Livid Farm

Mark Shanoudy

CS 1699 - FINAL DELIVERABLE

For this deliverable, I wrote a new application from scratch using TDD. I decided to write a software bot to automate an activity in the MMO, Runescape. The activity I chose is called Livid Farm. Livid Farm is required in order to unlock a set of Lunar Spells for your character. In order to unlock these spells, you must accumulate 850,000 produce points. At maximum efficiency, this would take a **minimum** of 44 hours and 50 minutes. Because of how long it takes to complete Livid Farm and how repetitive it is, I greatly wanted to find a way to automate the process so I wouldn’t have to sit in front of the computer for close to two days’ worth of time.

Produce points are earned by completing tasks around the farm. These tasks include fertilizing empty plots, curing diseased livid plants, fixing broken fences, and bundling harvested livid produce. Every 60 seconds, the farm’s layout will randomly change. Each round is guaranteed to have exactly three empty plots and exactly three diseased livid plants. Produce appears every third and fifth round.

A more detailed description of Livid Farm’s mechanics and strategy can be found at the Wiki: <http://runescape.wikia.com/wiki/Livid_Farm>

Due to the nature of the randomly changing plots, there was no way to simply script the mouse movements necessary to automate Livid Farm. I instead opted to take a computer vision approach by analyzing a screenshot of the info label that appears to the right of the cursor when you hover over a clickable object. My approach consists of converting the screenshot to an array of pixel color information (red, blue, green, alpha) which is then summed and compared against a pre-calculated set of summations to determine if the object being hovered over is a possible task to complete for produce points.

I tested the software bot by hand initially to make sure that the coordinate mappings and bot timing were correct and fully functional. I then went back and wrote unit tests for each public method implemented by the bot.

One issue I faced when writing this application’s tests was that all but one of the bot’s methods were of type void. Furthermore, due to the nature of how the bot operates, I could not find a way to test any side effects caused by these type void methods. Instead, I decided to test for thrown exceptions caused by invalid parameters being passed to the methods in order to verify the methods were functioning correctly.

There were no failed tests. However, it may be a problem that there was no way to directly test the majority of the bot’s functionality. This is in essence a prototype. Going forward there is potential to automate other aspects of Runescape using the underlying engine, which will require more tests be written. This application is not ready to be released.

<https://github.com/mshanoudy/deliverable5_repo>