Angry Bird Game

Harry Hong, hong0506@umn.edu

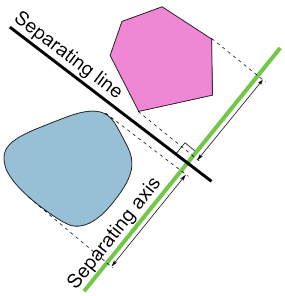
December 17, 2022

Project outline

In the final state, rigid bodies can collide naturally. They can be stacked together or rest on the ground without jitters. A bird (ball) can be launched by any force and angle to destroy bricks (rigid bodies) and break them into smaller bricks.

Key algorithms

The key algorithm used for collision detection is the hyperplane separation theorem. If two convex polygons do not overlap, there exists a hyperplane between them, although there may not be any space between the two sets. A separating axis is an axis that is perpendicular to a separating hyperplane, and the projections of the convex shapes onto this axis do not overlap. If a hyperplane does not exist, the polygons overlap. The detection job is to find a hyperplane with minimum gap.



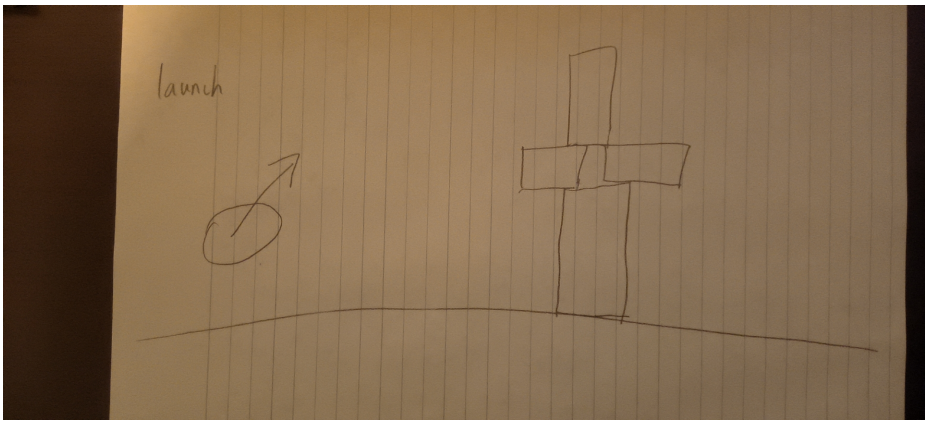
Time of impact is used to eliminate jitters when rigid bodies are stacked or resting on the ground. On my laptop, the animation is fluent and natural even when I increase the number of rigid bodies to 10 times or 100 times. I do not see any computational bottleneck.

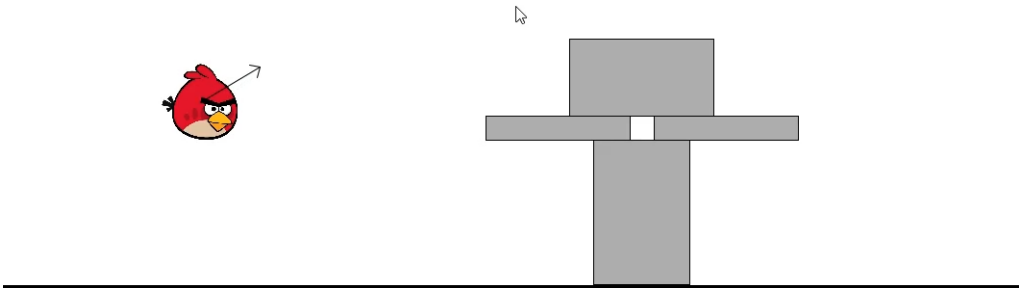
Specific questions

Collision detection and continuous collision detection are addressed.

Sketch vs Final state

The final state result is very similar to my sketch. What I did not expect was that I originally planned to fully use my own code, but ended up using an external physics library box2d because the animation was more natural and fluent.





Feedbacks

The feedback suggests the animation be more interactive, enabling the ball to break the bricks into smaller bricks instead of letting them disappear directly and implement a 3D version. I adopted the suggestion to break bricks into smaller bricks.

Comparison with state-of-art

This implementation calculates physics parameters and predicts their values in the next time step to handle collisions between rigid bodies. A state-of-the-art approach suggests using a neural network to learn the motion of rigid bodies. The neural network does not compute

the physics parameters directly, but rather learns and approximates the motions from given data. This approach addresses a limitation of the hyperplane separation theorem.

Future and limitation

I can add more shapes, such as concave polygons. The limitation is that the hyperplane separation theorem does not solve the collision detection between concave polygons. I have an immature idea for solving collision between concave polygons. We can treat the concave polygon as several bound convex polygons. Then, the problem becomes collision detection between convex polygons again. However, this approach would require more memory and computation resources. In addition, I can use a trained neural network from the state-of-the-art approach to approximate a natural motion of concave polygons.

Library

I used box2d for Processing.