Lockstep in a Nutshell

Chris Kissler • Eric Kaltman • COMP 499

What is it?

Lockstep in a Nutshell is an interactive learning tool that aims to be an entertaining and engaging way to learn about a rather technical game development concept. The actual software is a webpage with text explanations broken up by interactive (or playable) demonstrations. The concept being taught through the software is video game netcode, specifically the type that is used in fighting games. This explanation takes the form of a website because the visuals are simple enough to be rendered using sprites and a javascript drawing library (p5.js), so it seemed like an obvious choice to reap the benefits of an internet-hosted website, such as easy sharing and broad device compatibility. My teaching strategy with this website was to introduce concepts in text and allow the user to explore the concepts immediately after.

Inspiration and Early Steps

Around the summer of 2019 there was a spike in interest in and awareness regarding the poor state of netcode in fighting games. This coincided with my own deep dive into learning about rollback netcode. The concept seemed so elegant once I understood it that I was motivated to make a test project that exhibits the difference in feel between delay and rollback netcode. That project was made to test my understanding of rollback, but it did little to explain what it was doing to new viewers. This preliminary project can still be viewed here: http://whiffpunish.com/netcode

The initial project's lack of teaching ability served as my motivation for creating this project. Lockstep in a Nutshell aims to teach people about lockstep netcode in general, starting with the simplest (naive) implementation of lockstep netcode and gradually increasing in complexity.

Creating a Serializable Game

While a game that utilizes any form of lockstep netcode must be deterministic (the same inputs will always result in the same output), rollback netcode necessitates another feature: serializable game states. This means that any given state of the game must be able to be saved and loaded. In rollback's case, the game also needs to be able to re-simulate N frames of the game in less than 1 frame of actual game time, where N is the maximum amount of "tolerable" remote player latency (in units of frames). For my simple interactive demos this isn't an issue, as the gameplay logic is all very simple, and as a result an unnecessarily high number of states can be iterated through in the time span of a single frame. For more complex games, though, this can be a real issue, particularly if things like rendering aren't fully decoupled from the gameplay logic.

The Game Design

Though very simple by design, the game used for demonstration throughout the article did have considerable thought put into what it would contain in order to best showcase the effects of the different types of netcode. Rollback's major advantage over delay based netcode is that its input delay will never fluctuate while the latency remains under a generous, fixed threshold. Therefore I needed my game to involve a strict timing challenge that would be fairly easy for most people to accomplish, all while still being recognizable as more-or-less a fighting games.

The solution I settled on was to design the game with an easy infinite combo so I could task the player with performing that combo in different simulated network conditions on the different types of netcode. It's still very clearly a basic fighting game with its movement, attack, and combo counter, and the combo is literally as simple as just tapping a single button at the proper rhythm. Rollback prevents this rhythm from ever being interrupted by latency spikes, while delay does not.

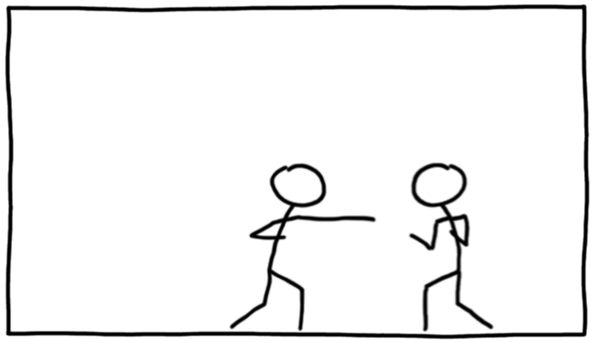

Fig. 1 – Basic Gameplay

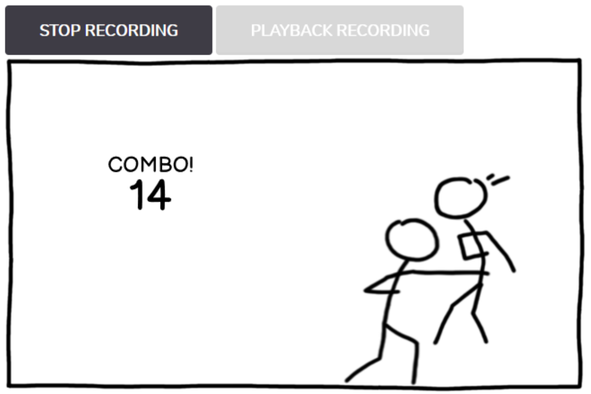

Fig. 2 – Game Determinism Demonstration

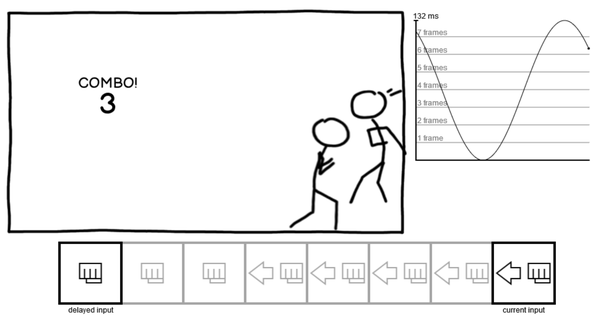

Fig. 3 – Data Visualizations

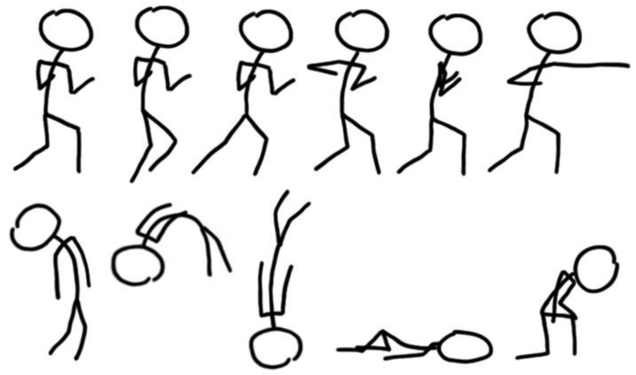

Fig. 4 – Partial Sprite Sheet

Software

The webpage was created using HTML, CSS, and JavaScript. The CSS was built off of a free template, and the JavaScript interactivity uses the P5.js drawing library.

Acknowledgments

Parable of the Polygons – An interactive article that explains a non-gaming concept using game-like interactive components. Provided a lot of the original inspiration/concept for how Rollback in a Nutshell would explain its concepts and be laid out.
https://ncase.me/polygons/

The video from which I personally learned how rollback works.
https://www.youtube.com/watch?v=7jb0FOcImdg

A web article that also seeks to explain rollback netcode, albeit non-interactively.
http://ki.infil.net/w02-netcode.html

Another video that talks about rollback. Newer and more accessible than the one I first used.
https://www.youtube.com/watch?v=1R15scXYhK0