Physics-Based Game: Angry Birds Clone

Game Description

Synopsis

This game is a physics-based clone of the popular Angry Birds game. Players use a slingshot mechanism to launch birds at structures containing pigs. The objective is to destroy all the pigs using the limited number of birds available. The game incorporates realistic physics including gravity, collision, and chain reactions when structures collapse.

Instructions to Play

1. **Launch Birds**: Click and drag on the slingshot to pull it back, then release to launch the bird
2. **Aim Carefully**: Try to hit the structures in strategic points to cause maximum damage
3. **Destroy All Pigs**: The goal is to knock down all pigs in each level
4. **Chain Reactions**: Hitting one pig can trigger a chain reaction that destroys nearby pigs

Game Screenshot

[Insert screenshot of the game showing the slingshot, birds, and pig structures here]

Game Genre

This game belongs to the physics-based puzzle genre. It is a direct descendant of the original Angry Birds game developed by Rovio Entertainment. The key similarities include:

- Slingshot mechanic for launching birds
- Physics-based destruction of structures
- Pigs as targets that need to be destroyed
- Chain reactions when structures collapse

The main differences from the original game include:

- Simplified graphics using Java 2D

- Custom physics implementation using JBox2D
- Modified physics parameters for more dramatic collisions
- Rectangular block structures instead of varied shapes

Technical Issues

Choice of Physics Engine

I chose to use JBox2D as the physics engine for this game for several reasons:

1. **Robust Collision Detection**: JBox2D provides excellent collision detection between different object types
2. **Performance Efficiency**: The engine is optimized for 2D physics simulation and has good performance characteristics
3. **Compatibility with Java**: Since the game is developed in Java, JBox2D integrates seamlessly with the rest of the codebase
4. **Community Support**: JBox2D has good documentation and examples available

Features Implemented

Physics Features:

1. **Slingshot Mechanism**:

```
public void release() {
    if (isStretching && loadedBird != null) {
        // Calculate launch vector (from slingshot center to pullback position)
        Vec2 launchVector = new Vec2(posX - pullbackPosition.x, posY -
pullbackPosition.y);

        // Apply a force proportional to the stretch distance
        float forceMagnitude = launchVector.length() * 25;
        launchVector.normalize();
        launchVector.mulLocal(forceMagnitude);

        // Make the bird dynamic again
        loadedBird.getBody().setType(org.jbox2d.dynamics.BodyType.DYNAMIC);

        // Apply the impulse to launch the bird
        loadedBird.getBody().setLinearVelocity(launchVector);
    }
}
```

This code implements a realistic slingshot mechanic where the force applied is proportional to how far the slingshot is pulled back.

2. **Gravity and Projectile Motion**: The game uses a custom gravity value of 20.0f (higher than Earth's gravity of 9.8) to create more dramatic arcs for projectiles.

```
public static final float GRAVITY = 20.0f; // Increased from 9.8f for more
realistic feeling
```

3. **Collision Detection**: The game implements collision detection between birds and pigs, as well as between different objects in the world.

```
private void checkBirdPigCollision(AngryBird bird) {
    Vec2 birdPos = bird.getBody().getPosition();
    boolean hasCollision = false;
    Pig collidedPig = null;

    for (BasicPolygon polygon : polygons) {
        if (polygon instanceof Pig) {
            Pig pig = (Pig) polygon;
            if (!pig.isDestroyed()) {
                Vec2 pigPos = pig.getBody().getPosition();
                float distance = pigPos.sub(birdPos).length();

                if (distance < 1.0f) {
                    hasCollision = true;
                    collidedPig = pig;
                    break;
                }
            }
        }
    }
}
```

4. **Chain Reactions**: When a bird hits a pig or structure, it can trigger a chain reaction that affects nearby objects.

```
if (hasCollision && collidedPig != null) {
    Vec2 collidedPos = collidedPig.getBody().getPosition();

    for (BasicPolygon polygon : polygons) {
        if (polygon instanceof Pig) {
            Pig pig = (Pig) polygon;
            if (!pig.isDestroyed()) {
                Vec2 pigPos = pig.getBody().getPosition();
                float distance = pigPos.sub(collidedPos).length();

                // Slightly reduced the chain reaction range
                if (distance < 2.0f) {
                    pig.activate();
                }
            }
        }
    }
}
```

5. **Rigid Body Physics**: The game uses rigid body physics for all objects, allowing for realistic interactions including rotation and momentum transfer during collisions.

Gameplay Features:

1. **Slingshot Interface**: A visual and interactive slingshot that responds to mouse input
2. **Block Structures**: Different arrangements of blocks and pigs that present unique challenges
3. **Visual Feedback**: Visual representation of the slingshot stretching
4. **State Management**: Tracking of game state including which pigs have been destroyed

Physics Principles Used

Elastic Potential Energy

The slingshot converts elastic potential energy to kinetic energy when released. The elastic band's stretch distance determines the force applied:

```
float forceMagnitude = launchVector.length() * 25;
```

Projectile Motion

Birds follow a parabolic trajectory after launch due to constant gravity:

```
public static final float GRAVITY = 20.0f;
```

Conservation of Momentum

During collisions, momentum is conserved, allowing birds to transfer their momentum to pigs and blocks upon impact.

Friction and Damping

Objects have friction coefficients that affect how they slide against each other:

```
super(sx, sy, vx, vy, radius, Color.YELLOW, 2.0f, 0.1f, 4);
// Density of 2.0f and friction of 0.1f
```

Bugs and Unfinished Features

Known Bugs:

1. **Collision Detection Precision**: Sometimes collisions between birds and small objects are not detected correctly
2. **Physics Instability**: At high velocities, some objects may pass through each other
3. **Performance Issues**: With many objects, the physics simulation can slow down

Unfinished Features:

1. **Multiple Bird Types**: The game currently only has one type of bird
2. **Level Progression**: No level system is implemented yet
3. **Score System**: No scoring mechanism has been implemented

Parameter Tuning

To make the game more playable and enjoyable, several physics parameters were tuned:

1. **Gravity**: Increased from 9.8f to 20.0f to make projectiles fall faster and create more dramatic impacts
2. **Bird Mass**: Set to 5.0f (higher than other objects) to increase impact force
3. **Slingshot Force Multiplier**: Set to 25 to provide sufficient launch velocity
4. **Maximum Stretch Distance**: Limited to 2.0f world units to prevent excessive force
5. **Pig Density**: Set to 2.0f to make them sufficiently stable but still destructible
6. **Chain Reaction Distance**: Set to 2.0f world units to create satisfying cascades of destruction

These parameters were tuned through iterative testing to find the right balance between challenge and satisfaction.

Reflection

Achievements

I successfully implemented a functional Angry Birds clone with realistic physics using JBox2D. The core gameplay mechanics work well, particularly:

- The slingshot launching mechanism feels satisfying and intuitive
- The physics-based destruction creates unpredictable and fun outcomes
- The chain reaction system adds strategic depth to the gameplay

Development Process

Easy Aspects:

- Setting up the basic JBox2D world and simple objects
- Implementing the mouse controls for the slingshot
- Creating the basic visual elements

Difficult Aspects:

- Fine-tuning the physics parameters to create the right "feel"
- Implementing accurate collision detection, especially for chain reactions
- Optimizing performance with many objects in the scene
- Creating visually appealing destruction effects

Unresolved Problems

- The collision detection system could be improved for more accurate interactions
- The game lacks visual polish and sound effects
- The pig activation system sometimes activates pigs that are too far from the impact

Proud Achievements

I'm particularly proud of:

1. The implementation of the slingshot mechanism, which feels responsive and intuitive
2. The chain reaction system that creates satisfying cascades of destruction
3. The overall physics simulation that creates unpredictable and entertaining outcomes

Apportioning Credit

Code Attribution

- **Original Code**:
    - `AngryBird.java`: The bird implementation
    - `Pig.java`: The pig target implementation
    - `Slingshot.java`: The slingshot mechanism
    - Bird-pig collision detection in `BasicPhysicsEngineUsingBox2D.java`
    - Chain reaction implementation
- **Modified from Lab/Course Materials**:
    - `BasicPhysicsEngineUsingBox2D.java`: Based on the lab wrapper but extensively modified
    - `BasicParticle.java`: Modified to work with bird physics
- **Unmodified from Lab/Course Materials**:

    - Basic physics engine components
    - GUI framework
    - Barrier implementations
    - Basic collision detection framework

Artwork Attribution

- Bird and pig sprites: [Source information]
- Slingshot image: [Source information]
- Block textures: [Source information]

[Note: Please replace the placeholder source information with actual attribution]