Testing JBox2D

Jacob Lopez

Daniel Moon

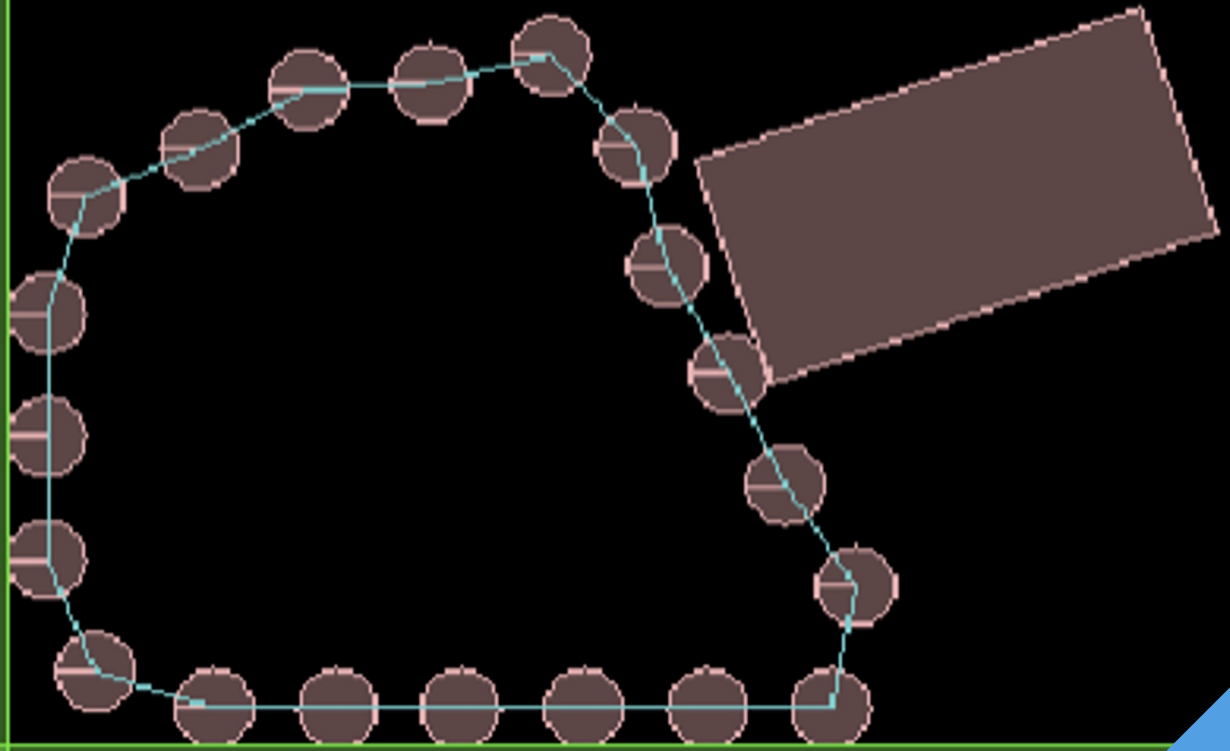
Maria Movsheva

Software Introduction

- Physics engine ported from Box2D
- Simulates realistic physics in 2D environments, runs on all platforms and OS, mobile and PC
- Users of the app include developers of 2D games and the customers for those games
- Developed in Java, but the original was developed in C++
- Angry Birds, King of Thieves, Happy Wheels, Shovel K

HAPPY WHEELS

Demo: JBox2D



Testing techniques

Mutation Testing (PIT)

1. Mutation Coverage: 70%+

Blackbox Testing (JUnit5)

1. Equivalence Partitioning
2. Boundary Value Analysis
3. Error Guessing

Whitebox Testing (Jacoco)

1. Statement Coverage: 90%+
2. Branch Coverage: 70%+

Results: Test Statistics

Tested 2 modules over 6 classes

Collision

Shapes

Chain	Circle	Edge
Polygon	MassData	

Developed 254 tests with 31 failures

Results: Whitebox

org.jbox2d.collision.shapes

Element	Missed Instructions	Cov.	Missed Branches	Cov.
PolygonShape		96%		81%
ChainShape		93%		76%
EdgeShape		99%		95%
CircleShape		99%		75%
MassData		100%		n/a
ShapeType		100%		n/a
Shape		100%		n/a
Total	96 of 3,253	97%	45 of 240	81%

Collision

Element	Missed Instructions	Cov.	Missed Branches	Cov.
collideEdgeAndCircle(Manifold, EdgeShape, Transform, CircleShape, Transform)		80%		62%
collidePolygonAndCircle(Manifold, PolygonShape, Transform, CircleShape, Transform)		86%		75%
findMaxSeparation(Collision.EdgeResults, PolygonShape, Transform, PolygonShape, Transform)		74%		45%
collidePolygons(Manifold, PolygonShape, Transform, PolygonShape, Transform)		95%		77%
findIncidentEdge(Collision.ClipVertex[], PolygonShape, Transform, int, PolygonShape, Transform)		98%		75%
getPointStates(Collision.PointState[], Collision.PointState[], Manifold, Manifold)		95%		71%
static {...}		83%		50%
edgeSeparation(PolygonShape, Transform, int, PolygonShape, Transform)		100%		90%
Collision(IWorldPool)		100%		n/a
collideCircles(Manifold, CircleShape, Transform, CircleShape, Transform)		100%		100%
clipSegmentToLine(Collision.ClipVertex[], Collision.ClipVertex[], Vec2, float, int)		100%		100%
testOverlap(Shape, int, Shape, int, Transform, Transform)		100%		100%
collideEdgeAndPolygon(Manifold, EdgeShape, Transform, PolygonShape, Transform)		100%		n/a
Total	250 of 2,689	90%	39 of 132	70%

Results: Mutation

Package Summary

org.jbox2d.collision.shapes

Number of Classes	Line Coverage		Mutation Coverage		Test Strength	
6	99%	530/537	82%	384/466	83%	384/463

Breakdown by Class

Name	Line Coverage		Mutation Coverage		Test Strength	
ChainShape.java	100%	103/103	71%	25/35	71%	25/35
CircleShape.java	100%	60/60	86%	78/91	86%	78/91
EdgeShape.java	99%	90/91	84%	81/96	85%	81/95
MassData.java	100%	14/14	100%	1/1	100%	1/1
PolygonShape.java	98%	256/262	82%	197/241	82%	197/239
Shape.java	100%	7/7	100%	2/2	100%	2/2

Faults Found

- Found numerous faults: going over three of them
- Collisions aren’t detected if they are within a margin error: documentation is not clear on what the margin is and neither is the code

```
final float k_relativeTol = 0.98f;
final float k_absoluteTol = 0.001f;

if (results2.separation > k_relativeTol * results1.separation + k_absoluteTol) {
  poly1 = polyB;
  poly2 = polyA;
```

```
pool.getDistance().distance(output, cache, input);
// djm note: anything significant about 10.0f?
return output.distance < 10.0f * Settings.EPSILON;
```

Faults Found

```
170   public Shape clone() {
171     ChainShape clone = new ChainShape();
172     clone.createChain(m_vertices, m_count);
```

count = m_count -> assert fails

```
215   public void createChain(final Vec2 vertices[], int count) {
216     assert (m_vertices == null && m_count == 0);
217     assert (count >= 2);
218     m_count = count;
219     m_vertices = new Vec2[m_count];
220     for (int i = 1; i < m_count; i++) {...}
228     for (int i = 0; i < m_count; i++) {...}
231     m_hasPrevVertex = false;
232     m_hasNextVertex = false;
233
234     m_prevVertex.setZero();
235     m_nextVertex.setZero();
236   }
```

You can not clone a **ChainShape** object:
an exception as the assert statements clash with function parameters and field values

Faults Found

```
public void setRadius(float radius) { this.m_radius = radius; }
```

```
public float getRadius() { return m_radius; }
```

A negative radius can cause issues, inconsistent in documentations.

```
public final void set(final Vec2[] verts, final int num, final Vec2Array vecPool, final IntArray intPool) {
  assert (3 <= num && num <= Settings.maxPolygonVertices);
  if (num < 3) {
    setAsBox( hx: 1.0f,  hy: 1.0f);
    return;
  }
  /*...*/
```

The **set()** method in **PolygonShape** also results in error when two vectors in the array are the same

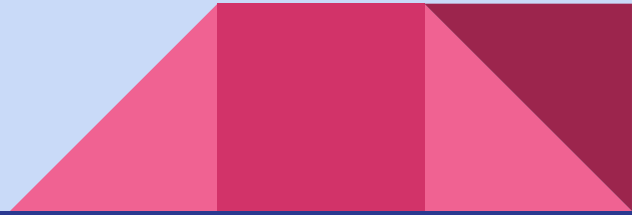
****

In general, there are a lot of unreachable statements and branches present within the classes

****

Software Fault Patches & Suggestions

- Write more unit tests for sanity checking correctness and API design
- Documentation for JBox2D is nonexistent, and they have methods not found in the original Box2D C++ implementation
- Wrap fields around getters and setters and perform data validation through them. (Avoids negative radius)



Summary

- Branch testing and mutation testing were the most effective in revealing faults in the SUT
- The code contained a lot of vulnerable areas that could be origin points for faults
- A lot of improvements needed for the library, especially since the last release was 10 years ago

Thank You

Questions?