Final Report
CS 4999 - Fall 2022 - Non-deterministic behaviors in Box2D - Prof. White
Leo Zhao

Abstract

We investigated the *Ragdoll Royale*'s non-deterministic physics behavior and found bugs in the source code. We did not verify that such bugs were causing the behavior due to difficulties with the game server. We implemented the **fixed-step-size physics** on the *Geometry lab (CUGL), Rocket game of physics lab (Libgdx), and Box2D demo (Libgdx)*, and **verified the deterministic** behavior through compare-benchmark analysis.

Deliverable

(PhysicsLab.zip & PhysicsLab.jar) The submitted code is developed based on the solution to Lab 4 Physics of the Intro class. Two major implementations are added to the source code: the **compare-benchmark** and the **fixed-step-size physics.**

Benchmark

The goal of the compare-benchmark is to run two games in parallel. Each of the game has its own Box2D-world and List of Obstacles, which contains references to body/fixtures that belong to the world. Therefore, the physics of the two games are completely independent; populateLevel() will initialize both games, while player input is applied to both games. In order to implement this, I duplicate the game by summarizing the physics (world, obstacles list, add-queue …) into the module of *WorldBenchmark*, while the *WorldController* holds two references of this module, the *real* and the *compare*.

There are a few differences between the two games.

1. Winning and losing conditions are determined by the collision in the real game only to simplify the code.
2. Only the objects in the real game are drawn to the canvas. If debug mode is on, objects in the compare game are drawn to the canvas in debug boxes. If there's a disparity between the objects and their debug box, then indeterminate behavior is observed.
3. The real game will always update the world in true intervals, according to the frame rate. The compare game will update the world in skewed intervals, which will produce a similar but different frame rate. The algorithm that determines the skewed interval **knows** the physics-step-size. This is necessary because we want the real and compare game to always step the same times, while having different remainingTime.

Physics

The goal of the fixed-step physics is to compare the behavior before and after the implementation. In the deliverable file, there are 7 games, summarized in the table below.

	Controller	Type	Physics	Draw remaining time (RT)
0	Rocket	Original	Step by frame rate	N/A
1	RocketPhysics	Det	Fixed step size	Neglect RT
2	RocketVelocity	Det	Fixed step size	Offset position by RT * velocity
3	RocketDoubleWorld	Det	Fixed step size	Step draw-world by RT
4	Platform	Original	Step by frame rate	N/A
5	PlatformPhysics	Unfinished	Fixed step size	Neglect RT
6	Ragdoll	Original	Step by frame rate	N/A

Controllers 0, 4, 6 are the original controllers of the lab. Their physics are non-deterministic, as expected. Controller 1, 2, 3 all implemented the fixed step size physics, while having different drawing techniques.

In controller 1 (and 5), objects are drawn with their real-position, neglecting the remaining time that is less than the step-size. In controller 2, objects are drawn with an offset which equals to its current velocity times the remaining time. If there are no forces in the interval, such drawings should be accurate. In controller 3, objects are drawn with the draw-body in the draw world. After updating the physics in the real world, we sync the draw-world to it and step the draw world with the remaining time. This drawing should always be accurate.

We compare the drawings for the three worlds with different step-size, and summarize in the table below.

Step size (ms)	**3.0**	**7.0**	**16.6**	**42.0**	**100.0**
Frame rate (Hz)	**333**	**144**	**60**	**24.5**	**10**
1 (neglect RT)	n	n	n	sig	sig
2 (velocity offset)	n	n	n	some	sig
3 (dup-worlds)	n	n	n	some	sig

For frame rate greater than 45 Hz, no non-harmonic behavior is observed in all three implementations. As frame rate goes down, there's significantly more non-harmonic behavior in the first controller than the remaining two. However, as the frame rate drops even further, all three worlds have significant observable non-harmonic behavior. In general, at the standard game frame rate (>60 Hz), no significant non-harmonic behavior is observed in all controllers.

However, this may be due to the fact that the rocket world generally has simple physics interactions, without involving components such as complex obstacles or bullets.

Controller 5 is the platform version of controller 1, and the platform game involves more physics. However, this controller is not finished due to bugs. When the avatar fires bullets while jumping on the rope-bridge, the avatar in the compare-game may run into a stage where the avatar can keep jumping without landing. This produced a disparity between the real and compare game, though not due to physics problems.

Conclusion

All evidence we process at this time supports the conclusion that fix-step-size physics implementation provides deterministic physics simulation in Box2D. All previous observations of non-deterministic behavior were due to issues other than the physics simulations. In terms of approaches to drawing the remaining time, combining the trade-off between the work of implementation and potential non-harmonic effect, the velocity-offset approach is the most preferable as the first attempt, since it requires much less work and behaves well.