Zen fish: post-game analysis

Game: https://jalcaldem.itch.io/zen-fish
JAM submission: https://itch.io/jam/brackeys-12/rate/2968739
JAM information: https://itch.io/jam/brackeys-12

¿Why make this game?
Inspired by a video (https://www.youtube.com/watch?v=qlfh_rv6khY), I decided to use the chained distance constraints technique to elaborate a prototype of a procedurally animated, worm-like shape in LÖVE 2D, the engine I currently use to make games. Then, when the theme for the Brackeys Game Jam 2024 was revealed ("Calm before the storm"), I decided to use this prototype to create a very calm game about a fish swimming. The game had to be short and simple, since most of the time would be spent on the fish itself and not on the mechanics, and also I couldn't use the whole week because the last 2 days overlapped with another jam. Also, since most of my games are mechanics-first and treat aesthetics and polish as secondary considerations, this would be a good opportunity to focus on these aspects instead. This ended up creating a very relaxed and pleasant experience for most players, but for me also hiding a surprisingly interesting puzzle to solve when "speedrunning" the game.

The players liked (in relevance order)…
- The overall aesthetics of the game including the visuals, the music and the minimalism of the mechanics.
- The fish design and animation.
- The little details, like the fish looking at the pellets, opening the mouth before eating or how the fish seems to "digests" the food.

The players did not like (in relevance order) …
- The fish controls seemed a bit "wobbly" for some players, probably due to the relatively fast camera movement. This aspect was also commented on by the only person playing the game before release, which resulted in changes to the camera movement parameters, although clearly they weren't enough to create a completely fluid experience. It is tricky to alter these parameters since they also affect the position of the fish in the screen, so more experimentation would be needed. Another possible solution would be to play on controller instead of mouse, in which case a joystick would directly control the angle of the fish. Controller input is currently not implemented, but maybe in the future I will experiment with this idea.
- It does not exactly fit the theme "Calm before the storm" because there is no storm. Originally there were plans to add raindrops at increasing frequency to move the pellets and make the game a bit harder over time, but this was scrapped because it would consume too many resources but not significatively improve the experience. A surprisingly large number of players at the Itch.io forum reported that the "storm" part of the theme was not present. The reason behind this could be attributed to not having other negative comments when giving constructive criticism to a creator, but this hypothesis is based on my own experiences given feedback, so the real cause can be

different. While adhering to the theme during a jam can be a good way of having limitations that stimulate creativity it can also hinder the final result if you try to force it, specially in this case, since the main reason behind the making of the game was to make use of the procedural animation system and create a minimalist game.
- Due to the random nature of the pellet coordinates, it is relatively frequent that players end up missing just one or two of them when they run out of time, causing the game to be “too hard” for some. After playing many times trying to get all the pellets faster and developing techniques to do it, I liked the variability this randomization creates because it adds a risk element to some of the more “greedy” techniques that prioritize eating visible pellets instead of more safe strategies based on area clearing. However, it is also true that bad randomness can create too much difference between plays, so the game could benefit from other approaches to spawning pellets like using space segmentation or procedural generation to create more stable distributions.

Potential improvements not mentioned by players

- The fish animation is not realistic, it should look like it is propelled by the tail fin instead of the undulating movements of the body itself.
- The fish movement design (more information on that later) is never properly explained to players, so they end up not knowing how to optimally move the fish if they don’t pay attention to understand it.
- The shape of the play space is squared, which resembles a fish tank but also creates some problems with the corners being less inaccessible, and creating situations in which collisions with the borders are inevitable. After some thought, I believe that the play space could be a circle instead, which would not only contribute to less overall collisions with the borders but also fit the other elements (the pellets are circles, the fish is made of circles).
- The pellets that spawn too close to the play space limits are unnecessarily hard to eat. This can be easily fixed by adding a margin to the pellet coordinate limit.
- Some problems with the HTML port, particularly the Ith.io embed version, which include a slight resolution decrease and visual glitches at the fins. This is hard to evaluate, since the port is made using an external tool (adapted from https://github.com/Davidobot/love.js) and I haven’t written that code.
- At some angles, the dorsal fin shape occupies more area than it should. This happens because the default polygon shape in LÖVE 2D cannot draw concave polygons, and it could be fixed with the use of an external library but it was deemed not critical.
- The calculations for the parametric curves are unnecessarily complex and created some problems during development, so probably they should be simplified with the use of external libraries too.
- More effort could be spent into creating a more engaging end-game screen, maybe adding a detailed timer to incite the player to get a better result, or giving some tips about how to play better if they failed.
- The cursor could be replaced by some kind of “bait” that thematically makes sense for the fish to chase, at least until distracted by the pellets.

Fish and camera movement design notes

The movement of the fish was designed so that it incites the player to guide the natural movement of the fish instead of making him move forcefully. It works like this:

- The mouse acts as a reference point (or just point, from here onwards) for the fish and the camera, both trying to move towards it at any time.
- The camera accelerates if far from the point up to a maximum speed, and decelerates if close down to a minimum speed. Also, it has a maximum distance, so that the camera never lags too far.
- If facing the point, it accelerates up to a maximum speed. If not, it rotates towards it and decelerates (at 3x the acceleration rate) down to a minimum speed. It has a maximum rotation speed, so it cannot rotate too fast.
- When accelerating or at maximum speed, the fish periodically changes its angle within a fixed limit to perform an oscillating movement and visually indicate that it is actively moving.
- If close to a pellet, the fish slightly decelerates and stops oscillating. This was implemented to facilitate eating the pellets, since the oscillation made it very hard and the maximum speed is too punishing when trying to eat. Also, it makes it seem like the fish purposefully stops swimming to focus on catching the food, which together with other visual changes (it looks at the pellet and opens its mouth) creates a slight illusion of intelligence.
- All movement parameters are configured in a way that at least half of the fish is always visible within the camera, but usually it is completely visible, to highlight the reactivity that the fish model has to the player input.

The result is that the player is encouraged to keep momentum by not making many sudden turns, but instead trying to keep the fish speed close to the maximum and follow the current movement direction. This creates interesting situations, because the most efficient movement patterns usually consist of dynamically reacting to the pellets at sight and trace an imaginary curved line that minimizes the turns instead of just a repetitive “go to the closer point” strategy. However, going at high speeds and more complex mouse inputs mean a higher chance of missing a pellet and being punished to have to go back to it, which in itself creates an opportunity to recalculate the optimal movement. Unfortunately, I believe that due to the brief nature of the game not many players spend enough time with it to experience these considerations and to give feedback about them.

Conclusion

Overall, I am very satisfied with the final game. The main objective was to use the procedural animation system, and the minimalism of the game really puts the spotlight on it. Making the fish look like an actual fish by creating the fins and their movement, and adjusting the color patterns… took a lot of unexpected effort but it was worth it. And since the game is exceptionally simple, both casuals and hardcore players can enjoy it. In the future, I will take more consideration into the aesthetic vision for the game to make it more attractive to all kinds of players (and also more visually appealing when compared against other games in the same jam).