CS486C –Senior Capstone Design in Computer Science
Project Description

Project Title: HydroCams Smart Survey Tool	
Sponsor Information:	Dr. Eck Doerry, Collaborative Computing Lab (CCL) School of Informatics, Computing, and Cyber Systems (SICCS) Northern Arizona University Eck.doerry@nau.edu

Project Overview:

Flooding is easily the most damaging natural hazard in the U.S. and around the world, with flood damage in the United States alone hovering around an average of $3.7 billion each year. With 99% of U.S. counties regularly affected by flooding, and an average of 120 deaths/year due to flooding, the human impacts of flooding are devastating. These figures have trended upwards in recent years, as the impacts of climate change exacerbate flooding frequency and severity, a trend that is predicted to accelerate in the coming decades.

We can't prevent the increasing severity of weather events, but modern IoT, big data, and cloud computing technologies hold tremendous promise for predicting flooding, informing effective flooding response, and keeping citizens out of harm's way. Our FloodAware project (https://floodaware.net/) contributed to this effort by exploring how "smart cities" could utilize cameras and network technologies to monitor urban flooding and warn citizens and first responders in near-real-time, with a central focus on images pulled from hundreds of traffic, agency, and public cameras available in many areas. The HydroCams project is a follow-on project funded by the USGS to further develop and field deploy one particular element of FloodAware: the solar-powered, cell-connected smart cams we invented that can be cheaply and easily installed to monitor for flooding anywhere.

The basic operation of a HydroCam is simple: it is mounted on a fixed structure (e.g. a pole) pointing at a "target area". Within this picture frame, a number of "gauging points" are identified; these could be a staff gauge in the image…but also could be dots painted on a wall, or even natural features in the image (knotholes on a tree, cracks in a rock) that are easily detected...and whose elevation above some zero point is known. Image processing can then be used on a particular image to

detect which of these gauging points is obscured (i.e. by being under water), allowing fairly accurate estimation of water levels at the site.

Ease of site installation is a major goal of HydroCams; it should be possible to install a site in an hour without the need for highly trained personnel or specialized equipment. A major obstacle remaining in this regard is the need to determine the height of the gauging points in the image relative to some defined zero point, and this is where this Capstone project comes in.

The Envisioned Product

The goal of this project is to explore the potential of image-based site metrology by developing a software tool for measuring the distances between the gauging points found in an image. The basic workflow we envision is simple:

1. the user (Hydrocam installer) installs the camera and points/frames/focuses it on the target area. A special object of known size is placed in the image at the zero point.
2. They then take out their smartphone and take a picture or short video clip of the target, standing by the installed camera.
3. This image (or video) is fed into the envisioned software tool, which detects the gauging points, detects a special zero point, and then returns the relative distances/elevations between them all by using geometric image processing techniques.
4. A structured "calibration file" (e.g. json) of relative elevations is returned and can serve as the basis of determining water levels in imagery received from the camera.

The basics here are super simple: it's child's play to calculate distances in pixels between points in an image. The challenge is actually making this useable technology: How to efficiently detect/ID gauging points, and how to figure out the scaling of distance and depth in the image to get an accurate real-world measurement. Possible solutions can be described at several levels of quality and complexity:

Level 0: a minimum viable product

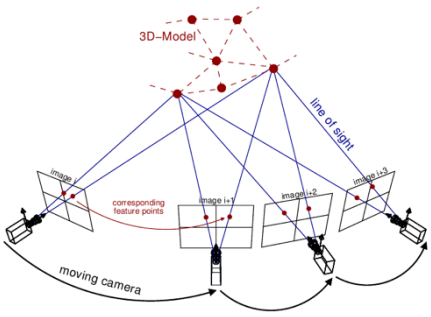
- A simple program that you launch in a terminal window, that opens a simple visual "workbench" for doing the image metrology. It shows the input image and related tools.
- The workbench allows you to zoom in/out on the image, and then outline the gauging points manually, e.g., by drawing a little box around the point in the image.
- Allows you to create/edit/delete the gauging points freely.
- Manually putting in the distance between any two gauging points results in all other distances being calculated automatically. This is a dumb Fred Flintstone solution to relating pixel distances to real-world distances.
- Can output the calibration file with all measurements on demand.
- Demonstrated accuracy at various image resolutions, scales/distances, and sizes of gauging points

Level 1: A real usable solution. In addition to above…

- Can also automatically detect the gauging points in the image. In practice, gauging points are often squares or dots of a particular color. It should detect and suggest these points when the image is input. User can still delete erroneous ones or add others.
- Can locate the special zero point scaling object in the image. Based on the (known) size of this object, it automatically calculates distances between all gauging points.

- Supports tuning: can specify the color (range) of the gauging points to look for…and provides various tuning tools to make that accurate in different lighting conditions, etc.
- A performance analysis/scoring tool to easily compare versions/algorithms you develop. Operates on a folder of images, each of which already has a calibration file, i.e., one that you've carefully measured by hand in the field (the "ground truth"). It runs your tool on all of them using your metrology algorithm, and compares your computed results to the actual measurements and reports out accuracy.
- Implemented in a simple Android app. Simplifies the process above by locating it on your phone: You can use the Android app to take the picture, and it runs the image processing to detect the gauging points and display the calibration info. Which could then be forwarded (text or email) somewhere as a textual calibration file.
- Implemented in Android first…but using a cross-platform dev framework like Ionic or React Native, for easy generation on iOS app from the same codebase later.

Level 2: to make it truly useful and deployment-ready



- The basic versions use simple 2-D geometry to calculate distances; this assumes that all objects are in the same plane (distance) from camera…which would result in significant error if gauging points were at varying distances (depths) from the camera. A much better approach would be to incorporate a technique called "structure from motion" (https://en.wikipedia.org/wiki/Structure_from_motion), where you essentially can calculate 3D structure by using multiple images from differing angles. There are multiple tools and packages for this already, and this is a relatively simple application of it. The user flow would now look like this: user stands at camera location and takes a short video clip, swiping their phone across the target area (like taking a panorama picture) or by simply taking two different pictures stepping a step or two sideways between them. You'd then use the multiple images to drive the Structure-by-Motion calculation.
- A working iOS version of the app. This should be fairly simple if you've used a cross-platform dev framework to start with.

A successful product will provide a proof-of-concept prototype that realizes the potential of this simple alternative to complex, spendy and time-consuming surveying. This solution would ultimately be integrated into the Hydrocams smart camera solution that we are working to produce, greatly streamlining the installation process.

Knowledge, skills, and expertise required for this project:

- Interest in and basic knowledge of image processing techniques and frameworks like OpenCV.
- Knowledge of mobile application programming frameworks, with particular emphasis on cross-platform frameworks like Ionic, Flutter, or React Native.
- Knowledge of geometry and interest in learning about cool techniques like Structure from Motion.

Equipment Requirements:

- There should be no equipment or software required other than a development platform and software/tools freely available online.
- Cell phone to take images to use in testing.
- Any other items (e.g. measuring tape, material for target gauging spots, potentially a tripod, etc.) required to complete the project will be provided by the client.

Deliverables:

- The software applications as described above, deployed and tested successfully with real data. Must include a complete and clear User Manual for configuring and operating the software.
- A strong as-built report detailing the design and implementation of the product in a complete, clear and professional manner. This document should provide a strong basis for future development of the product.
- Complete professionally-documented codebase, delivered as a repository in GitHub.