Technology Feasibility Document for Team "Fish Out of Water"

10/18/2023

Project Sponsor: John Fennell

Faculty Mentor: Sairsri Muttineni

Team Members:

Jack Shanley

Corey Moreno

Nicholas Robishaw

Jaron Bauers

Table of Contents

Introduction

Deep within a canyon on the border of Northern Arizona and Southern Utah resides the rainbow trout. Anglers from all over come to Lee's Ferry, a well known name in the trout fishing world, to board a boat onto the Colorado River with hopes of catching these trout. The Arizona Game and Fish Department has to keep a watchful eye over the fishing that happens at Lee's Ferry to ensure that the native trout population maintains a healthy number. The rainbow trout is an integral element in the ecosystem of the Colorado River, thus it is important to support the conservation of their species, as well as other species of fish that are there.

Implementation of a monitoring program has been a bit challenging, Arizona Game and Fish (AZGFD) sends representatives out a few times a month to interview anglers in person about their experience on the water. They ask them many questions including how long they were out and how many fish they caught. But the reality is that there is not enough time nor resources for the AZGFD to send representatives out there enough for them to get an accurate depiction of the environment that would allow them to gain insight into the status of the trout. In order to promote conservation for the trout, they need more data. In an effort to solve this problem, Arizona Game and Fish installed a game camera not too far from the dock to capture images of the dock every thirty seconds. This results in over 1,800 photos taken everyday. The addition of the camera has helped our client, John Fennell, and his team in procuring data for their research and analysis. The pictures contain boats and kayaks which can give them a better idea of how much traffic the river is seeing and how much fishing is going on. But there is still a problem, John and his team have to manually look at all of those photos to pick out the ones they need. Naturally, this is a very tedious task as it can take over five hours just to get through one day's worth of photos.

This is where we and our solution, "Fish Out of Water", comes in. We intend to create a desktop application that can perform automated analysis of these images, largely removing the human element. The application will be able to automatically identify and record how many boats it sees and it will link that information to a database that John and his team will interface with. If there are images that the program cannot confidently decipher, then it will flag them for manual review. This will hopefully cut down the photos that need to be reviewed from 1,800 to a few hundred or less. And ultimately, this will save John, his team, and the AZGFD countless hours of having to sift through thousands of pictures on a daily basis. This will give them more time to do their research, which is important work. Their research is directly impacting the quality of both the fishing experience and the awareness of overfishing/underfishing of the trout. Several hours to get through one day of photos can significantly slow down their process, we aim to change that.

Having explained the current situation up at Lee's Ferry, including our client's mission as well as their observed problems and inefficiencies within their processes and tools, next it is important to discuss some of the technical challenges that we are anticipating on our end. This project presents some unique challenges that will be discussed in more detail in the following section. We intend to discuss what they are and later on we will discuss how we plan to overcome them.

Tech Challenges

General challenges:

- Due to the theft concern and that Lee's Ferry is a part of the park service the AZGFD has placed the camera at a very far distant vantage point.
- Our program needs to be able to analyze photos with high glare, low light, high exposure, etc.
- Creating our output excel file to match the format for their database so the department will not have to update their end of the database.
- Gathering enough initial pictures to use for early development of the program and or train the program what to recognize.
- If pictures cannot confidently be run through the program and get an accurate output, the program will flag the issue for an operator to review the photo manually.
- With the poor camera placement it would be challenging to see any kayakers entering the colorado from behind the bush concealment.
- With the poor camera position it would be hard for the program to accurately keep track of the kayakers due to the poor resolution.

- Creating an object variable for each boat to house the arrive and departure time from boats on the colorado river.
- Creating a simple desktop application with an easy to use interface for officers to navigate and operate.
- Training the program to recognize the commercial rafts and skip them when counting residential boats.
- Avoiding duplication when counting boats and not confusing one boat's departure time from another.

Potential camera relocation:

- Finding a way to blur faces and license plates if the camera gets moved closer
- Finding a position where the camera will not get stolen or impede on park service land
- Finding a position where the camera has a clear view of the boat ramp

Current camera location:

- The current problem with the camera placement is that it is very far away from the boat ramp and it's in a position where the photos cannot clearly see who is coming and going from the boat ramp.
- There is a line of natural brush/trees that conceal the boat ramp from the current camera position.
- With the poor resolution of the photos it would be challenging to uniquely identify each boat to store as an object for the program to identify and edit later.
- Since creating a constructor for a boat/kayak object will be very difficult due to the resolution and camera placement there will need to be a new way used to correctly store and uniquely identify each boat/kayak.

Bare minimum program:

- The customer will accept an alternative program that will just flag each picture in a certain time interval when a new boat/kayak is identified. Customers would really like to slim down on the amount of pictures an operator has to go through when working on this task. This will allow the customer to gather more information to store in their database since it would take a quarter of the time. Our program should be able to at least cut down the amount of pictures an operator has to go through by around 80%.

Tech Analysis

Intro

The challenge we have at stake is to be able to make our program recognize when boats and kayaks leave the boat ramp and get onto the water. The program will also have to be able to count when a boat or kayak leaves the boat ramp and when it returns. Not only will our program have to be able to tell the difference between a boat and a kayak, there are also some plants covering the boat ramp from the perspective of where our camera is set. Also, we have to consider that the time of day will affect the glare and overall quality of the images that will be coming through to the program.

With all of this in mind, we will need a multitude of functions to consider making. One function will be the main image reading function. This will simply take in a photo and read what the photo looks like. Another function that will be necessary is a findBoat function that will check the photo to see if it can recognize a boat on the screen by comparing it to a database of what a boat should look like. We will then need a function that can check if there is any uncertainty from the photo and be able to flag it for the user to check manually. Lastly we will implement a function that will recognize when a boat is leaving or arriving at the dock based on the objects placement over certain frames.

Desired Characteristics

The desired vision for the program is to make a simple-to-use program that significantly decreases the number of photos and documents the operator has to analyze and document. Fish

Out of Water looks to package this program as a desktop application that AZGFD computers will be able to download. The application will need to be easy to use since multiple people will be likely to work on this task as they get time throughout the week. Most of these individuals will not have much experience and will need an easy-to-navigate program to successfully complete the task in a timely manner. This program will be very cheap and will require very little maintenance due to its simplicity. If maintenance is needed, Fish Out of Water will keep the source code easy to understand, very organized, and have plenty of documentation comments that explain what a code block will execute. Since the program will be very simple and efficient, it will require a very low amount of resources from the computers. Since our development team does not have much information about the computers used at the AZGFD, the program will be portable and can be used on almost any Windows PC.

Alternatives

One way we could fix some of these issues is moving the camera to a closer location than where it already is. With the distance between the camera and the boat ramp, sometimes glare and pixelation can cause visibility issues, which could take a toll on the accuracy of the program. Moving the camera closer would raise better pixel quality allowing the program to read the incoming photos more accurately. The downside to this would be that the Arizona Game and Fish Department at Lee's Ferry does not want any faces, license plate numbers, or boat registration numbers visible to the camera or to exist in any of the photos. In addition to that, where the camera is located is arguably the safest spot for it so the camera does not get tampered with or stolen.

Our client has also recommended us looking into using a program called Timelapse, which can take in the collection of photos and automatically compile them into a timelapse format. An idea we had was to create a software that can interact with Timelapse to do image recognition. Timelapse offers a feature where the user can highlight the changes between photos throughout the timelapse. This could be a helpful feature in implementing our algorithm to handle counting the boats and kayaks that leave the boat ramp. However, it could be a challenge to create a software that interacts with an existing software without any of the source code, which is unavailable to the public.

After doing research on Python libraries that could help us for this project, we came across a library called openCV2. This library will be able to give us the functionality required for the program to read the images and pinpoint any boats. Using this tool is simple and contains all of the functions that we need to be able to implement this feature. OpenCV2 is the updated version of openCV, where the newer version has more capabilities of integrating AI for image recognition. It was created by some engineers at Intel in the early 2000s and has been updated to more advanced features along the way. Corey found this library from a previous project he made for an AI game recognition software that he made which uses a very similar tactic needed for this project.

Analysis

As previously mentioned before, we are not able to bring the camera any closer to the boat ramp due to privacy issues, so we cannot implement any testing or analysis for that aspect. As for the openCV2 library, we did some testing on the video game Minecraft. We made a small database that contains different pictures of a redstone block and had the program try to display a flag when a redstone block was on the screen. After some tweaking, the program worked successfully. Whenever a redstone block was on the screen, the program would output that it could see the block and whenever we looked away from the block, the program stated that there were no blocks found. With this in mind, it seems that openCV2 is a reliable library to do this project with.

Chosen Approach

Some of the pros of this approach is that we will not be required to create any sort of plugin or low-level functions to get Python to read the pictures. Since we have the openCV2 library available and it is free, we can get started with the image recognition as soon as we want to. Another benefit with this approach is that training our program to recognize a boat will consist of a folder of a ton of pictures from different angles and qualities. All we have to do to train it is compile a bunch of screenshots of boats and kayaks, and the program will automatically compare every photo.

One of the cons of this approach is that we don't know at this moment how many screenshots of boats or kayaks we will need for it to be sufficient at comparing the photos. We will have to make adjustments to find out which type of photo will be the best kind for Python to recognize a boat. Another con to this approach is the possibility of false positives. Luckily, openCV2 has a variable called the 'confidence level', which ranges from 0 being nothing like the picture and 100 being exactly the same picture. If we can find the golden ratio of the confidence interval, we can have the code make a flag when it reaches below a certain threshold. This is as simple as changing an if statement to check if the confidence level is below a certain level.

	Easy to implement	Ease of operation	Portable	Speed	Maintainable
Moving Camera	0	0	0	0	0
Timelapse Extensions	0	0	0	0	0
openCV2 image recognition	0	0	0	0	0

Scale: Red - Poor, Orange - Average, Green - Excellent

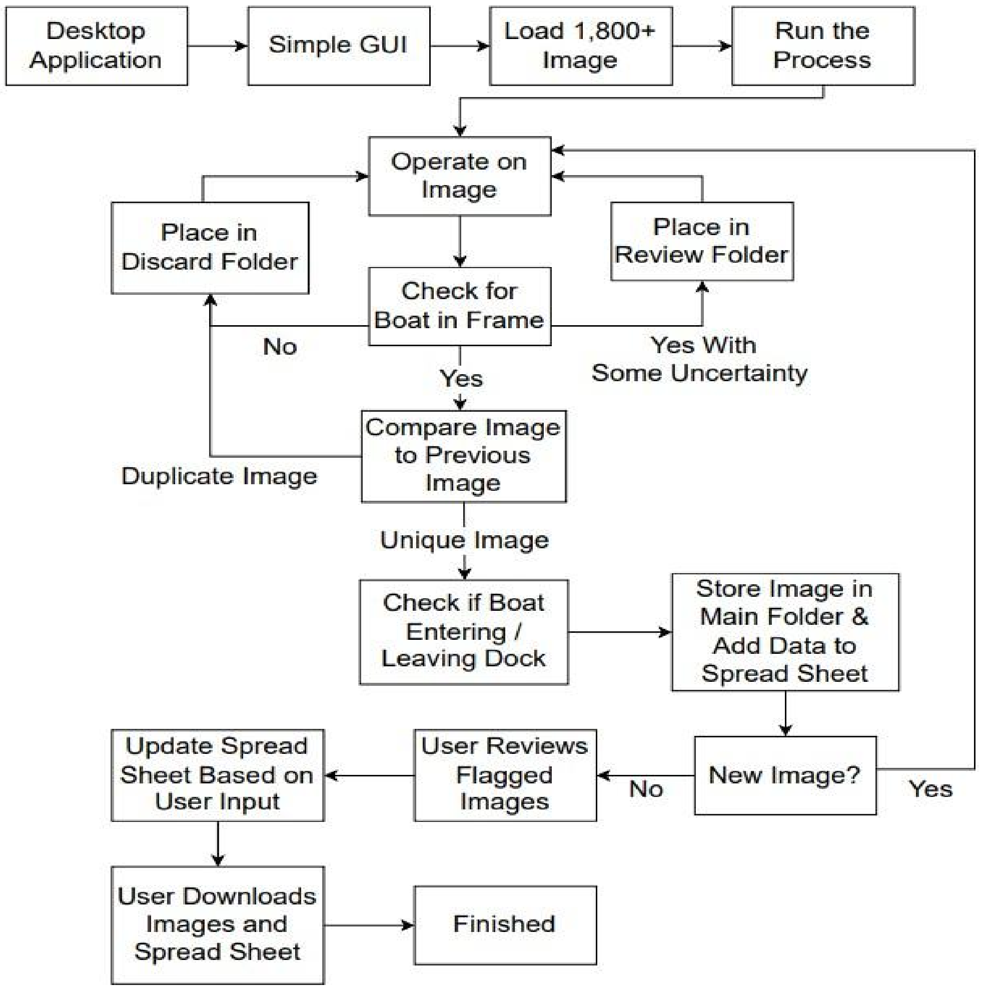
According to the options we have considered, it is clear that openCV2 is our best choice of implementation. In any route, implementing this project will be a challenge, but considering the desired characteristics needed, openCV2 is the best option. One thing to consider is the feature from Timelapse where it can highlight changes in the timelapse. It could be useful to implement that feature in our final program.

Proving Feasibility

There will be multiple demos planned before the final release of the Fish Out of Water program. These will include a simple demo showcasing the confidence intervals, which will be the main decision-making function that will sort the pictures. If the complete autonomous program is approved by our customer, the development team will prepare a demo to show off the privacy feature. The privacy feature will analyze a picture, look for any sort of vehicle license plate or a person's face, and blur them out of view. A demo that will be demonstrated very early on will be the functions that will be in charge of sorting out the pictures with no wanted objects. The wanted objects will simply be boats and kayaks, so if the program cannot confidently identify at least one of those objects, then it will be flagged and sorted into a junk file. After the previous demo, the development team will need to create a demo to successfully limit the number of duplicate pictures within a certain time frame. These duplicate photos are ones where there are no new objects within a certain range and will be sorted out into a junk file. Lastly, a demo will need to be created for the output of the program. This output will need to be an Excel file and will need to be as close to the existing Excel files the customer creates for the AZGFD database as possible.

Tech Integration

We acknowledge that there are many roadblocks ahead of us and we have accounted for them. We believe that our approach will be sufficient enough to cover for all of the problems we know we will encounter. Now it’s time to put this project together and be able to react and adapt quickly to any problems that may arise along the way. Below is a diagram of how we envision our desktop application to run based on our requirements.


Desktop Application
Simple GUI
Load 1,800+ Image
Run the Process
Operate on Image
Place in Discard Folder
Place in Review Folder
Check for Boat in Frame
No
Yes
Yes With Some Uncertainty
Compare Image to Previous Image
Duplicate Image
Unique Image
Check if Boat Entering / Leaving Dock
Store Image in Main Folder & Add Data to Spread Sheet
Update Spread Sheet Based on User Input
User Reviews Flagged Images
No
New Image?
Yes
User Downloads Images and Spread Sheet
Finished

Our program will be run on the users Desktop where they will be able to dump thousands of images into the program to be processed by our program. We plan on building a simple GUI using one of Python's many available frameworks to achieve this. After the user dumps their images into our program it will complete multiple checks including if a boat is in frame, unique images, and whether the boat is arriving / leaving. There will be three folders where we store the images that have been discarded, archived, or flagged for review. Once our program has iterated through every image the user will then be able to sort through the flagged images and update the spreadsheet as needed. Finally our program will provide the user with the completed spreadsheet along with the sorted images. The Spreadsheet will include data relating to the time stamp of the images that were discarded, archived, and whether the boat was leaving or arriving at the dock. Hopefully this process will decrease the processing time tremendously.

Conclusion

The Arizona Game and Fish Department's (AZGFD) mission is to conserve diverse wildlife resources and manage for safe, reusable outdoor recreation opportunities for current and future use. The AZGFD run into many different challenges when it comes to achieving their mission while they work with the unpredictable "mother nature". This is why they focus their attention more on the problems they can control rather than the ones they cannot. When it comes to preserving the native rainbow trout of the Lee's Ferry Fishery, the AZGFD are able to monitor and control the amount of angler boat traffic coming in and out of their dock.

As you may know, when one new solution comes into action it can bring multiple problems along with it. The process that the AZGFD has set into place to monitor the amount of angler traffic the Lees Ferry is getting is way to time consuming of a task for a human. The monitoring system in place takes more than 1,800 pictures a day and a human needs to fish through each image to tag every boat, kayak, and trailer that enters and leaves the dock. This process takes at least 5 hours to complete only one day's worth of images captured. The new problem the AZGFD has introduced is that they've created more manual work for their workers.

Our team, Fish Out of Water's goal is to cut down the amount of images our customer has to look through and ultimately try to automate this process the best we can. We hope to solve their time consuming problem by developing a desktop application that will process these images for them hopefully with little to no human interaction. Our program should create a summary of its findings and display this angler boat traffic data in an excel spreadsheet to be easily consumed by the AZGFD. We hope this program will be able to save them hours of work so they can shift their attention towards other projects that align with their mission.

Fish Out of Water is very passionate about this project being all members of the group having a love for fishing. The team only wants to see the fishing environment at the Lee's Ferry and other fishing spots of Arizona thrive and flourish. We will be hard at work this upcoming year to provide the AZGFD with this autonomous program for them to utilize for years to come. We are highly confident that our current proposal above will be sufficient to solve their problem and are all eager to get to work on our solution.