# ANGRY BIRDS

# ORE AJIBADE

ANALYSIS

# Outline

My project will be a recreation of my childhood favourite (and very popular) game called "Angry Birds". Angry Birds is a classic 2000s game involving flinging projectiles (Birds) at various structures containing targets (pigs) to hurt them sufficiently. The player is restricted from advancing to the next level unless all the targets have been eliminated. Though eliminating the game is the primary target, the player is also further meant to cause as much destruction as possible to gain as many points as possible. The elimination of targets and the amount of destruction caused in the process will add up to form the final score.

In the game, the entities included will be: projectiles (birds) to launch at the target; the target (pig) to be the aim of the projectile; projectile launcher (slingshot) to launch the birds, breakable and unbreakable obstacles (blocks) to make the game difficult. Whereas, objects visible to the player only will be the score indicator to show the current score and the stored highscore; bird (projectile) count to show how many birds the user has left, and a menu button to pause and play the game.

# Stakeholders

The original Angry Birds game is a mobile game, however, I have decided to make a PC adaptation of the game ( If time permits, I can recreate the game in mobile for more convenience). Therefore, this means this game is suited for the casual PC gamer, a person that takes the time out of his day to play PC games. Furthermore, due to the casual, puzzle solving nature of the game, the target demographic will be between mostly 10 - 22 years of age because this the optimal PC gamer age according to Statista.

This is why I have chosen Feyisayo Zollner as a representation of my target audience. He is a seventeen-year-old student in my computing class that is interested in video games. We have also both enjoyed playing Angry Birds together in the past and he will be delighted to see a PC version. Since we are in the same computing class, I will also have regular contact with him.

# How this Problem can be Solved by Computational Methods

The game can ideally be solved by computers due to how easy it is to abstract unnecessary variables and implement computational algorithms to simulate a certain game scenario.

## Thinking Abstractly

Abstraction is the process of removing unnecessary details for simplicity purposes. My game isn't a based fully in a realistic scenario, therefore, abstraction is required. The following are use cases of abstractions in my Angry Birds Game:

* Remove unnecessary variables like Air resistance.
* Add a bright, user friendly, cartoony environment to make the game visually appealling
* Restrict the game to 2 dimensions
* Add a bar at the top and bottom to show information like the number of birds remaining or number of points scored

## Thinking Ahead

Thinking ahead involves identifying the preconditions of a system like the inputs, outputs and reusable components. This is a way of looking ahead and knowing what you want. It allows you to plan a bit giving the solution of your problem a bit of structure.

* A points system will be utilized to give the user feedback on their performance in the game.
* Each game entity will be written as an object with different methods so it can easily be re-used in different components of the code
* Due to the nature of controlling a projectile launcher (slingshot), the user will be advised to make use of a mouse to play the game effectively
* I plan to use the Unity engine to simulate the physics because of its open source nature
* Due to my use of the Unity Engine, I plane to use the Unity IDE to develop my game

## Thinking Procedurally

A game as complex as Angry Birds will be required to be broken into many parts to simplify the process of making it. I have decided to make each entity in the game an object with a bunch of methods and attributes.

I have broken down Angry Birds into five main areas to ensure smooth gameplay and user interaction. These areas focus on the essential elements, from how the game progresses to how players control the birds and score points:

1. **GAMESTATE:** This area defines the various phases of the game, such as:

* **Menu:** Where the game begins, allowing the player to start a new game or review instructions.
* **Playing:** The primary state where the action occurs as the player aims and releases birds.
* **Paused:** Halts the game, providing a break when necessary.
* **Over**: Indicates whether the player has won or lost after all pigs are defeated or attempts exhausted.

2. **PLAYER:** This section handles the actions controlled by the player, such as:

* **Aiming:** The player adjusts the bird’s trajectory using the slingshot.
* **Release:** The player lets go of the slingshot, launching the bird toward the pigs.

3. **ENTITIES:** These are the in-game objects like birds, pigs, and blocks, which react to player actions:

* **Birds:** Can be in an unusedState (ready to be launched) or usedState (launched and interacting with blocks or pigs).
* **Pigs:** Can exist in a normalState (undamaged) or poppedState (eliminated after being hit).
* **Blocks:** Remain in an unBrokenState (intact) until hit, transitioning into a brokenState (destroyed).

4. **CONTROL:** This section manages how the player interacts with the game, including:

* **AimingControl:** Adjusts the slingshot’s direction and power.
* **LaunchControl:** Releases the bird from the slingshot.
* **Pause:** Freezes gameplay.
* **Restart:** Resets the current level, allowing a fresh attempt.

5. **SCORE:** This is the system that tracks and rewards player performance:

* **Win:** Achieved by defeating all pigs and achieving the HighScore.
* **Loss:** Occurs if the player fails to clear all pigs.
* **LevelScore:** Determined by the number of blocks destroyed, pigs popped, and remaining birds.

This structure ensures the game flows smoothly, with each area working together to create a fun and challenging experience for players.

## Thinking Logically

Thinking logically involves identifying points in which decisions will be made throughout the game. As the game is running, there will be a constant iteration searching for the following conditions:

* The "menu" state will determine if the game is paused or playing
* If a pig is hit, a decision if made to award the player more points than if a regular block
* The number of birds and number of pigs left will be continuously monitored to determine if the game should transition to a “game over” or “win” state. For instance, if there are no birds remaining but targets are still present, it results in a game over.

## Thinking Concurrently

Being a game, a lot of processes will be required to run at the same time to give real time updates to the user. Processes such as:

* Audio effects like bird launches, pig squeals, and block destruction occur in sync with the game’s visuals, ensuring that sound is triggered concurrently with corresponding events.
* The game constantly calculates the player’s score based on the number of pigs popped, blocks destroyed, and remaining birds. This score calculation is happening in real-time, even while the player is still launching birds.
* As the game progresses, the game continuously updates the states of objects (e.g., birds, blocks, and pigs). Birds switch between unused and used states, pigs between normal and popped states, and blocks between unbroken and broken states, all happening independently and concurrently.

## Conclusion

The earlier examples have shown that the characteristics of my problem can be addressed using computational methods, making it ideal for a computer program. These methods help structure the problem, making it easier to solve and thus well-suited for a computer-based solution.

# Research

## Game Comparisons

To properly research on how the game is meant to look like, I decided to examine a fairly similar game to what I want to create, Angry Birds 2 and Angry Birds Classic.

Angry Birds Classic is a popular puzzle game developed by Rovio Entertainment. In this game, players use a slingshot to launch various birds at green pigs stationed in or around different structures. The goal is to destroy all the pigs on the playing field. Each bird has unique abilities that can be used strategically to dismantle the pigs’ defenses. This is how I aspire the game to look like. This is a screen shot of a classic level in the game:

A video game screen with grass and plants

Description automatically generated

From the game, I have decided to implement the following features into my game:

**Gameplay**

Pause

Score

Pig

Background

Bird

A video game screen with grass and plants

Description automatically generated

Slingshot

High Score

Incoming Birds

Blocks

## A video game screen with a cartoon character Description automatically generated

Updated Slingshot

Attacking Bird

**Pause State**

A video game screen with trees and text

Description automatically generated

**Win State**

A screenshot of a game

Description automatically generated

**Level Selection**

A screenshot of a game

Description automatically generated

**Splash Screen**

A cartoon bird with a red round object

Description automatically generated with medium confidence

Next is Angry Birds 2, Which is a newer version of the Angry Birds Classic. They both have the same basic features except a radical change in design. The main reason I looked at this game for inspiration was because I thought some design changes like the card system worked better than the classic implementation incoming birds.

**Gameplay**

A screenshot of a video game

Description automatically generated

Card System

**Level Selection**

A screenshot of a video game

Description automatically generated

**Splash Screen**

A video game screen with angry birds

Description automatically generated

## Interview with Stakeholder

### Plan

The main points I hope to get from this interview include:

* Basic Requirements for the game
* Design Changes to the game
* Changes in the physics of the game

Expanding on these main points, I have come up with questions under the topic of these main points:

**Basic Requirements for the game**

* How many different birds should be made for the bare requirements?
* Should these birds have unique abilities like in the classic game?
* What other features would you want the birds to have?
* Is a score panel a key element for the game, or do you think the game could work without it?
* How important is a high score panel to keep players engaged?
* Should this be a feature, and how would you like it displayed (global leaderboard, personal best)?
* What must absolutely be included for this to feel like a complete Angry Birds game?

**Design Changes to the game**

* Having played Angry Birds Classic before, what changes will you wish to have implemented design wise?
* Should the game be more realistic, or should it keep its original cartoonish style?
* How important is the look of the birds and pigs in enhancing the player's experience?
* Would you like the backgrounds and levels to be more interactive or dynamic in nature?
* What sort of animations or visual elements would you want to see?
* Should there be more exaggerated reactions when pigs are popped or blocks are destroyed?
* Should the game feature more detailed environments, weather effects, or lighting changes?

**Changes in the physics of the game**

* Should the game physics be more realistic, or do you prefer the exaggerated, cartoonish physics from the original?
* If so, what would you like to implement?
* Should bird flight paths be affected by environmental factors (wind, etc.) to add realism or new challenges?
* Should blocks break more realistically based on material type (e.g., wood vs. stone)?
* Should birds interact differently with various obstacles or environments (e.g., water, ice, etc.)?

### Interview Script

**Q:** To start, how many different birds do you think should be made for the bare requirements of the game?

**A:** I think just adding "Red", the original bird with no special abilities, would be enough to start. We can always introduce more birds with abilities as the game progresses.

**Q:** Should these birds have unique abilities like in the classic game?

**A:** Yes, as the game progresses, introducing birds with unique abilities will definitely add variety and keep it interesting.

**Q:** What other features would you want the birds to have?

**A:**  Keep them mostly the same as the original. People already like the way they are, and it worked well before, so no need to reinvent the wheel there.

**Q:** Is a score panel a key element for the game, or do you think the game could work without it?

**A:** It’s nice to have, but not essential for the bare bones of the game. The main thing is that I can fling birds at pigs and eliminate them. The score is more like motivation to replay a level or compare with friends. If you are implementing the score panel though, I think that you should make it so that you can compare scores on different levels with people all over the world or at least your friends.

**Q:** How important is a high score panel to keep players engaged?

**A:** It’s not essential but would be awesome to have. I used to love comparing my high scores with friends, and it definitely made the game more competitive. But first, focus on the core gameplay, then add this.

**Q:** Should this be a feature, and how would you like it displayed? (Global leaderboard, personal best?)

**A:** Both would be great! It’s always fun to see a personal best, but having a global leaderboard adds that extra competitive edge.

**Q:** What else must absolutely be included for this to feel like a complete Angry Birds game?

**A:** It just needs to be a cool puzzle game where I fling birds at pigs and deal with a few obstacles in between. That’s what makes Angry Birds great for me.

**Q:** Having played Angry Birds Classic before, what changes would you wish to see design-wise?

**A:** I think the new "Angry Birds 2" is a bit too animated for my liking. I prefer the simplicity of the original. But I do like how in "Angry Birds 2" the birds are shown in a bar on the screen—it’s nice to see all the birds lined up like that rather than how it is in the classic.

**Q:** Should the game be more realistic, or should it keep its original cartoonish style?

**A:**  Keep it cartoonish. I don’t want it looking like the Minecraft movie

**Q:** How important is the look of the birds and pigs in enhancing the player's experience?

**A:** Very important. I think they already look good, so no need to change their art style much. The original design worked really well.

**Q:** Would you like the backgrounds and levels to be more interactive or dynamic in nature?

**A:** Yeah, a static background could look boring after a while. Maybe a background with trees and some movement to make it more interesting.

**Q:** What sort of animations or visual elements would you want to see?

**A:** Keep the animations similar to the classic game. Nothing too fancy.

**Q:** Should there be more exaggerated reactions when pigs are popped or blocks are destroyed?

**A:** Nah, keep it simple. The reactions in the classic game were fun without being over the top.

**Q:** Should the game feature more detailed environments, weather effects, or lighting changes?

**A:** It would be nice, but it risks making the game look too much like Angry Birds 2. So, be careful with adding too many effects.

**Q:** Should the game physics be more realistic, or do you prefer the exaggerated, cartoonish physics from the original?

**A:**  I like the cartoonish physics, but the only thing I didn’t like was how the birds lost all their momentum when hitting a block. It gets kinda annoying. Maybe tweak that a bit.

**Q:** Should bird flight paths be affected by environmental factors like wind, or would that complicate things?

**A:** No, please don’t add wind resistance. It would make things unnecessarily difficult.

**Q:** Should blocks break more realistically based on their material type, like wood versus stone?

**A:**  Yeah, I think that would be cool. Blocks should definitely break differently depending on their material.

**Q:** Should birds interact differently with various obstacles or environments (e.g., water, ice, etc.)?

**A:** That would be harder to implement, but yeah, it would be good to see birds interact differently with different materials.

### Review

After discussing the key elements, the stakeholder and I agreed that simplicity is essential for the game’s core design, with the main objective being a puzzle-solving experience where the player uses birds to eliminate pigs.

For the basic requirements, the stakeholder expressed that adding just the original "Red" bird with no special abilities would suffice to start, ensuring the focus is on delivering the fundamental mechanics of the game. The idea is to keep the game simple at first, then progressively introduce more features, such as birds with unique abilities and additional challenges as the game advances.

Regarding gameplay features, the stakeholder felt that tracking progress, such as scoreboards and high score panels, isn't crucial for the game's initial version. The primary focus should be on allowing players to complete the core objective of eliminating pigs using birds. However, once the basic gameplay is polished, the inclusion of a score panel or leaderboard would be a great addition to enhance competitiveness and replayability.

In terms of visual design, the stakeholder preferred keeping the cartoonish style of the original Angry Birds, avoiding an overly animated or realistic look. The key takeaway here was that the birds, pigs, and overall aesthetic should stay true to the classic game, with minor tweaks like having backgrounds that are slightly more dynamic to prevent the environment from feeling static.

When discussing physics, it was agreed that the exaggerated, cartoonish physics from the original should remain, with the only requested change being to adjust how birds interact with blocks to prevent them from losing all momentum when hitting obstacles. This small tweak would improve gameplay while still keeping the game fun and familiar.

In conclusion, the stakeholder emphasized that the game should start simple, focusing on its core mechanics of solving puzzles by eliminating pigs. Additional features, such as more birds, complex scoring systems, and enhanced visuals, can be scaled up later, once the core gameplay has been fine-tuned.

# Success Criteria

As per the feedback from the interview, I am going to divide my success criteria into parts; The Bare Essentials and the Extra Features.

**Bare Essentials**

|  |  |  |
| --- | --- | --- |
| **Index** | **Feature** | **Descriptions** |
| 1 | Slingshot | This is the main tool used by the player to launch birds at the structures. It adds a layer of strategy as players must aim and control the power of each shot. |
| 1.1 |  | Slingshot successfully launches birds |
| 1.2 |  | Slingshot animates when interacted with by the player |
| 2 | Bird | The bird acts as the projectile in the game, with each type having different abilities. This creates variety and adds depth to the gameplay. |
| 2.1 |  | Bird interacts with blocks by breaking them or not |
| 2.2 |  | Bird interacts with pig by popping them when they are hit with a certain force |
| 2.3 |  | When bird gets launched by slingshot, bird obeys the laws of physics (ignoring unnecessary factors like Air resistance) |
| 2.4 |  | Number of Birds reduces when a target bird has been launched |
| 3 | Blocks | These form the structures that protect the pigs. Different types of blocks (wood, ice, stone) behave uniquely, making the game more challenging. |
| 3.1 |  | Basic Physics for Blocks; Physics governs how blocks fall and break, which creates more dynamic and engaging gameplay as players have to predict how structures will collapse. |
| 4 | Pig | The main target in each level. Eliminating all pigs is necessary to complete the level. Their placement within structures makes strategic destruction essential. |
| 4.1 |  | Pig pops when hit with a certain force |
| 5 | Background | This sets the visual tone of the game and helps differentiate levels. While not crucial to gameplay, it adds to the game’s aesthetic and experience. |
| 6 | Gameplay | The general gameplay of the game |
| 6.1 |  | Simple sound effects, such as the bird being launched or blocks breaking, enhance the game's feedback and immersion. |
| 6.2 |  | The core gameplay mechanic. Successfully aiming the bird at the pigs and causing destruction is the main objective of the game. |
| 6.3 |  | Camera moves to bird of focus at all stages in the game |
| 6.4 |  | A simple Splash Screen that plays when the game begins |
| 6.5 |  | A simple Game Over screen that activates if the player loses a game |
| 6.6 |  | A simple Game win screen that activates if the player wins a game |

**Extra Features**

|  |  |  |
| --- | --- | --- |
| **Index** | **Feature** | **Descriptions** |
| 1 | Menu | The central hub of the game where players can start a game, view options, and navigate to other sections. It provides an easy way to access different game modes and settings. |
| 2 | Pause | Allows the player to pause the game and take a break, offering the ability to resume or quit without losing progress. Essential for longer gameplay sessions. |
| 3 | Score System | Allows players to track their best performances across different levels, adding a competitive element as players aim to beat their previous records or compete with friends. |
| 4 | High Score System | Allows players to track their best performances across different levels, adding a competitive element as players aim to beat their previous records or compete with friends. |
| 5 | Powerups | Special abilities that can be used to enhance gameplay, such as making birds stronger or giving them extra abilities. These add excitement and variety to levels. |
| 6 | Levels | The individual stages of the game, each with its own unique layout of pigs, structures, and challenges. Players progress by completing each level. |
| 7 | Level Viewer Screen | Displays all available levels for the player to select and shows which levels have been completed or unlocked. This provides a sense of progression. |
| 8 | Loading Screen | Briefly shown when levels or game elements are being prepared. This maintains immersion by keeping players informed during transitions between gameplay and menus. |
| 9 | Menu Page | The primary screen that provides navigation options such as starting the game, viewing achievements, or adjusting settings. It serves as the game’s entry point. |
| 10 | Different Types of Birds | Adds variety to the gameplay, with each bird having its own unique ability. This challenges the player to use strategy in selecting the right bird for the right situation. |
| 11 | 10 Levels | A baseline number of stages to provide enough content for players to engage with the game without being too overwhelming in early development. Provides a balance between variety and scope. |
| 12 | Extra Block Types | Additional block materials (such as metal or glass) that behave differently when hit by birds, adding complexity and strategy to how players approach each level. |

# Limitations

The stakeholder mentioned that if we implement the score panel feature, that it should be available to compare with other players around the world. Though this would be a really good feature to implement, I noticed that implementing a feature like this will involve finding a place to store the user data online, and socket request which is beyond the scope of my project because there isn't enough time to implement this

# Software and Hardware Requirements

|  |  |
| --- | --- |
| **Requirement** | **Justification** |
| **Hardware** | |
| Processor: 2GHZ+ | Will need around 2GHZ in order to do the graphics and animations of the game as well as the tasks. |
| Memory: 1GB+ RAM |  |
| Graphics: 12\*MB+ | You will require minimum graphics to run this game. |
| Hard Drive: No more than 250MB |  |
| Mouse | Required to launch the bird |
| Monitor | Way of being able to view the game and play it. |
| Speaker | To hear Sound effects of the game |
| **Software** | |
| Operating system: WINDOWS 7/10/11 , macOS Mojave 10.14+, Linux Ubuntu 20.04 | These are the operating systems that Unity will work with |

DESIGN

# Systems Diagram

As recommended by my stakeholder, I have chosen to keep the project simple for now before implementing more features. After my analysis, I did some further research into how the game can be broken down into more concise components, to reduce the problem of creating the game into more achievable tasks.

This hierarchy ensures clarity in structuring the game's components, states, and interactions, making it easier to develop and expand upon the Angry Birds game. By employing a top-down module design, I can effectively identify the necessary components before beginning the coding process. This approach allows me to decompose the problem into smaller, more manageable parts, enabling a focused examination of each segment. Utilizing the computational thinking skill of "thinking ahead," I can determine the required inputs and outputs. Additionally, "thinking logically" helps me pinpoint where decisions are needed and understand their impact on other parts of the solution. This structured methodology ensures that my working solution is both systematic and efficient, guiding me towards a well-organized and effective outcome.

The Angry Birds object serves as the main container for the entire game, encapsulating its three major states:

1. **Gameplay:** The interactive phase where the player engages in launching birds and targeting pigs.

2. **Game Over:** Triggered when all available birds are used but some pigs remain active.

3. **Game Win:** Triggered when all pigs are eliminated.

### Game Play

This state represents the core interactive gameplay, comprising:

#### Background

1. **Sky:**

* Represents the top portion of the screen. The sky provides aesthetic appeal and acts as a visual backdrop.

2. **Ground:**

* Forms the base where structures, pigs, and the slingshot are placed.
* Ground physics ensure birds, blocks, or pigs that fall onto it behave realistically.

3. **Camera Movement:**

* The camera follows the currently active bird. This ensures the player maintains focus on the action and enhances immersion.

#### Entities

These are interactive elements within the game world:

1. **Birds:**

The primary projectile, with distinct states:

* + **Not In Use:** Birds waiting in a queue for their turn to be launched. A subtle idle animation and sound effect could enhance realism.
  + **In Sling Shot:** Birds being aimed and prepared for launch. Includes a sound effect (e.g., a stretching sound).
  + **Used:** Birds that have been launched and either hit a target or missed.
  + **No. of Available Birds:** Tracks how many birds are left for launching.

2. **Pigs:**

The targets in the game, with two primary states:

* + **Active:** Pigs are alive and require elimination. They might display idle animations and sound effects.
  + **Popped:** Pigs disappear with an animation and a “pop” sound when hit with sufficient force.

3. **Blocks:**

Structures that protect the pigs. They can be destroyed with force and have the following characteristics:

* + **Active:** Undamaged and in position.
  + **Broken:** Destroyed, typically via collisions with birds or other blocks.
  + **Broken-Sound:** A unique sound effect plays upon destruction.
  + **Types:** Rectangular and Square blocks offer variation in gameplay strategy.

4. **Sling Shot:**

The launching mechanism for the birds. States include:

* + **Inactive:** When not in use.
  + **Active:** When the player interacts with it to aim and launch a bird. Includes an “active sound” for stretching rubber.

### Game Over

This state signifies the end of the game if the player loses:

1. **Game Over Sound:** A single sound effect to signal failure.
2. **Game Over Text:** Displays a message such as “Game Over” on the screen.
3. **Restart Button:** Allows the player to restart the current level.
4. **Quit Button:** Provides an option to exit the game or return to the main menu.

### Game Win

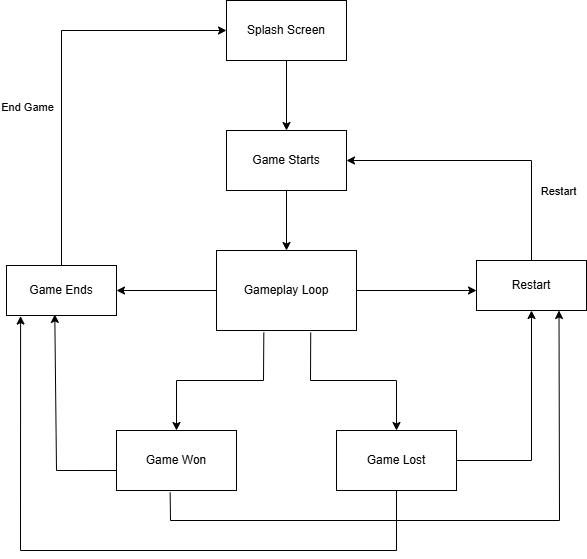
This state is triggered when the player successfully eliminates all active pigs:

1. **Game Win Sound:** Plays celebratory audio upon victory.
2. **Game Win Text:** Displays a congratulatory message.
3. **Restart Button:** Enables the player to replay the level for better performance.
4. **Next Level Button:** Progresses to the subsequent level (if implemented).
5. **Quit Button:** Allows the player to exit the game or return to the main menu.

# Game Progression Overview

To further decompose the problem, I have decided to go over how the player will progress through every game module. Because I have two development success criteria (Bare features and Advanced features), I have designed a progression overview that covers the scope of each of these criteria. To reiterate, Bare Features serve as the foundation for core gameplay while Advanced Features add complexity and enhance player engagement. A modular design approach enables incremental development and testing of each component.

## Bare Features Game Progression



### Game Start

* + The game begins with a simple splash screen.
  + Player clicks "Start" to enter the first level.

### Gameplay Loop

#### Camera Movement

* + The camera centres on the slingshot and active bird.
  + Tracks the bird during its flight, ensuring focus remains on the action.

#### Bird Launch

* + Player interacts with the slingshot to aim and launch the bird.

#### Slingshot Animation:

* + Stretches dynamically based on drag distance and angle.
  + Upon release, the bird obeys basic physics (ignoring air resistance).

#### Target Interaction

Birds collide with blocks and pigs:

* Blocks: May fall or break depending on the force applied.
* Pigs: Pop when hit with sufficient force.

#### Level Conditions

* + Player wins the level if all pigs are eliminated.
  + Player loses if all birds are used without clearing the pigs.

### Game End States

#### Game Win Screen

* + Triggered when all pigs are eliminated.
  + Displays a congratulatory message and a "Restart" button.

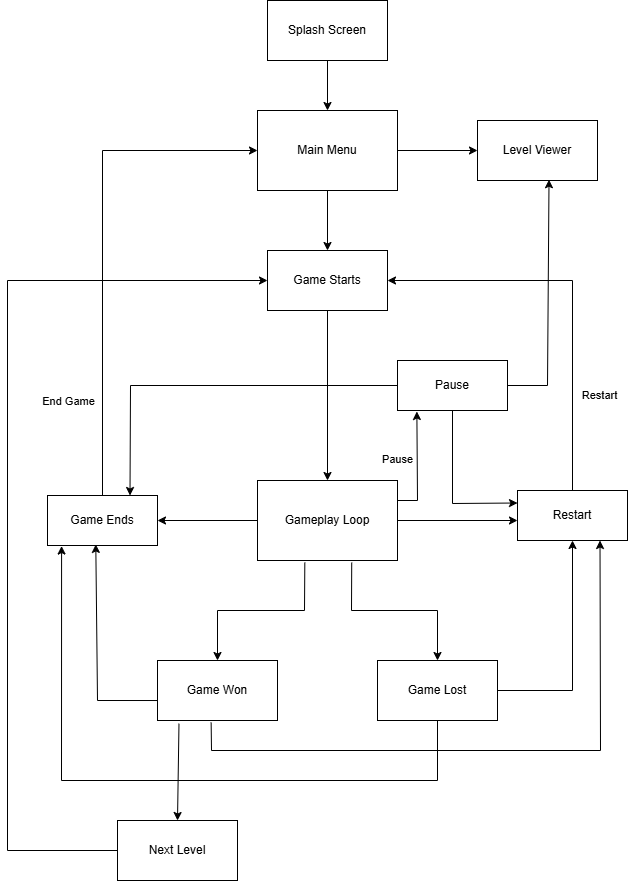
#### Game Over Screen

* + Triggered when birds are depleted, and pigs remain.
  + Displays "Game Over" and provides "Restart" and "Quit" buttons.

### Conditions

* + Win Condition: All pigs are popped.
  + Lose Condition: All birds are used, and pigs remain.
  + Restart: Reinitializes the current level.
  + Quit: Exits to the splash screen.

## Advanced Features Game Progression



### Main Menu

The player starts at a menu page with options:

* + Start Game: Launches the first level or the last unlocked level.
  + Level Viewer: Displays a list of available levels and their statuses.
  + Quit: Exits the game.

### Loading Screen

Briefly displayed during transitions between menu and gameplay, improving immersion.

### Gameplay Loop

#### Pause Feature:

* + Player can pause the game.
  + Options include "Resume," "Restart," or "Quit to Menu."

#### Score System:

* + Tracks points for each level based on performance (e.g., number of birds left, destruction).
  + Adds replay value as players aim for high scores.

#### Powerups:

* + Introduces temporary abilities for strategic advantage.
  + Examples include stronger birds, explosive impacts, or extra birds.

#### Camera Movement:

* + Enhanced tracking with zoom-in/zoom-out mechanics for better visualization.

#### Advanced Bird Types:

* Birds with unique abilities (e.g., split into smaller birds, explosive impact).
* Adds variety and strategy to the game.

### Levels

* Includes 10 unique levels with increasing complexity
* Level Viewer Screen tracks progress, showing completed and locked levels.
* Each level introduces different structures, pig placements, and challenges.
* Extra block types (e.g., stone, glass) behave differently under impact, increasing strategy.

### Game End States

#### Game Win:

* + Unlocks the next level if available.
  + Provides "Restart," "Next Level," and "Quit" options.

#### Game Over

* + Prompts replay with the same options as the basic version.

### High Score System

* + Tracks best scores per level across sessions, encouraging competition.

### Conditions

* Win Condition: All pigs are eliminated before using all birds.
* Lose Condition: All birds are used, and pigs remain.
* Next Level: Available if the current level is cleared.
* High Score: Stored if a new record is set.
* Powerup Usage: Players must manage powerups strategically, as their availability may be limited per level.

# Algorithms

### Gameplay Flow

This algorithm outlines the step-by-step process of a game flow. The game starts by displaying a splash screen and then takes the player to the current level. The game runs in a loop where it checks if the player wins the level. If the player wins, a game win screen is shown, and the player is given the option to play again. If the player chooses to play again, they can either proceed to the next level or end the game. If the player does not win the level, a game over screen is shown, and the player is given the option to play again from the current level or end the game. This flow is a structured way to manage game progression, player choices, and game state transitions.

START

Display Splash Screen

Take Player to Current Level

WHILE game is running

IF Player wins level THEN

Display Game Win Screen

IF Player wants to play again THEN

IF Player wants to go to next level THEN

Take Player to Next Level

ELSE

END GAME

ELSE

END GAME

ELSE

Display Game Over Screen

IF Player wants to play again THEN

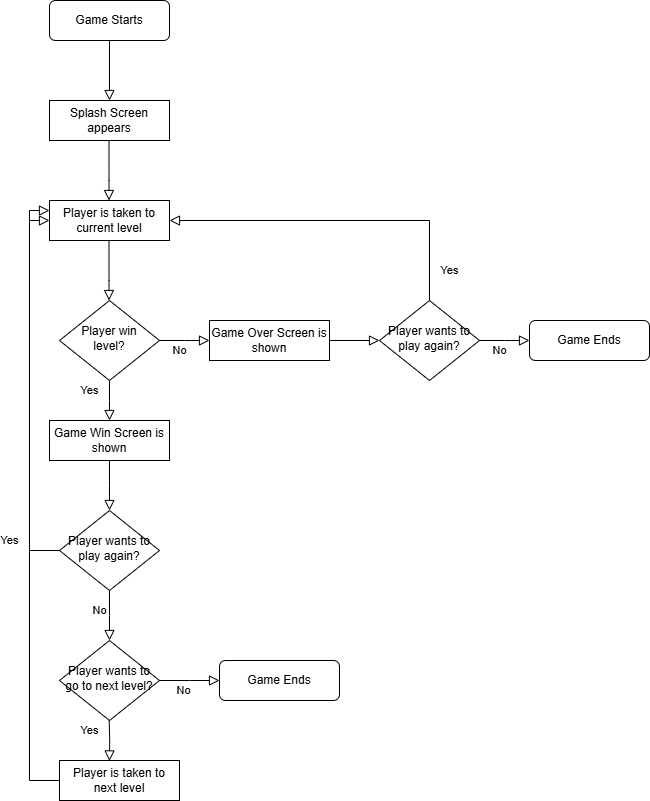
Take Player to Current Level

ELSE

END GAME

END WHILE

END



### Detecting if the Player wants to handle the Sling Shot

The process begins when the player left-clicks and holds the mouse button. The system checks if the click is close to the slingshot. If it is, the elastic bands stretch to the click location, and the next bird appears in the slingshot. A stretched sound is played, and the system waits for the player to release the hold. Once released, the initial bird speed is calculated based on the length of the slingshot stretch, and the angle of projection is determined. The bird counter is then reduced by one, and the bird is released according to the calculated speed and angle.

START

Player initiates action by left-clicking and holding

IF click is close to slingshot THEN

Stretch elastic bands to click location

Next bird appears in slingshot

Play stretched sound

WHILE player has not released hold

Slingshot stays in stretched state

END WHILE

Calculate initial bird speed based on slingshot stretch length

Calculate angle of projection

Reduce bird counter by 1

Release bird according to speed and angle

ELSE

Slingshot makes no action

END IF

END

A black and white image of a black background

Description automatically generated

### Detecting if a Block is Broken

The process begins with the block staying intact. During the game, the system continuously checks if the block is experiencing a force. If a force is detected, the system then checks if the force is larger than a predefined threshold. If the force exceeds the threshold, the block object is removed from the gameplay, and a breaking animation and sound are played. If the force is below the threshold, the block's speed and direction are calculated based on the laws of motion, and the block moves accordingly.

START

Block stays intact

WHILE game is running

IF Block is experiencing a force THEN

IF force is larger than threshold force THEN

Block Object is removed from the Gameplay

Play Block breaking animation

Play Block breaking sound

ELSE

Calculate speed and direction based on Laws of Motion

Block moves at calculated speed and direction

END IF

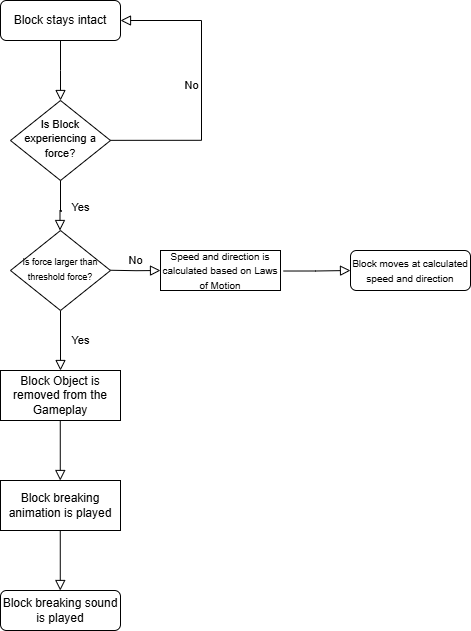
ELSE

Block stays intact

END IF

END WHILE

END



### Detecting if a Pig has been popped

Similar to the Block Breaking Algorithm, the process begins with the pig staying intact. During the game, the system continuously checks if the pig is experiencing a force. If a force is detected, the system then checks if the force is larger than a predefined threshold. If the force exceeds the threshold, the pig is removed from the gameplay, the pig counter is reduced by one, and a popping animation and sound are played. If the force is below the threshold, the pig's speed and direction are calculated based on the laws of motion, and the pig moves accordingly.

START

Pig stays intact

WHILE game is running

IF Pig is experiencing a force THEN

IF force is larger than threshold force THEN

Pig is removed from the gameplay

Reduce pig counter by 1

Play pig popping animation

Play pig popping sound

ELSE

Calculate speed and direction based on Laws of Motion

Pig moves at calculated speed and direction

END IF

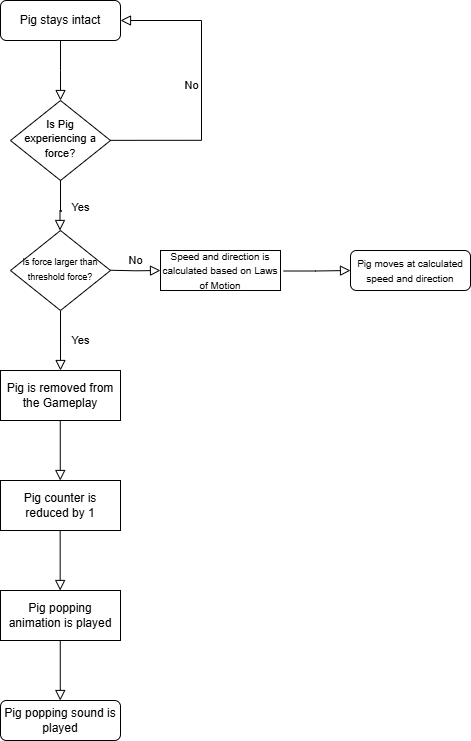
ELSE

Pig stays intact

END IF

END WHILE

END



### Detecting if a is Game Over

This algorithm outlines the game flow logic, focusing on the number of birds and pigs. The game starts by recording the number of birds and pigs and setting the variables `birdNo` and `pigNo` to the number of available birds and pigs left, respectively. During the game, the system continuously checks if `birdNo` is equal to 0. If it is, the “game over” screen is shown, and the game ends. If `birdNo` is not 0, the system then checks if `pigNo` is equal to 0. If it is, the game win screen is shown, and the game ends. If neither condition is met, the game continues. This process is crucial for managing game progression and determining the game's outcome based on the player's performance.

START

Game Starts

Record number of Birds and Pigs

birdNo = number of available birds

pigNo = number of pigs left

WHILE game is running

IF birdNo == 0 THEN

Show Game Over Screen

END GAME

ELSE

IF pigNo == 0 THEN

Show Game Win Screen

END GAME

ELSE

// Continue game logic

END IF

END IF

END WHILE

END

