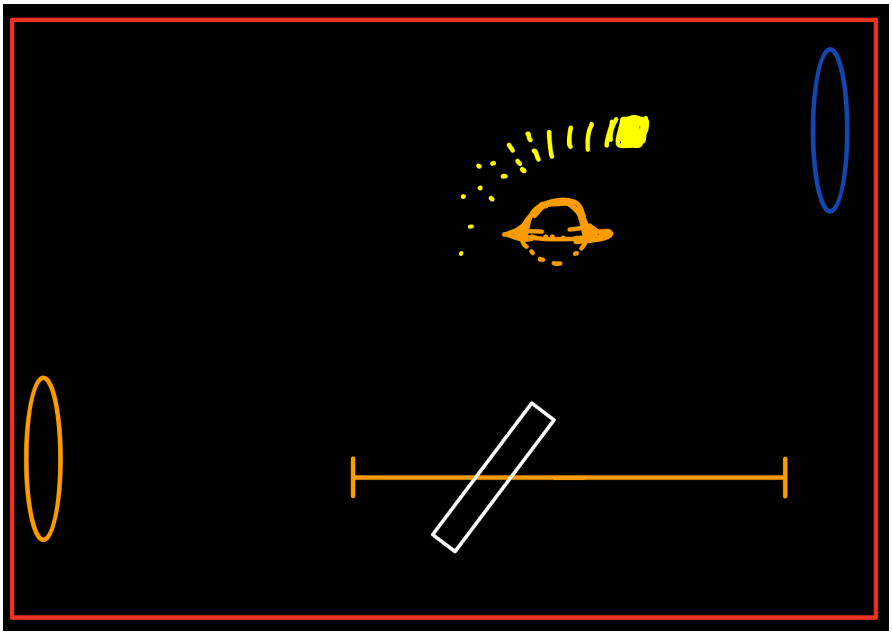


- **Level 3: Introducing moving mirrors**
  - **Detailed description:**
  - Another level that builds on the previous two. Now the added mirror from level 2 has been rotated to direct the light into the hole, but it is positioned between the centre mirror and the barrier. The added mirror is on a track.
  - The player is must move it to the right, rotating the centre mirror to bounce it off the added mirror at the correct angle into the top hole
  - **Diagram:**



- **Levels 4 – 7: Challenging mirror and barrier puzzles**

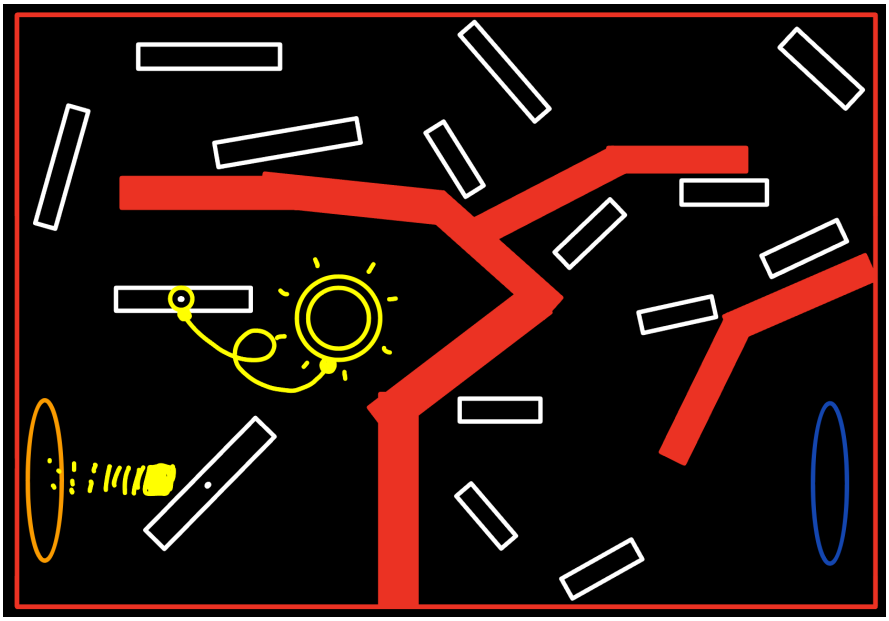
        - Multiple mirrors must be correctly rotated and moved to navigate light through complex mazes of barriers in each of these puzzles.
        - The specifics of these designs will be determined with playtesting prototypes of the game and with physical models.
    - **Level 8: Introducing black holes**
        - **Detailed description:**
        - The light enters from a hole on the lower left side of the level and must exit through a hole on the upper right side of the level. A mirror at a fixed angle and moveable on a horizontal rail at the same height as the entrance hole reflects the light into the level's top wall.
        - A black hole at the centre of the level must be used to bend the light into the exit hole.
        - The player gains an intuition for the black hole's strength as they move the mirror until they can find the right horizontal distance from the black hole to bend the light enough to reach the exit hole.
        - **Diagram:**



    - **Levels 9 and 10: Challenging puzzles with black holes**
        - Multiple mirrors must be correctly rotated and moved to navigate light through complex mazes of barriers and around black holes in each of these puzzles.
        - The specifics of these designs will be determined with playtesting prototypes of the game and with physical models.
    - **Level 11: Introducing mini sun**
        - **Detailed description:**
        - Light enters from a hole on the lower left side of the level. A cable snakes from an uncharged (dark) mini sun above and to the right of the entrance hole up to a mirror above and to the left of the entrance hole

sending the light into a barrier, on a central pivot but not yet able to rotate. A rotatable mirror directly beside the entrance hole bounces the light to the mini sun-connected mirror. Other mirrors form a complex path around barriers to an exit hole in the lower right side of the level.

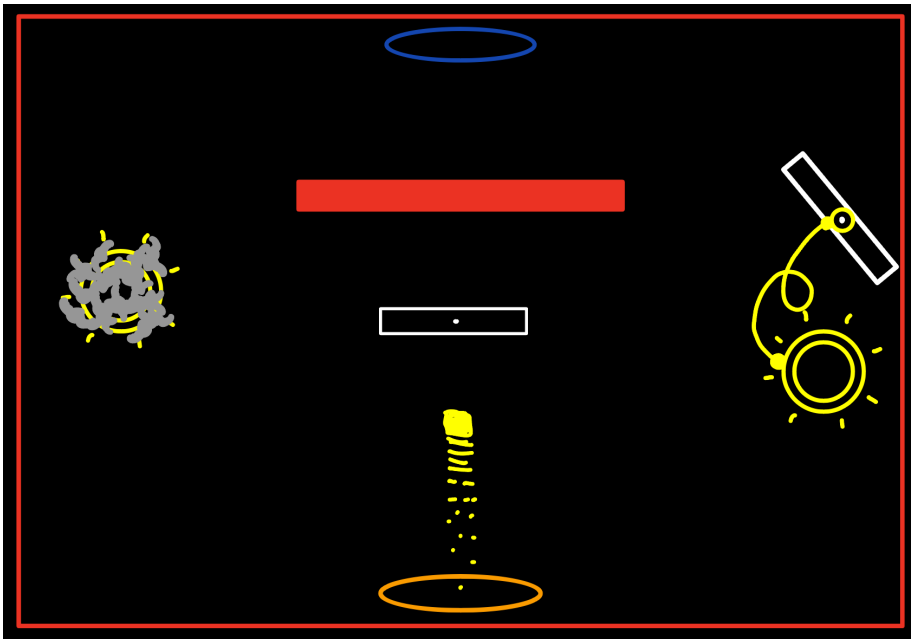
- The player must rotate the mirror above the entrance hole to charge up the mini sun, causing the pivot on the connected mirror to glow and allow the player to direct the light into the complex path to the exit hole.
- **Diagram:**



- **Levels 12 – 14: Challenging puzzles with mini sun**
  - Multiple mirrors must be correctly rotated and moved to navigate light through complex mazes of barriers and around black holes in each of these puzzles, activating certain elements by charging (and avoiding charging) connected mini suns.
  - The specifics of these designs will be determined with playtesting prototypes of the game and with physical models.
- **Level 15: Introducing swarm**
  - **Detailed description:**
  - Light enters from a hole on the bottom of the level, hitting a rotatable mirror that reflects the light back through the entrance hole. Flies swarm around the level randomly, those close to the light clustering slightly around it. One mini sun on each side of the level is reachable using the mirror. A barrier above the mirror blocks the exit hole at the top of the level. A mirror connected to the right mini sun needs to be powered to allow it to be moved.
  - If the player charges the right mini sun, they will learn that the flies will cluster around it, covering its light and preventing its connected mirror

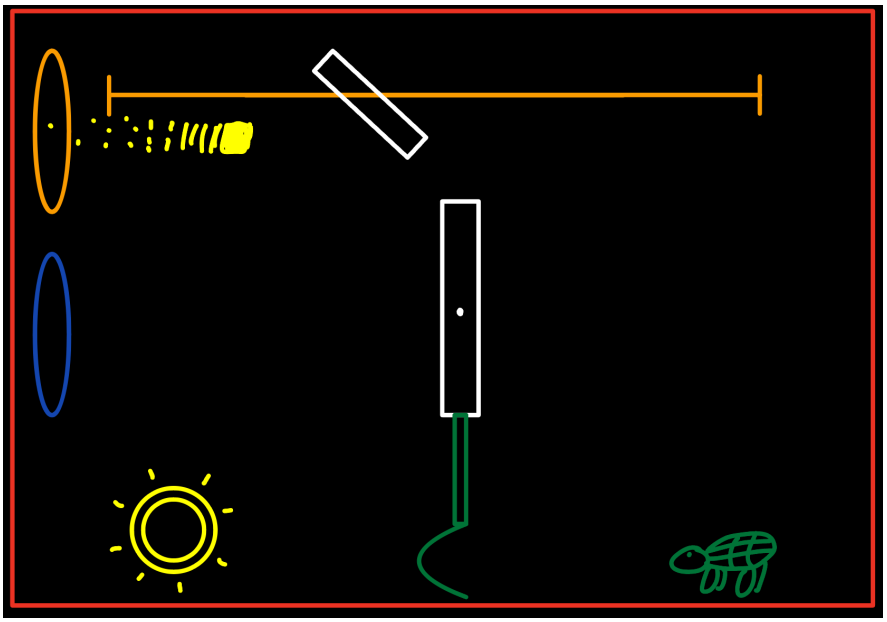
from being powered. Instead, after restarting the level, they must charge the left mini sun first, causing the flies to swarm around it and ignore the right mini sun once it's charged, allowing them to move the connected mirror and solve the level.

- **Diagram:**



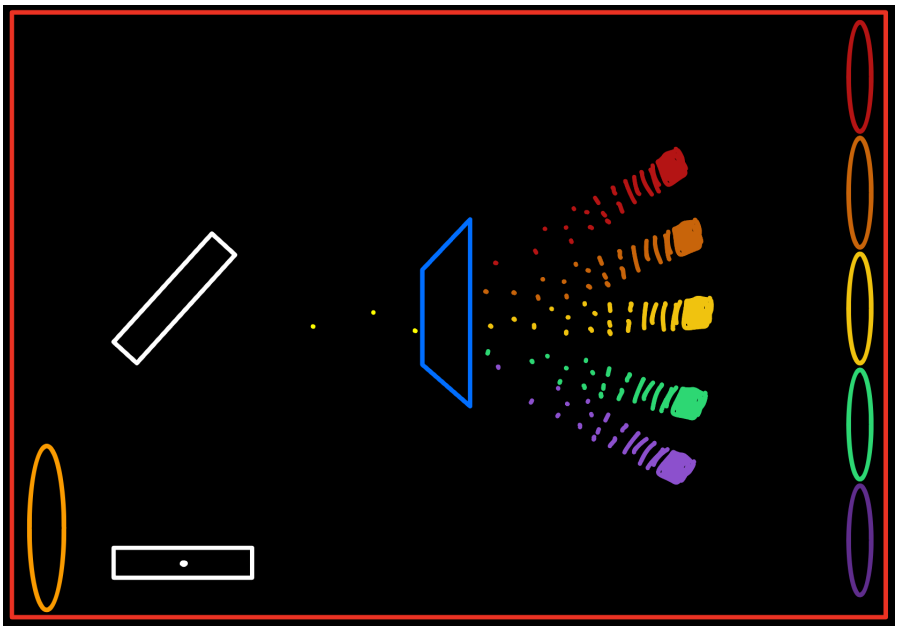
- **Levels 16 – 19: Challenging puzzles with swarm**
  - Multiple mirrors must be correctly rotated and moved to navigate light through complex mazes of barriers and around black holes in each of these puzzles, activating certain elements by charging (and avoiding charging) connected mini suns while dealing with the swarm.
  - The specifics of these designs will be determined with playtesting prototypes of the game and with physical models.
- **Level 20: Introducing turtle**
  - **Detailed description:**
  - Similar to Level 1, a single mirror not controllable by the player sits in the centre of the level with a lever arm extending below it, angled vertically. The exit hole is at the same height as the mirror on the left wall. Light enters from a hole on the top left of the level and is reflected downward by a mirror on a rail. An uncharged mini sun sits on the bottom left of the level and a turtle on the bottom right, both reachable by moving the top mirror.
  - The player must move the mirror on the rail to charge the mini sun, which will cause the turtle to walk toward it, moving the lever arm in the process and enabling the centre mirror to reflect light from the mirror on the rail into the exit hole.

- **Diagram:**



- **Levels 21 – 23: Challenging puzzles with turtle**
  - Multiple mirrors must be correctly rotated and moved to navigate light through complex mazes of barriers and around black holes in each of these puzzles, using the turtle and mini suns to move certain elements, and activating other elements by charging (and avoiding charging) mini suns while dealing with the swarm.
  - The specifics of these designs will be determined with playtesting prototypes of the game and with physical models.
- **Level 24: Introducing prism**
  - **Detailed description:**
  - Light enters through a hole on the bottom of the left wall of the level. Exit holes of various colours line the right wall of the level. A prism sits at the centre of the level, as well as a mirror at the same height fixed to angle light from below into the prism, and a rotatable mirror below the fixed mirror angled horizontally, allowing light to pass over in the start state and go through one of the exit holes — not ending the level.
  - The player must angle the rotatable mirror to direct the light into the prism, resulting in the beam splitting into multiple coloured beams (corresponding to the exit holes). The coloured beams will then enter the exit holes, completing the level.

- **Diagram:**



- **Levels 25 – 30 (variable depending on available time): Complex puzzles with all mechanics**
  - **Detailed description:**
  - Multiple mirrors must be correctly rotated and moved to navigate light through complex mazes of barriers and around black holes in each of these puzzles, using the turtle and mini suns to move certain elements, and activating other elements by charging (and avoiding charging) mini suns while dealing with the swarm. Prisms must be used to satisfy multiple exit holes, as well as other elements like mini suns and turtles if colour-indicated.
  - The specifics of these designs will be determined with playtesting prototypes of the game and with physical models.
- **Ending cutscene:**
  - **Shot 1:** The light ray zooms through holes, picking up speed (imagine the first-person perspective of an object dropped through vertically stacked portals), until it flies through a final hole through which is a withering plant. Time slows and we watch as the light slowly illuminates the plant, then hits it, causing the plant to become vibrant and an exploding wave of colour to explode out from it.
  - **Shot 2:** The earth in space. Emanating from a single point, a wave of light and colour expands outward, revitalizing the earth and bathing it in colour.
  - **Shot 3:** Credits over the spinning, now revitalized, earth.

Plan B

Above is the proposed level plan, but to ensure we provide an enjoyable and complete experience if implementation takes longer than anticipated, we will be implementing the levels in the following order: 1, 2, 3, 8, 11, 20 (all of the introductory levels excluding swarm and prism), then 4, 9, 12, 21 (four challenge levels with each of those mechanics), followed by the remaining levels in order if time permits.

Physics:

- The behavior of light rays that are reflected on a mirror will be governed by the **Law of Reflection**. This will apply to plane mirrors, concave mirrors, convex mirrors, etc.
- The behavior of light rays that are refracted through a medium will be governed by **Snell's Law.**
- Light passing through (coloured) bandpass filters will only retain the bands of light that can pass through that filter (i.e. the light in the colour of the filter), governed by **optical transmittance** principles.
- We will make use of prisms and other objects that **refract** light into its constituent colors.
- When light moves through one medium to another (for eg. water to air), the intensity of the light can be affected by **internal reflection.**

Advanced Technical Elements:

Rendering

- The ray of light illuminates the area around it as it travels through the level, casting dynamic 2D shadows.
- We will add bump maps on top of our textures to give the assets more depth and improve the overall reactivity to light in the game. The maps can be generated using a free tool like Laigter, and it can be sampled in the lighting calculations in our fragment shader.
- If time permits, we will implement a particle system based animation for the trail of light behind the head of the light ray. These animations could also be used in collisions of the light ray with different surfaces (to visualize the light ray "losing photons" on each collision).

Level Editor

- Players are given access to a level editor using which they can make their own levels. This would allow players to keep their gameplay experience fresh and express their creativity. It'll also help us as developers to build the levels.
- We hope to have a click-and-drag based UI by the end of the term, but if we aren't able to reach that stage, we hope to at least have a DSL or similar headless abstraction that we can use internally.

AI (additional)

- Flies that display swarm behaviour and can pathfind in 2 dimensions. Flies will move towards fully charged mini suns and obstruct their light, preventing them from powering objects or attracting turtles.
- Will use the Boids algorithm for swarm pathing: https://eater.net/boids

Devices:

PC Only:

Mouse

- Left-click
  - Select an interactive object
    - Move object by dragging inside the object
    - Rotate object by dragging outside the object
  - Navigate menus
  - Select menu option
- Right-click - N/A
- Scroll-wheel - N/A

Keyboard

- Not preferred for player
- Arrow Keys
  - Move cursor
  - Navigate menus
- Selection Button (E)
  - Select menu option
  - Click at cursor position
  - Hold button to drag
- Esc
  - Open up pause menu

Controller

- Not supported

Touch

- Not supported

Tools:

Specify and motivate the libraries and tools that you plan on using except for C/C++ and OpenGL.

2D Lighting Algorithms

- https://www.redblobgames.com/articles/visibility/ will be useful for implementing lighting from the light rays
- https://www.roguebasin.com/index.php?title=Field_of_Vision
- **libfov**, **permissive-fov**, and **libtcod**

Rigid Body 2D Physics

- https://github.com/slembcke/Chipmunk2D

Team management:

We will be using a trello board in order to assign and track tasks. This board contains the following categories:

- Needs Points
- Ready for Development
- In Progress
- In Review
- Done
- Sprint Backlog
- Product Backlog

Tasks will be created using task templates, and then story-pointed by the team. It will then be assigned to people based on **interest** and **skill.** The goal is to assign everyone equal amounts of story points to ensure a fair workload. We are unsure of what is everyone's expected **points per sprint** for this first sprint so we will hold a retrospective at the end to determine it.

Roles - Each member will be given roles. This does not mean they solely focus on their roles, it is assigned based on **experience** such that if we have an ambiguous decision for a certain topic we can refer to the specified team member for the last opinion.

Jackson - Product Management, User Experience

Suraj - Rendering

Gwen - Assets, Gameplay Logic

Arpit - Assets, Physics, Story / Theme

Jocelyn - Level Design, Story / Theme, Rendering

Muhammad - Physics, Level Design

Development Plan:

Provide a list of tasks that your team will work on for each of the weekly deadlines. Account for some testing time and potential delays, as well as describing alternative options (plan B). Include all the major features you plan on implementing (no code).

Milestone 1: Skeletal Game

#	Category	Task Breakdown	Marks
[1]	Rendering	Textured geometry • background - gray (not interactive) • foreground - black (interactive) • light ray - white square • mirror - thin light blue rectangle • Start zone and end zone - will not have texture for now, it is an opening section from foreground sections	8
[2]	Rendering	Basic 2D transformations • Mirror object can be translated by the player • Mirror object can be rotated by the player	8
[3]	Rendering	Key-frame/state interpolation • Linear interpolation for light ray's position and color	8
[4]	Gameplay	Keyboard/mouse control • User mouse input to change mirror position or orientation	8
[5]	Gameplay	Random/coded action • Periodic firing of light rays to be automatic	8
[6]	Gameplay	Well-defined game-space boundaries • Light rays either **dissipates** or **reflects** off surfaces	8

		• Mirror object can not to move past foreground boundaries	
[7]	Gameplay	Simple collision detection & resolution (e.g. between square sprites) • Light ray collision detection between mirrors, surfaces, and endzones using axis-aligned bounding boxes	8
[8]	Stability	Stable framerate and minimal game lag. • Will be tested as part of QA tickets	4
[9]	Stability	No crashes, glitches, or unpredictable behaviour. • Will be tested as part of QA tickets	4

[10]	Software Engineering	Test plan - a list of player or game actions and their expected outcomes.
		• On game startup, level 1 loads in
		• Player click mirror, selects mirror
		  ○ Dragging inside mirror will translate the mirror
		  ○ Dragging outside mirror will rotate the mirror
		• On player successfully redirecting light rays to endzone, level finishes
			4

[11]	Reporting	Bug list (preliminary) - Google Sheets or Microsoft Excel spreadsheet.	4
[12]	Reporting	Demonstration video (3 min. max) showcasing assignment required and creative features.	8

Milestone 1 <u>Creative</u> Elements:

- Basis Physics: Inelastic Collision - Light rays bouncing off of mirror
- 2D Dynamic Shadows - Light rays illuminate the map as they travel

Week 1

- **[1] [2] Add basic graphic rendering for Level 1.**
- **[4] Implement the movement control for mirrors.**
- **[5] Add light rays periodically firing throughout the level.**

- **[3] [6] [7] Add collision handling for light rays against boundaries and mirrors.**

Week 2

- **[8-12] QA, Testing and reporting**
- **Players should complete the level when light rays hit the end zone.**
    - **Add VFX for stage completion**
- This timeline may be modified based on how much progress is made by Milestone 1

Milestone 2: Minimal Playability

#	Category	Task Breakdown	Marks
[1]	Improved Gameplay	Game logic response to user input. • Highlight selected environment objects • Play SFX on selection, translation, rotation	15.0
[2]	Improved Gameplay	Sprite sheet animation • Sprites for light ray, its moving animation, its dissipation animation • Animations for movement of flies and the light-loving turtle	15.0
[3]	Improved Gameplay	New integrated assets. • Sprites for objects, such as mirrors, blackholes, flies, the light-loving turtle, etc.	10.0
[4]	Improved Gameplay	Mesh-based collision detection. • Implement the interaction between the turtle and the irregularly-shaped lever (example on page 7)	10.0

[5]	Improved Gameplay	Base user tutorial/help.
		• Tutorial is integrated throughout levels as new elements are added
		• Focus on using bare minimum text:
		  ○ “Use mouse”
		  ○ “Click here”
			5.0

		○ “Drag here”	
[6]	Improved Gameplay	Frames-per-second (FPS) counter in the game window title or on the screen.	5.0
[7]	Playability	2-minutes of non-repetitive gameplay ● Will be tested as part of QA tickets	5.0
[8]	Stability	Minimal lag (as demonstrated by the frame rate counter ● Will be tested as part of QA tickets	2.5
[9]	Stability	No crashes, glitches, or unpredictable behaviour. ● Will be tested as part of QA tickets	2.5
[10]	Software Engineering	Updated test plan - updated list of player or game actions and their expected outcomes.	2.5
[11]	Reporting	Updated bug list - includes open and closed bugs.	2.5
[12]	Reporting	Demonstration video (4 min. max) showcasing assignment required and creative features.	5.0

Milestone 2 Creative Elements:

- **2D Dynamic Lighting**
- **Complex physical interactions with the environment via blackhole**

Week 1

- [1] [2] [3] Implement and integrate assets and game logic response (SFX and environment highlights)
- [5] Implement tutorial
- [4] Implement the irregular-shaped lever and the turtle colliding with it
- If ahead, work on a level editor to streamline level design

Week 2

- [6] Implement FPS Counter
- Implement both creative elements (2D Dynamic lighting and complex physical interactions with blackhole)
- Testing, bug-patching, demonstration video

- Make at least 3 separate levels, one of them is a tutorial, one of them with turtle and lever, and one of them with a black hole

Milestone 3: Playability

#	Category	Task Breakdown	Marks
[1]	Playability	5 minutes of non-repetitive gameplay • Will be tested as part of QA tickets • Complete enough levels so that the player has a steady stream of new content to play with for the duration of the 5 minutes.	15.0
[2]	Robustness	Memory management • Play the game for an extended period of time, attempting and completing multiple levels in one session. Run a profiler to make sure there are no memory leaks during this process.	5.0
[3]	Robustness	Handle all user input • Test to make sure valid mouse and keyboard inputs respond accordingly in the game (rotating mirrors, translating objects, interactions with UI elements), and invalid inputs don't crash the game. • Handle alt-tabbing and minimizing windows.	5.0
[4]	Robustness	Real-time gameplay • Play through levels with a profiler to keep track of frame rate and note any lag spikes. • If there are any performance bottlenecks found, use the profiler and debugger to track and fix them.	5.0

[5]	Stability	Prior missed milestone features & bug fixes. Consistent game resolution. • Run game on different monitors and resolutions to ensure game has a consistent experience across them (e.g. no cut off levels or UI elements) No crashes, glitches, unpredictable behaviour. • Will be tested as part of QA tickets	20.0
[6]	Software Engineering	Updated test plan - updated list of player or game actions and their expected outcomes.	2.5
[7]	Reporting	Updated bug list - includes open and closed bugs.	2.5
[8]	Reporting	Demonstration video (5 min. max) showcasing assignment required and creative features.	5.0

Milestone 3 Creative Elements

- **Physics-Based Animation**: Add physics-based water animation, with potential refraction interaction with light.
- **Swarm behaviour**: Implement fireflies (see Advanced Technical Elements) using Boids.

Week 1

- [2] [3] [4] Ensure game is robust (no memory leaks, performance bottlenecks, handles user inputs)
- [5] Complete and polish any remaining features from the last milestone, or fix any bugs introduced in the last milestone. Test the game on different monitors/resolutions. Continue testing the game for crashes/glitches.

Week 2

- [1] Ensure there is at least 5 minutes of non-repetitive gameplay
- [6] [7] [8] QA, Testing, and Reporting. Update bug list for sprint. Update test plan.

Milestone 4: Final Game

Week 1

Final Gameplay Integration

- Implement advanced gameplay features (black holes, beam splitters, mini-sun, turtle interactions).
- Test the advanced gameplay features to ensure they work as intended.

Playability Testing

- Ensure gameplay is engaging and non-repetitive for at least 10 minutes.
- Test level progression for smooth difficulty scaling.

UI/UX Finalization

- Polish menu screens.
- Finalize level selectors.
- Refine in-game feedback systems (hints, tooltips, etc.).

Stability & Performance Optimization

- Optimize memory usage.
- Improve game performance across different hardware (PC, mobile, etc.).

Bug Fixes & QA Testing

- Fix major bugs encountered during development.
- Conduct playtests to find and address additional glitches.

Documentation & Bug List Update

- Update test plans and documentation.

Week 2

Advanced Feature Testing

- Refine and test advanced gameplay mechanics (black holes, rays, beam splitters, swarm AI, turtle behavior).
- Ensure balance and smooth interaction between these elements.

User Playtesting

- Conduct user playtesting sessions.
- Collect feedback to balance difficulty and improve tutorials based on player input.

Final Bug Fixes & Stability Checks

- Fix any remaining bugs reported during playtesting.
- Ensure the game has no crashes or major glitches.

Final Asset Polishing

- Finalize and polish all visual assets (sprites, animations, lighting effects).

Final Testing & Optimization

- Optimize the game for memory, framerate, and resolution across various hardware setups.
- Test for performance stability.

Final Demo Video Creation

- Record key gameplay moments showcasing major features.
- Edit and finalize the demo video.