2017 Solar Eclipse

Bret Lorimore, Jacob Fenger, George Harder

October 14th, 2016

CS 461 - Fall 2016

Abstract

On August 21, 2017 a total solar eclipse will pass over the United States. The path of totality will stretch from Oregon to South Carolina. The Eclipse Megamovie Project is a collaboration between Google and scientists from UC Berkeley and several other institutions with the aim of compiling a large dataset of eclipse observations. Acquiring coronal data is of particular interest as the corona is not normally visible from Earth. Specifically, the project will crowdsource photos of the eclipse from photographers at various locations along the path of totality. These images will be aligned spatially and temporally and stitched into a unique movie that shows the eclipse over a period of 1.5 hours as it passes across the United States. Additionally, the complete photo dataset will be open sourced so that independent researchers may do their own analysis.

Google will contribute applications providing, among other things, backend image processing, photo upload capabilities, and static informational content. This senior capstone project will consist of two distinct sub-projects, specifically, improving/implementing an image processing algorithm facilitating the classification and spatial/temporal alignment of solar eclipse images before they are stitched into a movie and a location-based eclipse simulator.

I. Problem Definition

A. Eclipse Image Pre-Processor

Solar eclipses offer researchers a rare chance to observe the Sun's corona. Normally only visible with highly specialized instrumentation, the Sun's corona becomes easily visible during a solar eclipse. The total solar eclipse in 2017 will offer solar researchers a unique opportunity to enlist the help of photographers across the United States to gather images of the Sun and its corona.

The Eclipse Megamovie project will crowdsource images of the eclipse and use these to create a movie of the eclipse as it passes over the US. This will enable researchers to observe the Sun's corona over an unprecedented time window. In order to stitch these images into a movie, they must be ordered temporally and resized/aligned spatially. The combination of a wide variety locations and angles at which the crowdsourced images will be taken, the lack of reliable GPS and time-stamp data on most modern DSLR cameras, and the sheer volume of images that will be collected makes this a non-trivial problem.

B. Eclipse Simulator

The Eclipse Megamovie website eclipsemega.movie, in addition to enabling photographers to upload their photos during/after the eclipse, will provide information for photographers and the public to learn about this specific eclipse and eclipses in general. While this eclipse will be visible from much of the United States, only select areas lie in the path of totality. This motivates a location based eclipse simulator allowing users to preview what the eclipse will look like from their specific location before it happens. This will not only increase engagement before the eclipse, but will enable users to better select a viewing location. Ideally, this simulator will also include location-specific contextual information, so that users can determine whether they will have a desirable vantage point from their selected location.

II. Proposed Solution

A. Eclipse Image Pre-Processor

The eclipse image classifier application will ingest photos of the eclipse and align them spatially and temporally so that they are ready to be stitched into movies. With images sourced from photographers all with different camera equipment/configurations and with no guarantees of accurate timestamp and GPS coordinate data, this is a challenging problem.

We plan to create a standalone C++ application using existing image processing implementations including OpenCV and other libraries wherever possible. Specifically, this application will ingest eclipse images, perform the processing described above, and export them along with any meta-data required to compile them into movies. This application will not perform any of the stitching into movies itself, nor will it interface with Google Cloud Storage, where the eclipse images are stored - it will simply be a binary that can be invoked by some larger system developed by Google.

B. Eclipse Simulator

We plan to explore two general approaches when building the eclipse simulator. The first and ideal approach is to use the Google Earth API and render the eclipse within a 3D Google Earth window using an image sprite overlay. This approach will give users unprecedented context as to what the eclipse will look like from their selected location. They will be able to tell, for example if there is a skyscraper, or football stadium that may inhibit their view of the eclipse. It is currently unclear whether or not this approach will be possible with the Google Earth API. If it is not, we will implement a 2D, stylized simulator. In this case, Google will provide the visual design for the simulator, and in either case, Google will supply all the visual assets necessary to build it - including images, SVG graphics, etc.

Whichever solution is used, either 2D or 3D, the eclipse simulator will incorporate a time slider that allows users to view the eclipse up to 12 hours before the eclipse and 12 hours after it. This will provide users valuable information in addition to location information that is already being provided. To enhance the user experience even further, as a user changes the time on the slider the eclipse's appearance will change in an animated fashion.

III. Performance Metrics

A. Eclipse Image Pre-Processor

Given the large amount of images that users will be uploading to the site, each image must be processed accurately in a reasonable amount of time. Each image should not take more than 5 seconds for pre-processing. While 1 second per image is ideal, quality is of high value for this project and we will likely face a performance-quality tradeoff.

From a quality perspective, the pre-processor must identify solar/lunar disks with 95-98% accuracy. It must also be able to classify images as being of full-disk/crescent/diamond-ring/total eclipses with 95% accuracy. These accuracy metrics will be computed from a set of hand labeled, golden data. Achieving these accuracy metrics is key to ensure the movie that is exported is smooth and accurate.

B. Eclipse Simulator

For the Eclipse Simulator we have several metrics to determine if we created a useable solution. These metrics deal with timing and the speed at which the simulator is responding to user interactions. First, we want our eclipse simulator to load within 150 milliseconds. This load speed makes users more likely to use our simulator and gives users a positive impression of the site. Next, our goal is for the simulator to be running at 60 frames per second (fps). This metric relies on two factors: the hardware on which the simulator is running and which style, 2D or 3D, we implement. For a 2D solution we should achieve 60fps on any hardware. If we build a 3D implementation, the speed will rely somewhat on the user's hardware. Whichever solution we implement, and no matter the hardware a given user has, the simulator should not fall below a 30 fps.

David Konerding, Project Sponsor	Date
Bret Lorimore	Date
George Harder	Date
Jacob Fenger	Date