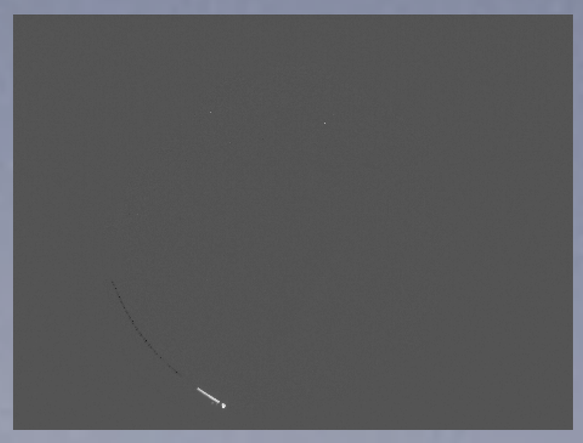
Find Meteors in the Night Sky

Benjamin Cooper
bcooper8@terpmail.umd.edu
Science, Discovery, and the Universe
Double Major in Mathematics and Computer Science

Introduction

The goal of my capstone was to write code which could automatically perform analysis on image data from the UMD Observatory's AllSky camera.At the beginning of the project, we had no automated system in place to capture and analyze image data. We have since found, and installed a program to automate the picture-taking process, however this program had weak meteor detection software and came up with many false positives.I decided to undertake this capstone because I enjoy coding, and knew I could gain a lot of relevant experience from it.

On the left is an image containing a fireball (a bright meteor). On the right is an image containing two airplanes and a meteor.



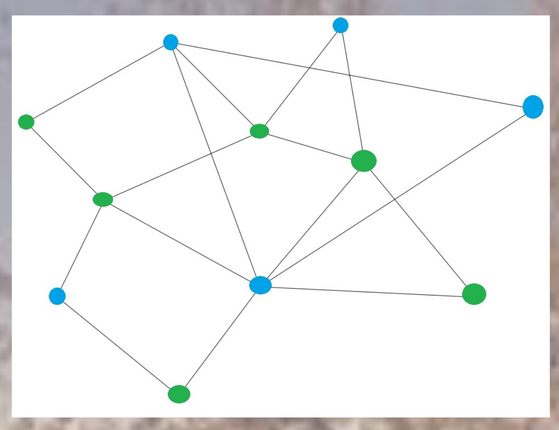
On the left is an image containing a barely visible airplane. On the right is a difference image, showing how the airplane becomes more visible when compared with the next frame.

Methods

- Coded in Python
- Used NetworkX, SQLite, numpy, and astropy modules
- Meteor-Find algorithm:
  - First take difference image between two consecutive frames.
  - Iterate through difference image and find all points that differed significantly from previous image.
  - Perform iterative depth-first-search over those points, to form contiguous shapes
  - Perform linear regression analysis to determine if a shape is meteor-like

Goals

The goal of the analysis is primarily to find meteors, and secondarily to upload images to a large online database, so anyone who wishes can view and download the data. The meteor detection algorithm we already had in place to detect meteors did not work well and came up with many false positives. My code, while slightly less efficient, found algorithms much more effectively.



```python
def dfs_mod(self, graph, node, arr):
    visited, stack = set(), [node]
    while stack:
        vertex = stack.pop()
        if vertex not in visited:
            visited.add(vertex)
            stack.extend(graph.neighbors(vertex))
            arr.remove(vertex)
    return list(visited)

def find_shapes(self, arr):
    graph = self.build_graph(arr)
    shapes = []
    for i in arr:
        shapes.insert(0, self.dfs_mod(graph, i, arr))
    return shapes
```

On the left is an image of a graph, which is the data structure that allows this algorithm to work. On the right is a snippet of code from the algorithm itself.

Conclusions

This project has helped me learn more about coding, and how to use several new standard libraries in Python. In my work, I've learned many new algorithms, and was even lucky enough to invent some of my own. This was the biggest project I've worked on to date, and I'm excited to see how I can apply the knowledge I've gained in future endeavors.

A special thank you to Elizabeth Warner, Dr. Peter Teuben, and the UMD Observatory