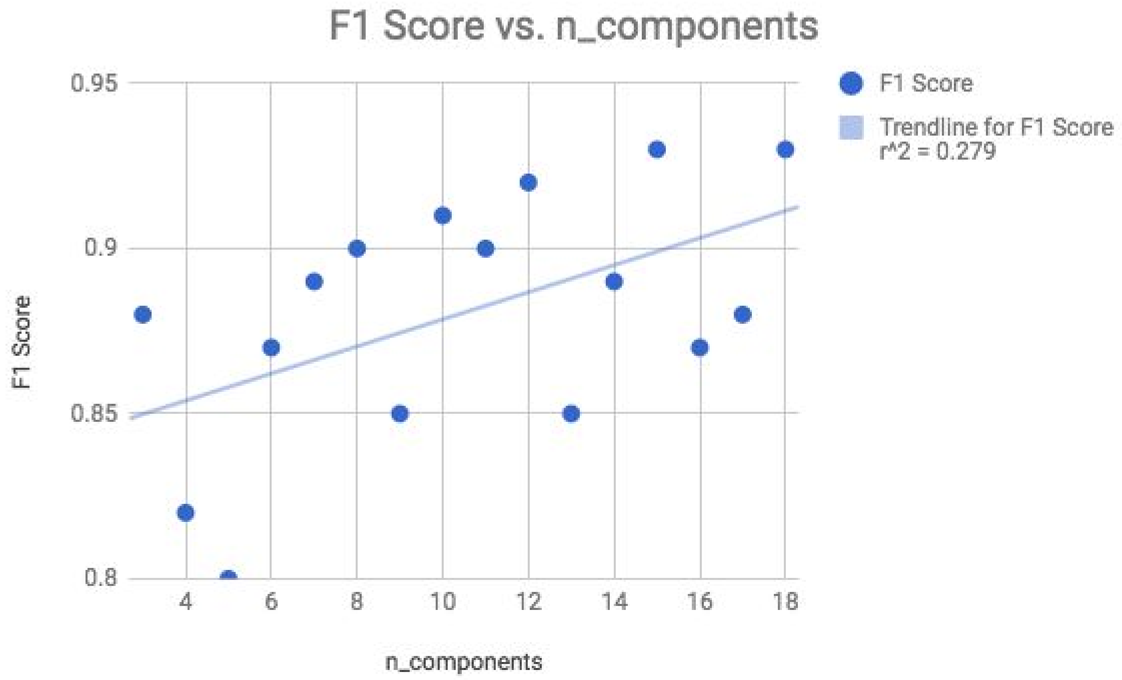
Jonathan Pearce 260672004
Comp 417
Professor Dudek
December 6, 2017

Assignment 4 Discussion

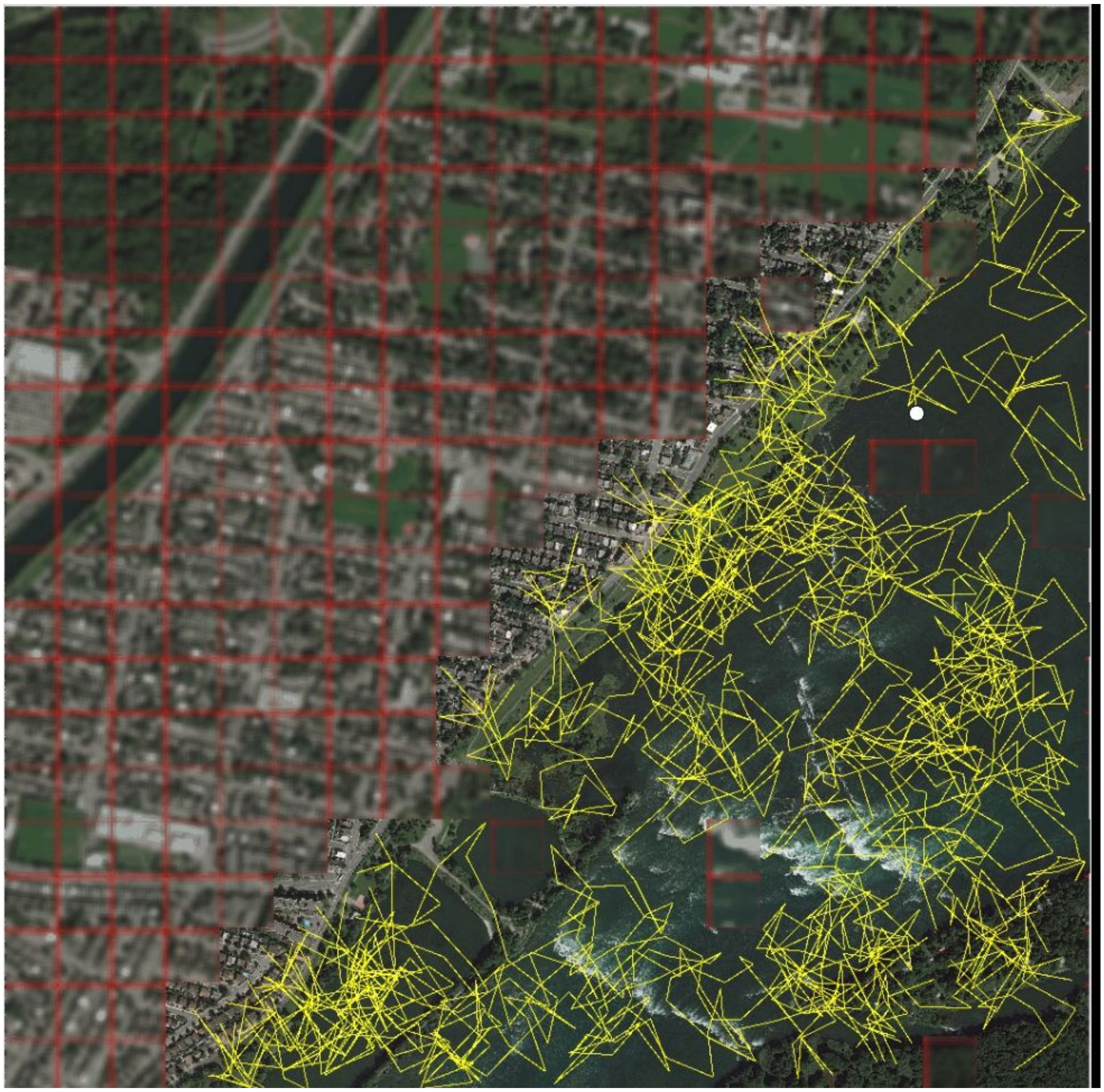
1. PCA Analysis



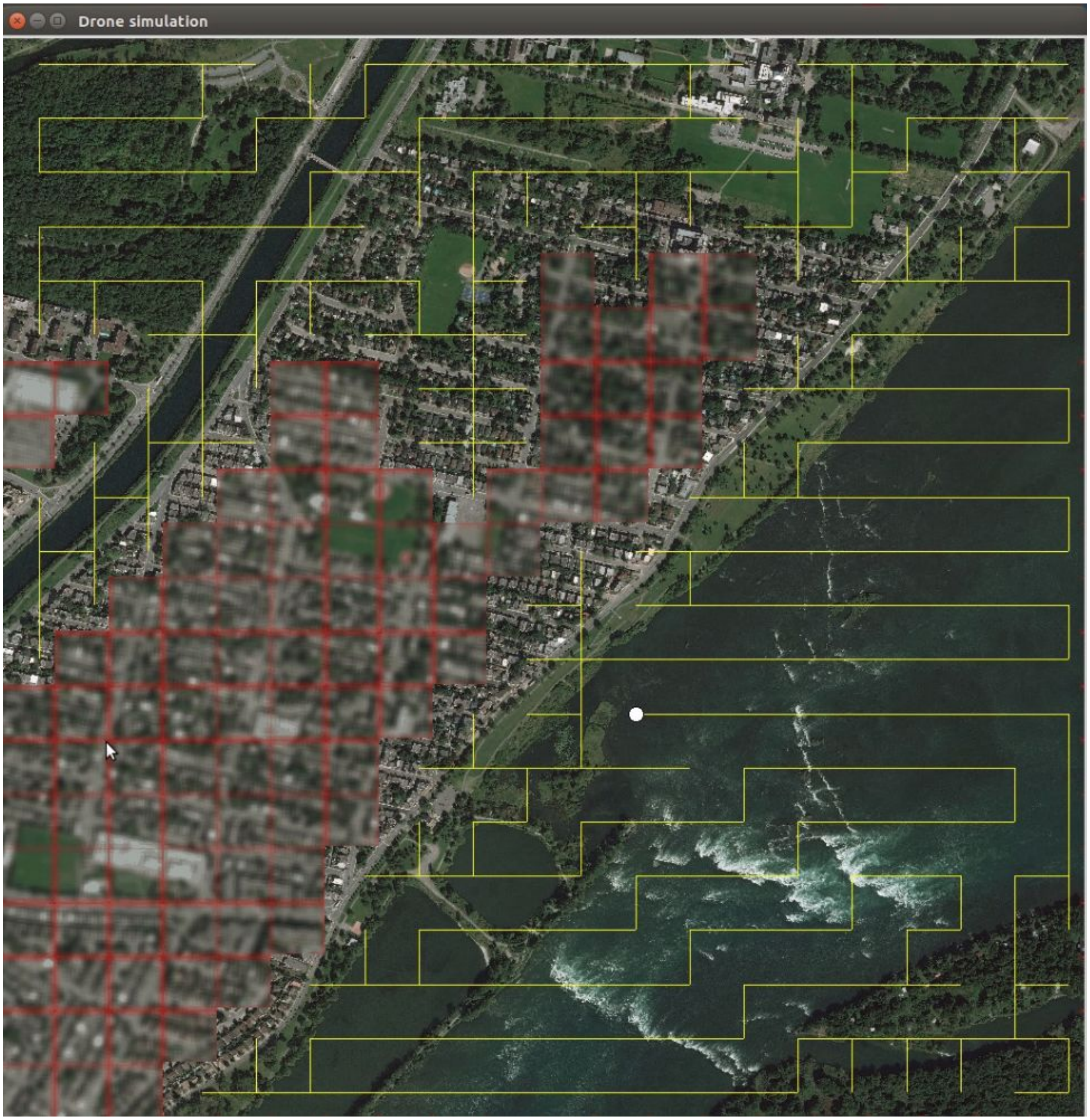
Although the correlation is weak, there appears to be a small positive linear relationship between the number of components parameter in the PCA algorithm and the model's F1 score. This shows us that having a low number of components (3,4,5, etc) for PCA causes an oversimplification of the the data and this results in the performance drop. Where as using a slightly larger number of components (15,16,17,18) seems to do a better job of properly simplifying the data set of images without losing too much of the valuable information needed for classification. From this analysis, using the data and trendline, we conclude that the optimal number of components for the PCA algorithm is 18.

2. Coverage Algorithms

Brownian Algorithm Screenshot



DFS Coverage Algorithm



Note: In my completely random algorithm I would generate a random number from -1 to 1 for tx and ty. However this was far to slow for covering a decent amount of territory so I then multiplied these values by 50 in order to make the video recording process more reasonable. For my Depth First Search coverage algorithm I also changed the movement rate to be approximately the same to the random algorithm movement rate so that the data/results were comparable. My code reflects all of this.

Brownian Algorithm Description: Completely random movement. If an urban tile was detected it would go back to its previous location and generate another random move. If a move was going to push it out of bounds it would generate a different random move until one was found that was within the image

DFS Algorithm Description: Regular depth first search, looks left, down, right then up, if it finds a square it hasn't visited yet, it goes there and marks that it's been visited. If a new square is urban it immediately backtracks. If all 4 options (L, D, R, U) have been visited it backtracks to the square that it came from.

Results

Measure	Brownian (Random)	DFS
Total Number of Tiles Visited	1239	618
Number Urban	135 (11%)	65 (11%)
Number Water	409 (33%)	146 (24%)
Number Arable	695 (56%)	407 (65%)
Distinct Tiles Visited	184	305
Tile Ratio (=Distinct/Total)	0.149	0.494
Total Distance (pixels)	64,174	31,129

The DFS algorithm had the drone travel only half the distance that the Random algorithm had it travel and yet was able to visit 130 more unique tiles in that distance. In this experiment the DFS algorithm was about 3 times more efficient at finding unique tiles while visiting a set number of total tiles (as explained by the Tile Ratio). From this and the images above we can clearly see that the DFS algorithm is much more efficient at covering the map area (as expected). It is also interesting to note that the percentage of total terrain explored (Urban, Water and Arable) remained relatively the same between the 2 Algorithms

Random Algorithm

- Pros
    - Easy to code
- Cons
    - Distinct Tile discovery quickly flattens out since it does nothing to detect whether tile has been explored or not - makes algorithm quite costly if you really need to explore most of the map
    - Can miss tiles that are very close to start position

DFS

- Pros
    - Still relatively simple to code
    - Very efficient at exploring the entire map
        - Achieves this by keeping track of where we have been so we don’t revisit unnecessary tiles