Detailed Game Specifications: *This Penguin Flys!*

Course: *COMP 2659, Winter 2024*
Instructor: *Paul Pospisil*

Author: *George Rostek*
Last Modified: *April 8, 2024*

1 General Game Overview

This Penguin Flys! is a modified version of the original Flappy Bird game, featuring nearly identical mechanics to its predecessor. Fundamentally, Flappy Bird is a 2D side-scrolling game.

In the game of *This Penguin Flys!* players control a penguin's flight, aiming to navigate through pairs of icy posts emerging on a scrolling background from right to left. Points are earned by successfully guiding the penguin through the safe zone between oncoming posts, scoring one point for each successful pass. Players may navigate around posts but will not receive any points for doing so. The game concludes if the penguin contacts the ground or any part of the post object outside the designated safe zones (see Figure 5). The objective is to challenge players to navigate through as many posts as possible, striving for a new high score in each round. Initially, the penguin is positioned at coordinates $x = 220\, px,$ and $y = 172\, px$ (see Figure 3). The game commences when the player takes control of the penguin, or first presses the spacebar.

The player controls the penguin' flight by pressing the spacebar, altering the vertical, or y, component of the penguin's flight velocity with each press while leaving the horizontal, or x, component unchanged. Moreover, although velocity is a vector with both horizontal and vertical components in a 2D setting, the penguin only experiences a change of velocity in the vertical direction – the penguin is fixed on the horizontal x–axis, and only moves along the vertical y–axis. Each spacebar press sets the penguin's velocity to $-9\, px/frame$, and unless moving upwards, the penguin descends at $1\, px/frame$ due to simulated gravity. Flight can be maintained with well-timed successive presses of spacebar, while holding spacebar engages continuous lift.

Posts are considered a single object with two primary components: the top post, and bottom post. Post objects spawn at regular intervals each with a random height from the right edge of the screen; they traverse leftwards across the screen at a constant rate of $-4\, px/frame$, wrapping around the screen at a new random height only after moving completely past the left edge of the screen. This random height position is confined within a predetermined range — ensuring that passage for the penguin remains perpetually feasible without reliance on safe zones. The game maintains a maximum display of three post objects (see Figure 1).

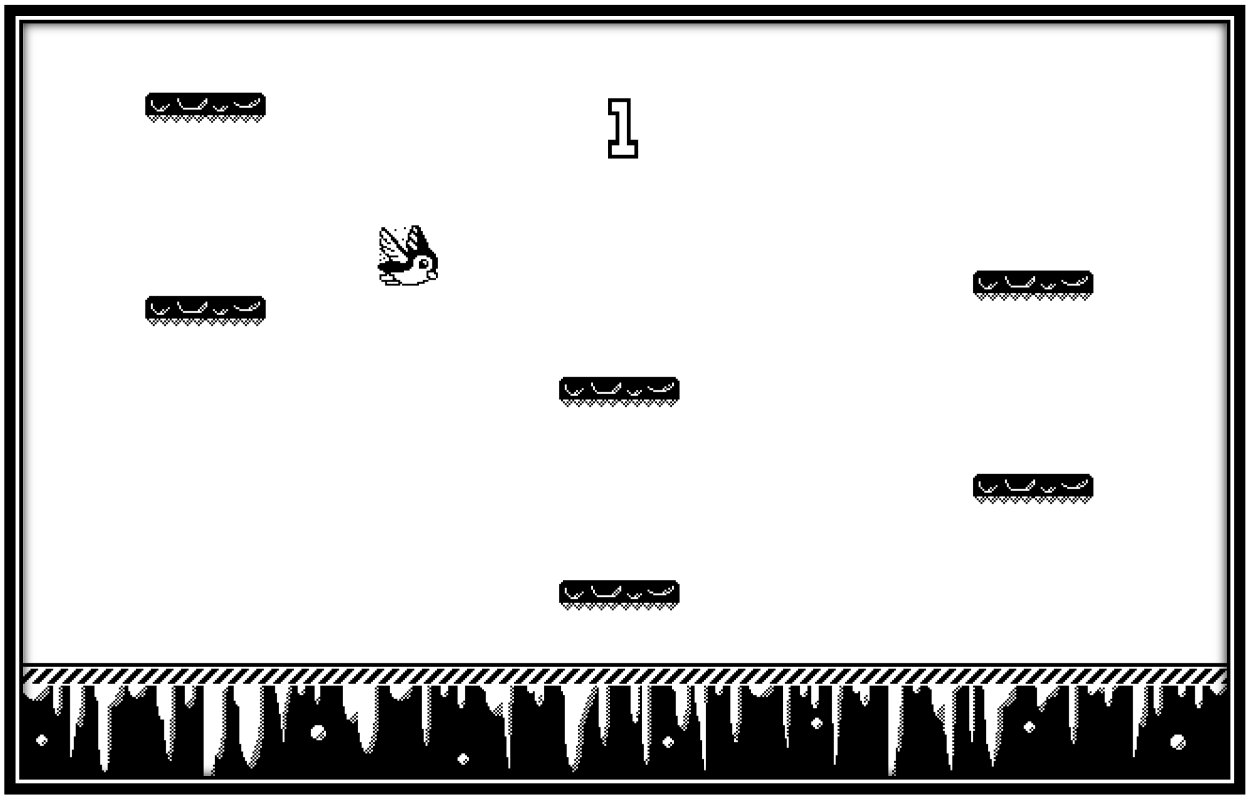

Figure 1: This screenshot is taken mid-game during the round state, displaying the penguin after successfully clearing the first set of posts. It highlights the game's focus on precision in navigating through successive obstacles.

2 Game Play Details for Core 1-Player Version

2.1 Start State

The game begins on a home screen which displays the title and logo of the game giving the user three options to choose from: normal, hard mode, or quit. Each of these options is given by an interactive digital button on the screen which can be clicked upon using the mouse.

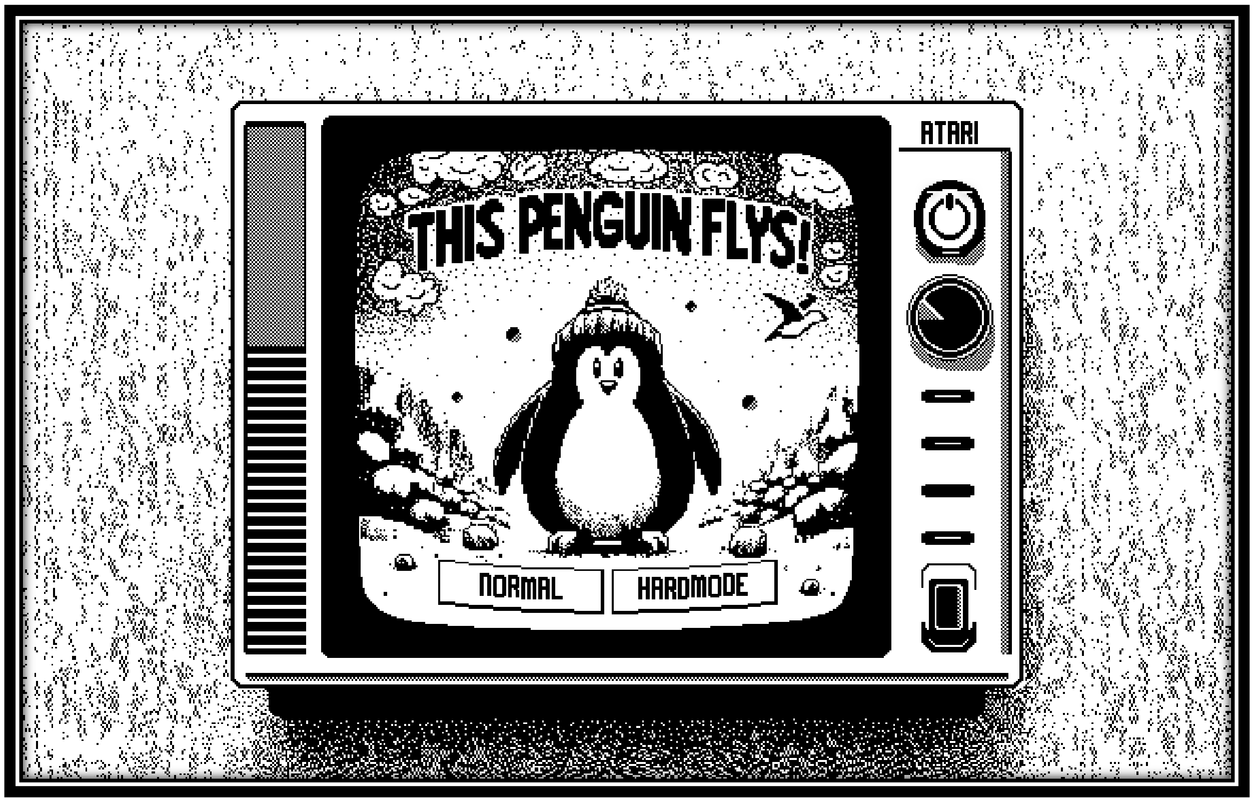

Figure 2: This screenshot is taken during the start state, immediately after loading the game. The player is given three options to choose from: normal, hard mode, or quit (power-off button).

2.2 Idle State

The idle state of each round consists of the following elements:

i. Penguin – positioned at coordinates $x = 220\,px$, and $y = 172\,px$ (see Figure 3). The penguin is in animated flight, oscillating slightly upwards and downwards as it flies; it is waiting for the player to take control with the initial press of the spacebar.

ii. Score – positioned at coordinates $x = 279/287/295\,px$, and $y = 40\,px$ (x position depends on the scores number of digits). Each round begins with the score set to zero.

iii. Ground – static image starting at coordinates $x = 0\,px$, and $y = 340\,px$, giving a reasonable amount of space for the penguin to fly.

Note: post objects are not present during the start state of the game and will only start appearing after the player first takes control of the penguin by pressing spacebar.

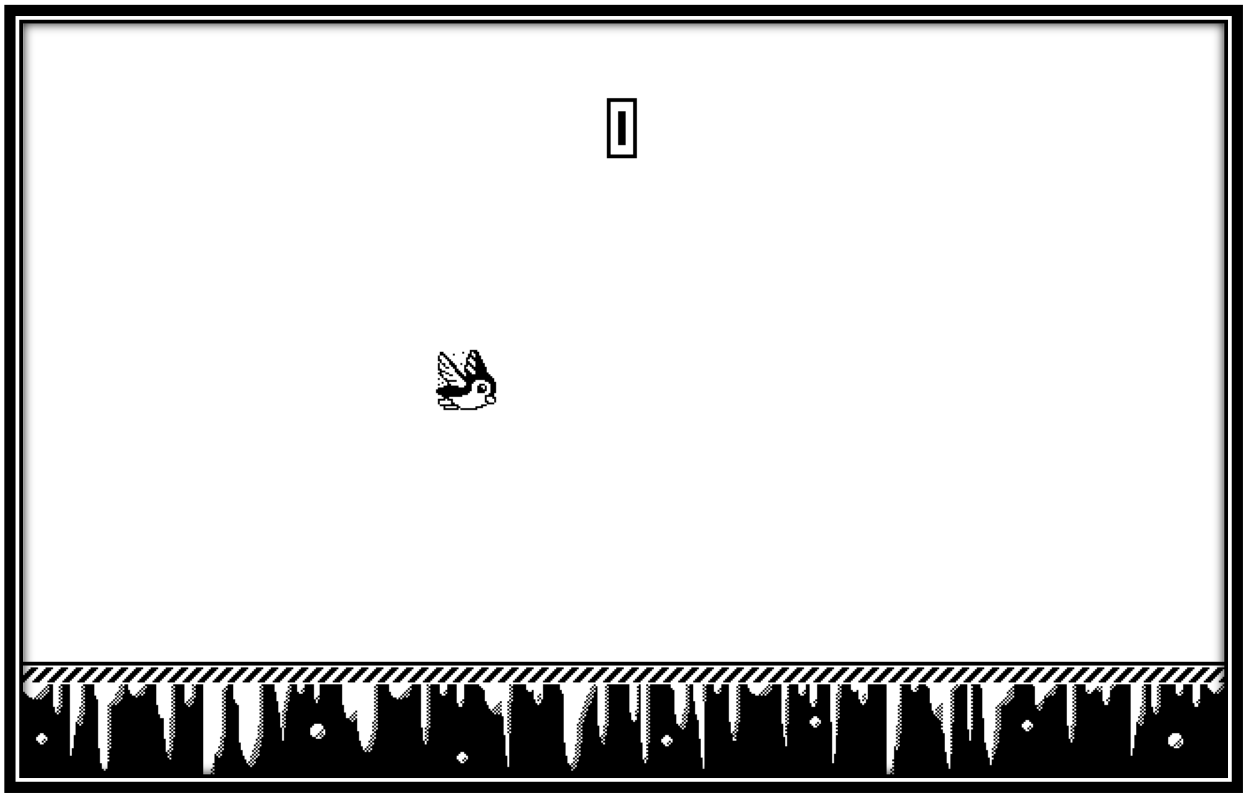

Figure 3: This screenshot is taken early-game during the idle state, before the player takes control of the penguin. Penguin is positioned at coordinates $x = 220\ px$, and $y = 172\ px$.

2.3 Final State

Each round ends on a final screen which displays the corresponding score for that round, giving the user two options to choose from: restart, or menu. Both options are given by an interactive digital button on the screen which can be clicked upon using the mouse.

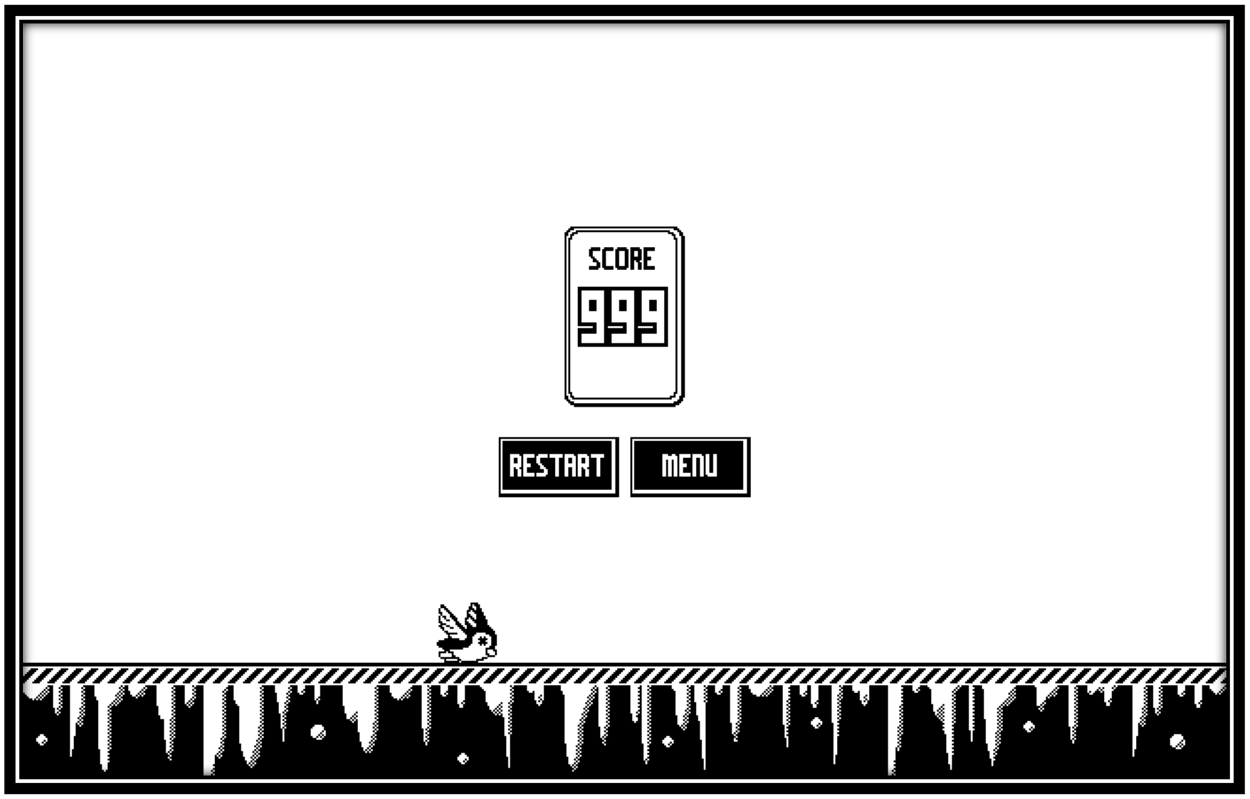

Figure 4: This screenshot is taken late-game during the final state, after the player has lost. The player is given two options to choose from: restart or quit.

Note: players can quit the game by returning to the main menu, or by pressing the escape key.

2.3 Objectives and Rules

In *This Penguin Flys!* players earn points by skillfully maneuvering the penguin between the safe zone of oncoming posts. Successful navigation through the gap between these posts results in the acquisition of one point per pass. The round concludes upon the penguin's contact with the ground or any part of the post object outside the designated safe zones. Additionally, players may navigate above or below post objects safely, but will receive no points for doing so.

The primary objective of the game is to challenge players to navigate through as many posts as possible, striving to surpass their own or their opponent's high score in each successive round. Winning the game is not achieved through traditional means; rather, victory is synonymous with achieving a personal best.

Once the round commences, the penguin descends at $1\ px/frame$, simulating the effects of gravity. The player controls the penguins' flight using the spacebar. A single press of the spacebar sets the penguins velocity to $-9\ px/frame$ once with each press – remember that upwards motion is negative. Ascent is accelerated with rapid presses of the spacebar, and the penguin descends when there is an absence of spacebar input due to simulated gravity.

Once the round has commenced, post objects emerge from the right edge of the screen at regular intervals, persisting as long as the player remains alive. These post objects move from right to left across the screen at a constant velocity of $-4\ px/frame$, and wraparound at a new random height once they completely move past the left edge of the screen. At any given moment, there will be a maximum of three different post objects visible on screen. Notably, the safe zone or gap within these post objects varies randomly with each spawn, adding an element of unpredictability to the challenge, but although the height of the safe zone varies, the size of the gap does not.

It is important to highlight that when the penguin contacts the top edge of the screen, the round will not conclude; instead, the penguin will seamlessly pass through the edge and continue to play off-screen as normal to a maximum height of $x = -130\ px$.

2.4 Objects

Object Name	Properties	Behaviors	Graphics
Penguin	• Position (integer pair) - Constant $x = 220\ px$ - Variable y • Velocity (integer) • Life status (Boolean) • Rectangular hitbox $32x26\ px$	• Movement along y–axis - Fly's upwards - Falls downwards • Toggle alive or dead	Total size: 32x32 bit image
Post	• Position (integer pair) - Variable x - Variable y • Speed (integer) • Rectangular hitbox $64x16\ px$ Three Components: 1. Top post 2. Bottom post	• Movement along x–axis - Leftwards motion • Generate randomized variable height, or y, of gap within the range of $y = 26\ px$ to $y = 206\ px$ • Safe zone or gap height of $102\ px$	Total size: x2 64x16 bit images
Score	• Position (integer pair) - Constant $x = 295/287/279\ px$ - Constant $y = 40/140\ px$ • Score (integer) - Three individual Arabic numeral digits ranging from 0 to 9	• Increment by one • Reset to zero	Total size: 16/32/48x32 bit image (size depends on number of digits)
Button	• Position (integer pair) • Toggle status (Boolean) Four Types: 1. Toggle normal 2. Toggle hard mode 3. Toggle restart round 4. Toggle return to menu	• Toggle on or off	Total size: 64x32 bit image

2.5 Asynchronous (Input) Events

Event Name	Triggering Input Event	Description
Flight request	Spacebar key is depressed	Prompts the penguin object to fly upwards, setting its velocity to $-9\ px/frame$

2.6 Synchronous (Timed) Events

Event Name	Trigger Timing	Description
Penguin falls	Every 1/70th of a second (1 clock tick)	Penguin object moves downward $1\ px/frame$
Post moves	Every 1/70th of a second (1 clock ticks)	Post object moves forward $-4\ px/frame$

2.7 Condition-Based (Cascaded) Events

Event Name	Triggering Condition	Description
Successful pass	Penguin has successfully passed *between* the post object	While the penguin object is in motion, it passes *between* a post object. Penguin hitbox contacts score bounding line (see Figure 5)
Penguin collision – Type 1 (round over)	Penguin has moved and penguin bounding box intersects with post bounding box	While the penguin object is in motion, it contacts any part of the post object outside the designated safe zones (see Figure 5), ending the round
Penguin collision – Type 2 (round over)	Penguin has moved and penguin bounding box intersects with ground bounding box	While the penguin object is in motion, it contacts the hitbox of the ground, ending the round
Penguin collision – Type 3	Penguin has moved and penguin bounding box intersects with top edge of the screen	While the penguin object is in motion, it contacts the top edge of the screen, moving off screen flying as normal to a maximum height of $y = -130\ px$
Post collision	Post is in motion and post bounding box intersects with left edge of screen	While the post object is in motion, it contacts $x = -64\ px$ off the left edge of the screen, gradually moving off screen before wrapping around at a new randomized height

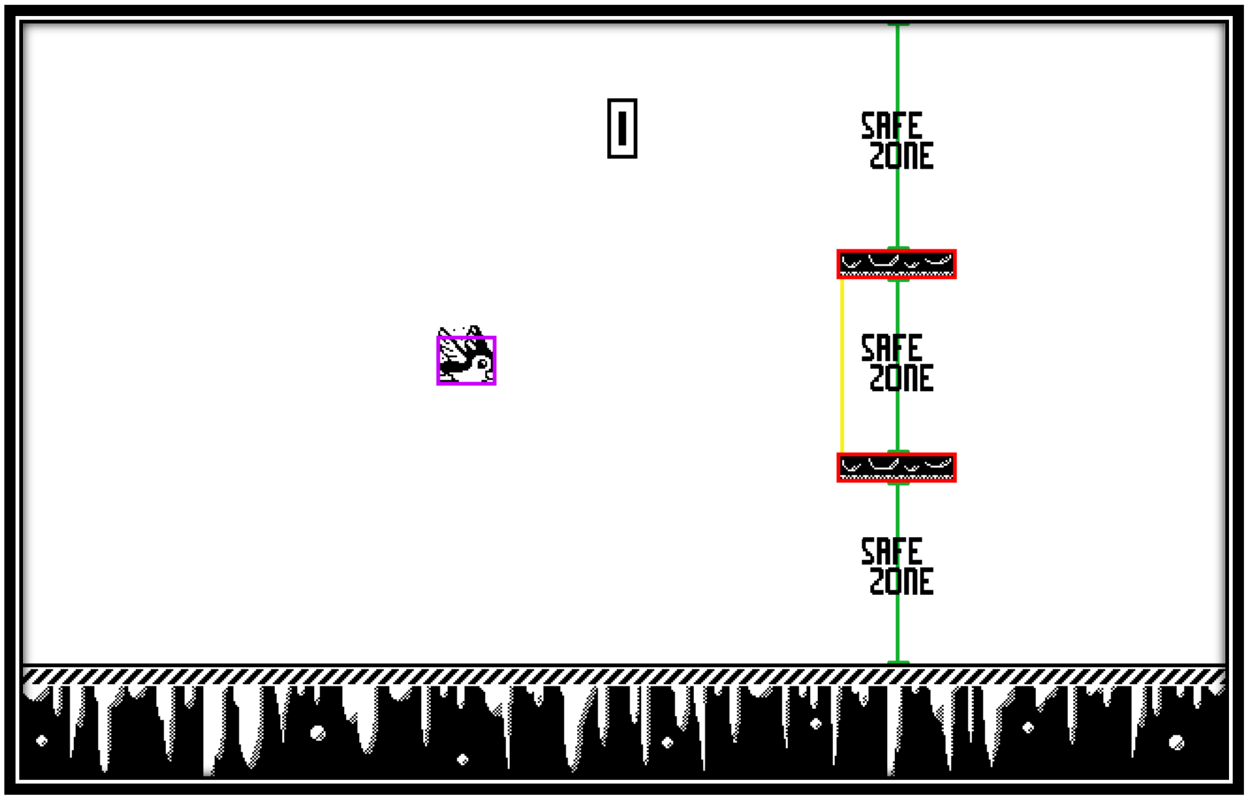

Figure 5: This edited screenshot is taken mid-game during the round state, highlighting various collision boundaries. The penguin objects hitbox is shown in purple ($32x26\ px$). The post objects hitbox is shown in red ($64x16\ px$). The collision boundary for a successful pass is shown in yellow ($1x102\ px$). The designated safe zones are shown in green.

2.8 Hypothetical Gaming Session

The game begins with loading of the home screen (see Figure 2), offering the player a choice among three options: normal, hard mode, and quit. The selection of 'normal' progresses the game as outlined below.

The start state of the round is loaded (see Figure 3), and the penguin gracefully hovers in animated flight, waiting for the player to take control. There are no posts on the screen, and the score counter is set to zero. Once the round begins, the player takes control of the penguin [presses the spacebar].

Now that the player is completely responsible for the penguin's flight, they decide to take the few seconds before the posts spawn to get acquainted with the controls, gauging how they handle with rhythmic presses of the spacebar. The player discovers the ability to maintain controlled flight or hover in place with well-timed successive spacebar presses, adjusting the speed of ascent and descent with their clicking rhythm (faster to fly upwards, and not at all to fall downwards). As the player maneuvers the penguin to the center of the screen, the first post object spawns from the right edge, with the center of the safe zone halfway between the center and top edge of the screen. As the first post object moves leftward, the player navigates the penguin upwards from the center of the screen, towards where the safe zone is located. They press the spacebar faster to fly upwards, and when they have achieved their desired height, they go back to hovering. While hovering, the player observes a second post object spawning, with the safe zone centered between the center of the screen and the ground.

The first post object continues to approach the player as they hover the penguin at the corresponding height of the first safe zone. The posts finally reach the penguin, and the player manages to press the spacebar just before potential contact with the top edge of the bottom post of the first post object, resulting in upwards flight at just the right moment to pass through the first pair of posts. The player's success is affirmed by a chime and a visible score increment from zero to one. Ready for the next challenge, the player navigates the penguin downwards from the previous gap to the new safe zone between the second pair of posts, adjusting spacebar input for a slower descent; they press the spacebar more slowly to fall downwards, hovering in place as needed. However, as the second set of posts approaches, the player loses control, resulting in the penguin colliding with the left edge of the bottom post, concluding the round, and progressing the game to its final state.

In the game's final state, it freezes the frame at the time the round had ended, where all posts stop moving, and the penguin remains where it had crashed. The score counter is replaced with a score board that shows one point, accompanied by two digital buttons for interaction: restart, or menu (see Figure 4). The player is feeling sad about losing the round; however, they decide to test their luck and start a new round by engaging the spacebar.

3 Game Play Details for Core 2-Player Version

When in two player mode, the game mechanics will play completely as normal, however, two player mode will be different in that it is a competitive alternating turn game with an objective of reaching the highest score. All rules, and gameplay will remain identical to the core one player version, but there will be minor differences graphically.

The main difference between the core one and two player version of the game is in the game (program or code) loop itself. For example, after two player mode is selected, the first player will be prompted to take their turn with the game in the start state of a round, after player one has completed their round, the second player is immediately prompted to take their turn with the game in a start state of a round; the game will only progress to its final state once both players have completed their respective turn. In the game's final state during two player mode, both players scores will be relieved with a distinct winner unless a draw is made, and the players will have the option to either restart or quit; the game will only restart if both players have clicked the restart button, however, the game will quit if either player click the quit button.

Graphically, there will be some minor differences between the core one and two player versions of the game. For example, in the start state of each round in two player mode, a moderately sized graphical text will display the respective players turn. Furthermore, during a given round during two player mode, there will be no score counter display for either player. Therefore, both players' scores will be unknown to themselves and each other until the round has concluded, adding an extra layer of mystery. Finally, in the final state of two player mode, both players scores will be displayed with a moderately sized graphical text stating the winner or a draw.

4 Sound

4.1 Music

Koji, Kondo. "The Legend of Zelda 1986 NES Intro." *Nintendo*.
https://www.youtube.com/watch?v=uyMKWJ5e1kg&ab_channel=ZeldaDungeon.

Note: players can toggle the music to start and stop by pressing the backspace key, or by clicking the volume knob (below the exit, or power button), on the main menu (Figure 2).

4.2 Sound Effects

Sound Effect Name	Brief Description	Event which Triggers Playback
Flap	Quickly sweeping white noise, making a distinct change of tone (lower to higher)	Flight motion of the penguin
Crash	Classic game over music, a few dramatic notes played relatively quickly	Penguin contacts ground or post
Chime	Fast transient high pitch bit, producing a 'ding,' like coin sound from Mario	Penguin successfully passes through a pair of posts

5 Additional Features (Time Permitting)

Hard mode. Hard mode will incorporate post objects with an additional mechanic. In hard mode, post objects will move along both the x– and y–axis. The velocity at which post objects travel horizontally will remain the same at $-4\ px/frame$, but post objects will also traverse vertically up or down the screen at $1\ px/frame$ or $-1\ px/frame$. It's important to mention that the post objects won't wrap around the screen but will instead bounce between the top and bottom boundaries; therefore, the post objects vertical velocity will alternate between the positive and negative direction. This dual motion, combining horizontal movement from right to left with vertical oscillation, adds an extra layer of challenge for players, offering an engaging twist to the gameplay.

6 Finite State Machine Diagram

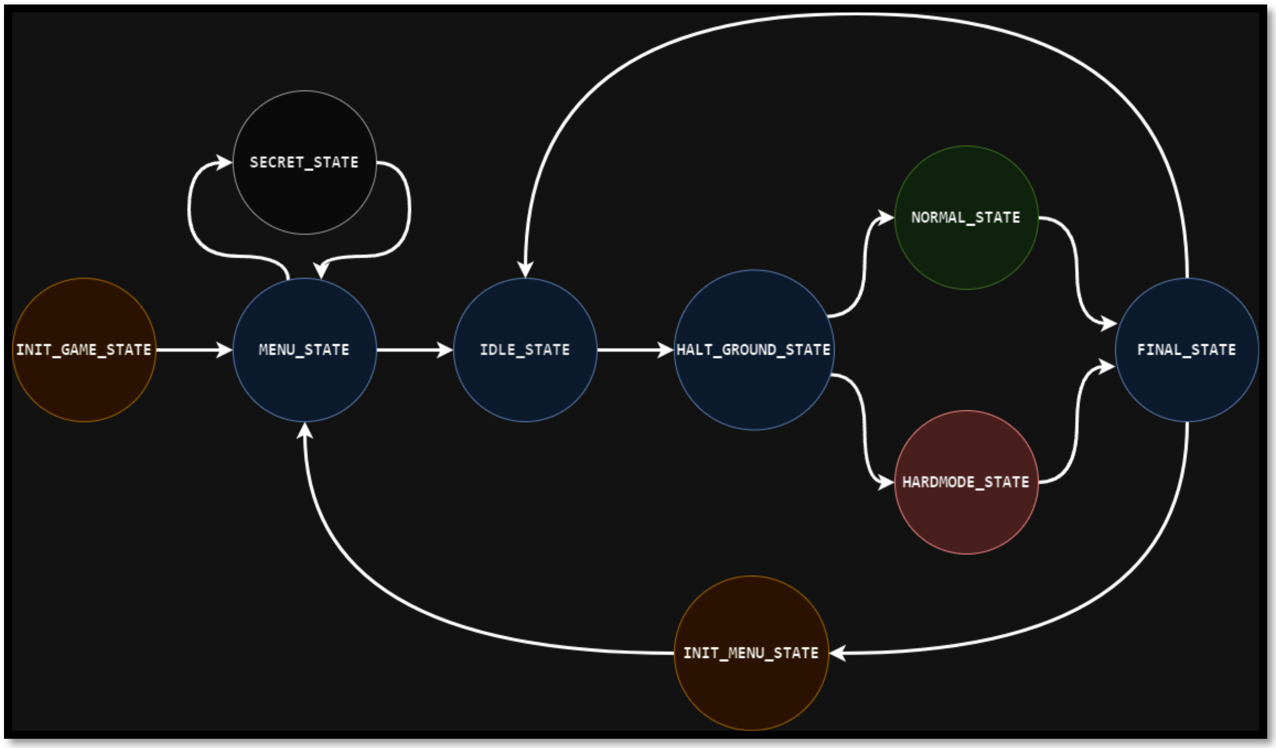

Figure 6: This diagram represents the Finite State Machine governing the game logic in '*This Penguin Flys!*'. The state machine transitions between states based on user input and internal game events, operating at a frequency of 70 updates per second.