Reflection

Project Overview [Maximum 100 words]

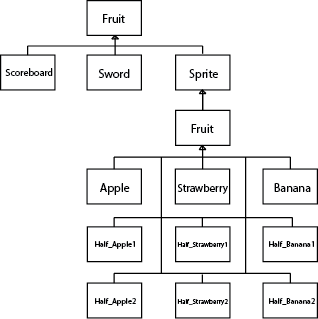
Based on the mobile app game 'Fruit Ninja', our project combines some of the game mechanics with a motion controlled user interface. ninja.py uses the OpenCV library to track hand motion, and correlate any movement to a fruit-slicing sword.

Results [~2-3 paragraphs + figures/examples]

Our first deliverable takes the form of a mouse controlled version, found in tests/ninja\_oldv.py.

Our second changes the user input to take the form of webcam sensing, and tracks the user's hand to correspond with fruit.

The final game shows the fruit being cut in half once the sword intersects with it. A simple scoreboard is kept at the top to let the user know their stats. The game is infinite: have fun!

Implementation [~2-3 paragraphs + UML diagram]

We used three objects in this project: Fruit, Sword, and Scoreboard. The fruit object was called on 9 separate occasions, with the only difference between the different fruits the image that appeared on the screen. Otherwise, all of the fruits would act the same way. The three fruits, Apple, Strawberry, and Banana were tossed into the air using the class defined ‘toss’ function. The halves of the fruits fell according to the ‘fall’ function. [talk about these more]

The sword object was used to represent input from openCV. The object was set up to take information from an easily changeable input, like a mouse or openCV, and move around the screen accordingly. When the rectangles associated with the different sprites collided, the fruit was removed from the fruits cache, and two corresponding half fruit objects were generated.

Design decision: While we could have made two classes here, Fruit and Half\_Fruit, we thought that for the reader’s sake, half a fruit was still technically a fruit. [required]

Reflection [~2 paragraphs]

Our workflow for this project was opposite of how most projects work: we essentially finished our project in the first week. This was mostly so that our second week could be freed up for our other classes, but had a secondary effect of burning us out. That being said, we were originally worried about the scoping of our project, but we had framed our idea in such a way that the MVP was achievable within hours, and some of the stretch goals were reached by the end of the week. We split the project up into manageable chunks: creating falling fruit, cutting the falling fruit with a mouse, throwing and cutting the fruit. After we got a decent MVP working with a mouse, the openCV input was added to give the game another dimension.

As a team, we worked very well together. A lot of the programming and learning was done outside of the meetings, and we were never working on the same issue at once. Once the MVP was created, one of us worked on game mechanics like the motion and cutting of the fruit, and the other worked on the openCV input. During the work phase of this project, we would check in almost every other day to check and compile each other's work, and create a new plan for moving forward. While this was a very productive approach, this project became more about reaching a final project instead of creating experience pair-programming. Honestly, we would both probably use this approach next time since both of us are goal-oriented rather that process-oriented.