Nature Photography Robot

Justin Kiefel, Sidhant Motwani, Fernando Paulino

Department of Electrical and Computer Engineering, Carnegie Mellon University

Abstract— **Nature photography is a mundane and time-intensive process. Photographers must wait for sparse animal appearances and spend even more time editing the photographs. Remote-controlled photography robots exist, but these systems still require constant human attention. Our solution to this problem is a robotic system capable of performing nature photography and photo editing. We propose a photography pipeline, where the robot searches for, tracks, and photographs animals then performs automatic editing.**

Index Terms— **Computer vision, design, motion planning, object detection, photo enhancement, robot, search**

I. Introduction

Animal photographs are widely used across the internet and social media. From advertisements to raising awareness for conservation efforts, there is a clear demand for high-quality animal photos [1]. However, the process of acquiring these photos is far from easy. Animals are constantly moving and often avoid humans. The photographer may need to wait an extended period of time to have the opportunity to capture a photo. Afterward, more human effort is required to edit the photos. The cost of human time and labor in this process is undoubtedly high.

As a solution, we propose a photography robot, which can find and photograph animals. This robot will be designed to sit stationary in nature. It will then rotate to capture images of animals in its environment and automatically edit the photos. While a photographer needs to be paid per hour or day, a robot only requires a one-time payment. Furthermore, a robot will not get distracted or tired while waiting. This difference will enable companies to search for photographs for longer hours and on more days. The reduced costs may also allow companies to purchase multiple systems and survey a larger area for the same cost.

The idea of photography robots is not new. For example, a remote-control buggy has been used to take photos of dangerous animals from close up [2]. In our research, we found many examples of remote-controlled photography robots, but this approach does not solve the problem of high human labor costs. Furthermore, the autonomous photography systems we found were for very simple applications, like photographing a stationary object from a close distance [3]. Our system extends the idea of autonomous photography to solve a shortcoming of modern animal photography. Our goal for this project is to prove that this approach is feasible by producing a functioning robotic system.

II. Use-Case Requirements

To adequately emulate the capturing of animals in nature, the system must be able to detect animals up to 25 meters (the necessary distance to photograph birds in nearby trees) away with a recall rate of 75%. Recall is a more important metric than precision or accuracy, because animal appearances are sparse and removing irrelevant photos is a quick process. An autonomous system with 75% recall would photograph as many animals as a perfect human would in just 33% more time. While humans certainly do not have 100% recall, we decided that this is a reasonable tradeoff considering the reduced human effort.

Animals are not stationary, so the detection must happen in a timely manner and the robot must follow animals after detection. We decided that the system must detect animal within 15 seconds. Photographing a running animal or flying bird is difficult even for many humans (especially when done without a professional grade camera), but the system should be able to follow and photograph a walking animal. A walking animal moves approximately 2m/s and could walk the entire search diameter in 25 seconds. Assuming animals will only walk through the center of our search radius is unreasonable, so we decided that 15 seconds is a more practical time window. Furthermore, the robot should be capable of following an animal moving at 2 m/s to continue taking pictures.

Performing professional level photography will be difficult, as the cost of most DSLR cameras far exceeds our budget. However, the most common and accessible form of photography is phone photography, and we think this is an obtainable goal. Most modern phones have 8-12MP cameras and in-app editing software. In addition to the technical capabilities, human photographers have the ability to properly zoom and focus when capturing photos. Along with being shot with an 8MP sensor, the photo should be of a quality indistinguishable from a human shot and edited photograph. To quantitatively measure this, we will have human testers attempt to distinguish our photos from human captured photos. These testers should not do better than guessing (50% accuracy) with any statistical significance when labeling photos as robot or human pictures. By this metric, the system's camera should also be capable of at least 2x optical zoom. The system should also be able to pan its camera 360° in the x-direction and 180° in the y-direction to be able to track/detect all accessible animals in its vicinity.

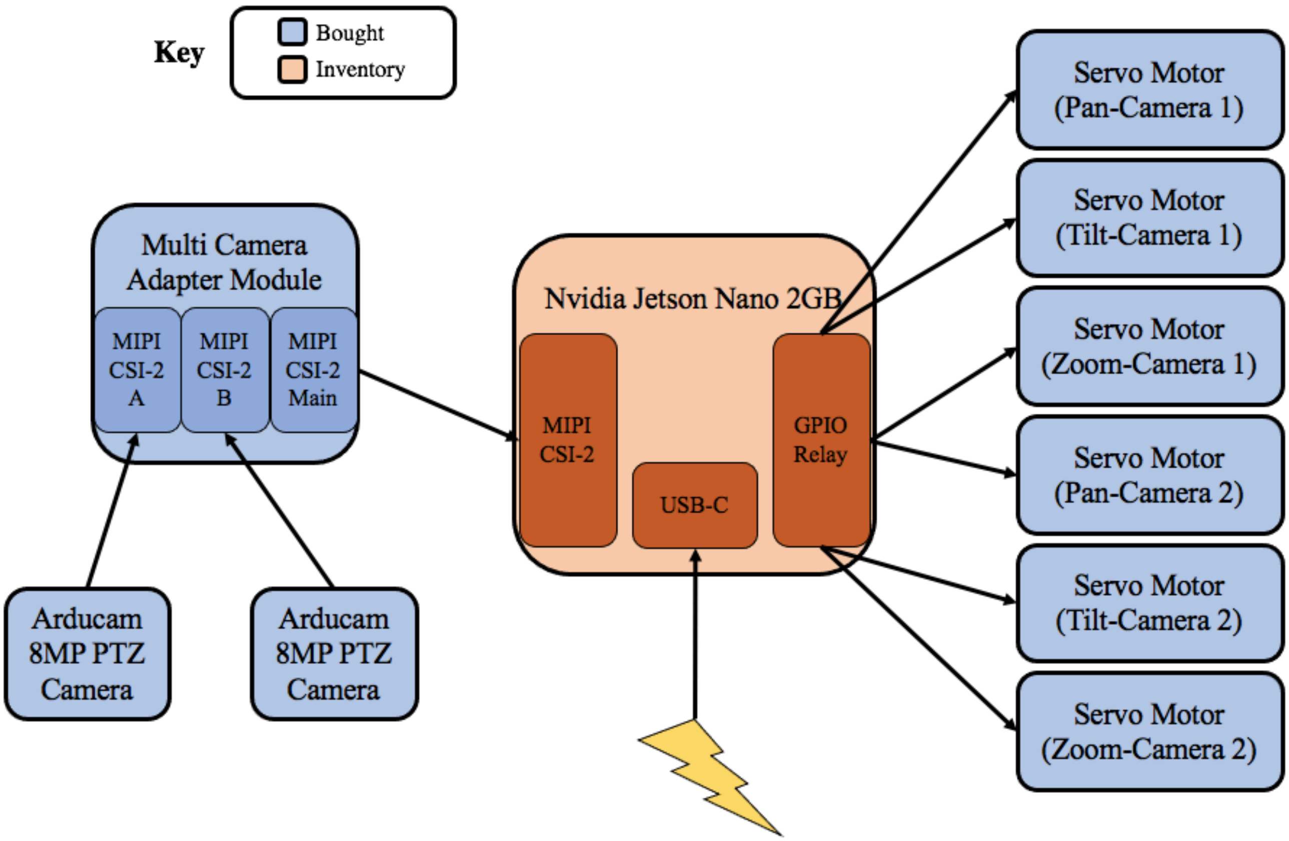

Key
Bought
Inventory
Multi Camera Adapter Module
MIPI CSI-2 A
MIPI CSI-2 B
MIPI CSI-2 Main
Arducam 8MP PTZ Camera
Arducam 8MP PTZ Camera
Nvidia Jetson Nano 2GB
MIPI CSI-2
USB-C
GPIO Relay
Servo Motor (Pan-Camera 1)
Servo Motor (Tilt-Camera 1)
Servo Motor (Zoom-Camera 1)
Servo Motor (Pan-Camera 2)
Servo Motor (Tilt-Camera 2)
Servo Motor (Zoom-Camera 2)

Fig. 1. The hardware layout for our nature photography robot

III. Architecture and/or Principle of Operation

Figure 1 depicts a software diagram for the system's principle of operation. The camera pans and scans for animals in its environment. In the case of multiple animals in frame, the system chooses the first detected animal. Once an animal is detected, the system's KLT begins to track the target and attempts to place the animal in focus using Camera 2, assuring that it is appropriately centered and that the camera has zoomed in so that the target is occupying approximately 40% of the photo frame. A set of photos are quickly taken and then run through the system's photo editing algorithm to produce a more professional grade photo of the target animal. If the target exits the KLT frame of tracking and is lost, the KLT is halted and Camera 1 begins to search for an animal target again.

Figure 2 contains a hardware diagram of the systems architecture. Camera 1 and Camera 2 are the two Arducam 8MP PTZ cameras that are each connected to a MIPI CSI-2 port on the Multi Camera Adapter Module. This module is used to compensate for the lack of multiple MIPI ports on the Nvidia Jetson Nano. The adapter module connects to the singular Nvidia Jetson Nano MIPI CSI-2 port for the sending of images and video and power. The Nvidia Jetson Nano draws power from its USB-C port. All other electrical components connect to the GPIO Relay ports on the Nvidia Jetson Nano. Each camera is mounted on a PTZ stage consisting of 3 Servo motors (one motor for each of the pan, tilt and zoom functionalities of the cameras).

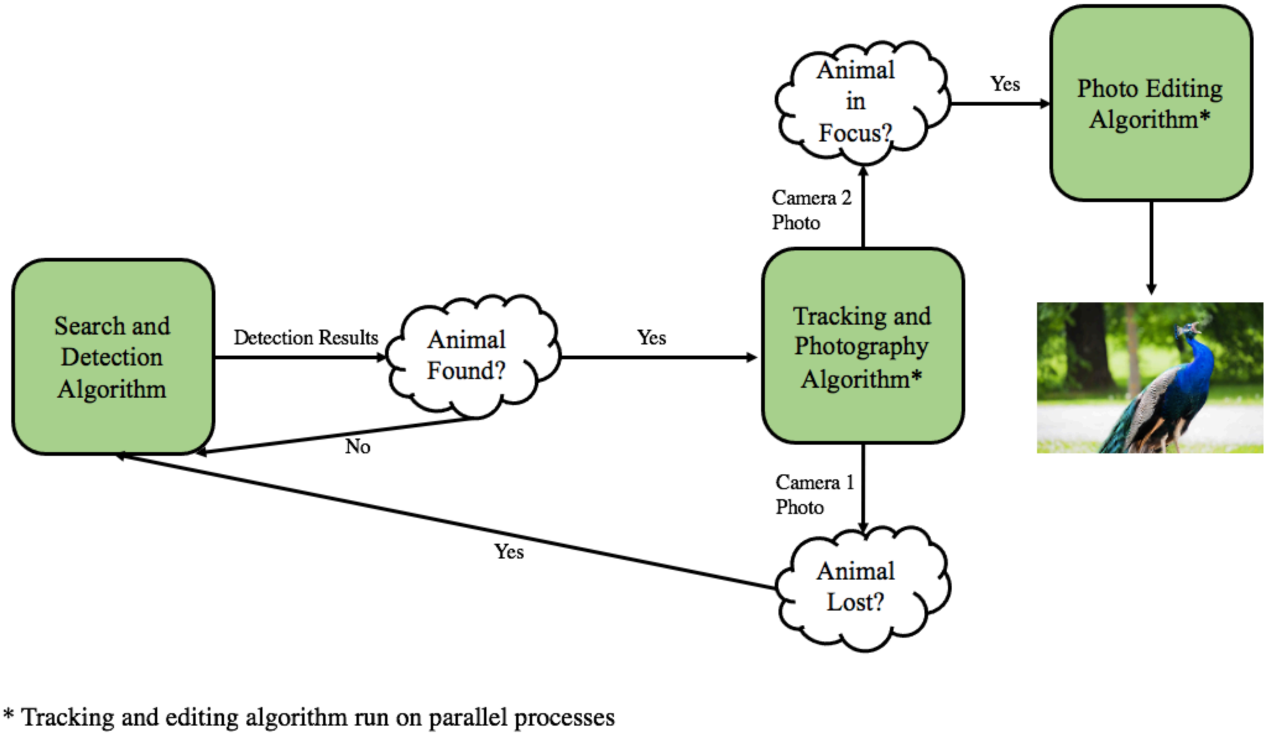

Search and Detection Algorithm
Detection Results
Animal Found?
Yes
No
Tracking and Photography Algorithm*
Camera 2 Photo
Animal in Focus?
Yes
Photo Editing Algorithm*
Camera 1 Photo
Animal Lost?
Yes
* Tracking and editing algorithm run on parallel processes

Fig. 2. The software layout for our nature photography robot

IV. Design Requirements

In order to pass the use-case requirements outlined in Section 2, our proposed hardware and software layout must pass as a set of more specific design requirements. For example, to detect animals in a 25m radius within 15 seconds with 75% recall, **we must certainly have an animal detection algorithm with at least 75% recall.** The equation for recall is shown in Equation 1. Even with a perfect searching algorithm, reaching the desired recall level would otherwise be impossible. Furthermore, to enable a complete search of the 25m radius, **the cameras must be able to pan 360 degrees and tilt 90 degrees up and down.** The time constraint outlined by the detection requirement also outlines a joint requirement for the detection algorithm and the embedded computer. The more images that can be processed in 15 seconds, the more complete the system's search can be. As a result, **once the 75% recall level is reached for the detection algorithm, the speed of the hardware/detection algorithm pairing should be maximized.**

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \quad (1)$$

In order to properly track and photograph animals moving at 2m/s, the system must pass another set of necessary benchmarks. At 2m/s, an animal 5m away could escape our camera's field of view from the center in under half a second. With this in mind it is essential our tracking algorithm can process multiple frames in this time to estimated and react to the animal's motion. We believe **at least 15 frames per second will be necessary for the tracking algorithm**, though testing will give a more accurate number. In order to avoid losing the animal while photographing we decided to use multiple cameras. However, this choice creates the issue of camera alignment. Because the cameras are stacked vertically, the alignment must only occur over the tilt. This alignment is time sensitive because, animals may walk out of the photography radius. With this in mind, **we will require the cameras to align in 1 second.**

Our goal for photo quality is to have the robot's photos be indistinguishable from photos taken by a novice using a smart phone. In order to meet this goal, our photo editing algorithm must have the ability to make photo adjustments seen in smart phones. **As a result, we will require the implementation of the most commonly used algorithms: temperature, tint, exposure, contrast, vibrancy, saturation, and sharpness** [5]. Additionally, **the camera will need to be 8MP or above** to meet the smartphone quality use-case requirement. It is possible to meet the 8MP requirement using a lower quality camera and a up sampling algorithm. However, completing this method effectively would be quite complex and require a large time commitment. As a result, we have decided to avoid this route and buy an adequate camera.

V. Design Trade Studies

Meeting the use-case and design requirements is a difficult task. There is no clear-cut way to accomplish our goals, and choices benefiting our progress towards one requirement may hinder our progress towards another. With this in mind, it is essential to evaluate tradeoffs between potential implementations for each of our sub systems. Where possible, we use pre-existing research for these evaluations, but some tradeoffs must be evaluated through our own testing. The hardware design requirements outlined are all filled by our purchases. For example, the camera has the rotation and quality requirements necessary for our project.

Our design requirements require the animal detection algorithm to perform with over 75% recall with as high of a speed as possible on our hardware system. When considering embedded systems, the two choices available in our class inventory were the Raspberry Pi's and NVIDIA Jetson Nano. We found that no popular CNN backbones ran faster than 3FPS on a Raspberry Pi, so we decided to use a Jetson for our project [4].

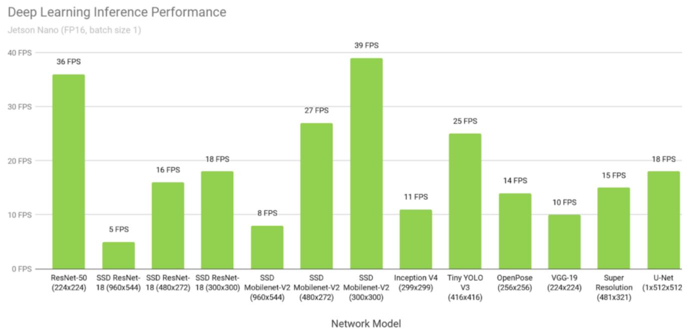


Fig. 3. A graph depicting runtimes of popular computer vision networks on the NVIDIA Jetson Nano. Not all results shown are for object detection. The fastest algorithm used for detection is the Mobilenet-V2 (300x300) [4]

Most modern papers on CNN's do not present recall as a metric, but their accuracies on challenging datasets far exceeds 75% [6]. Furthermore, our algorithm only needs to detect one class, animal. In contrast, research papers use datasets with many classes that are closely related, so our network should outperform the results presented in papers. It appears that most state-of-the-art approaches for object detection will perform well enough for our recall requirement, so our focus in selecting an algorithm is instead on speed. As an initial implementation, we plan to use the Mobilenet-V2 (300x300) because it was the fastest algorithm tested for object detection in NVIDIA's benchmarks [4]. There are multiple opensource implementations repurposing this CNN backbone for object detection, which we plan to use. For training we decided to use the WCS camera trap dataset, because this was the largest and most diverse dataset we could find for animals with bounding boxes. This dataset has 375,000 bounding box annotations for 675 species [7].

Photo editing is a far less studied problem than CNN's for object detection. Most methods existing use pixel-by-pixel manipulations using GAN's or other Deep Neural Networks [8]. These methods may work well, but they do not align with our goals. Our use case is to emulate human photography on a phone, which involves applying a certain number of algorithms like sharpening. Using a pixel-by-pixel approach can distort an image making the original impossible to reconstruct. It also provides no explainability or modifiability to the result. Automatic editing algorithms like we are looking for exist in photoshop and on iOS but are not opensource. With this in mind we will need to experiment with our own methods for this task.

VI. System Implementation

A. *Search and Detection*

B. *Tracking and Photographing*

C. *Photo Editing*

VII. Test, Verification and Validation

To test the robot's capabilities, a set of animal pictures are placed in the environment. The pictures included will be of different sizes and shapes (to account for orientation of the target animal) and will be placed at a variety of positions.
Once the robot has given the edited image output we plan to show human testers pairs of photos and ask them to try and distinguish the pictures to measure how well the system performs.

A. *Tests for Detection Requirements*

We specify that the robot must be able to detect animals that are more than 50% visible and are within 25 meters. To evaluate this, we measure the recall of the system to greater than 70% since animal appearances are sparse. We do so by placing sets of animal pictures under different levels of occlusion and lighting conditions.

We aim to be able to detect new animals and locate them within 15 seconds since we predict that walking animals do not stay in the same position for a long time.
We measure this by timing how long detection takes when a new subject is introduced to the environment. We place targets at varying distances (5m, 15m, 25m) to ensure that the detection algorithm works in the required timeframe at all possible distances less than 25m.

Similarly, we ensure that the robot can detect in different environmental conditions by varying lighting conditions when we place the animal pictures and repeating the above mentioned testing method.

Failure of the Detection algorithm from performing properly could be corrected by trying a slower/more accurate CNN model or a more representative dataset.

B. *Tests for Tracking Requirements*

Considering that the speed of a flying bird or running animal is too fast to follow even for human photographers, we aim for the robot to be able to follow a walking animal at a speed ~2m/s. The robot's animal tracking system is to be tested at 3 different speeds and even different distances as referenced in the Tests for Detection Requirements. To test the tracking, we move pictures of animals at a slow speed (0-1m/s), medium speed (1-3 m/s) and a fast speed (3+ m/s). We see how the tracking algorithm performs at these speeds and the different specified distances to check whether the system can successfully keep the target in frame.

After measuring these results, we can verify the reliability of the Tracking algorithm and choose to refine it or change it if the system is unable to track animal subjects in the specified range of speeds and distances.

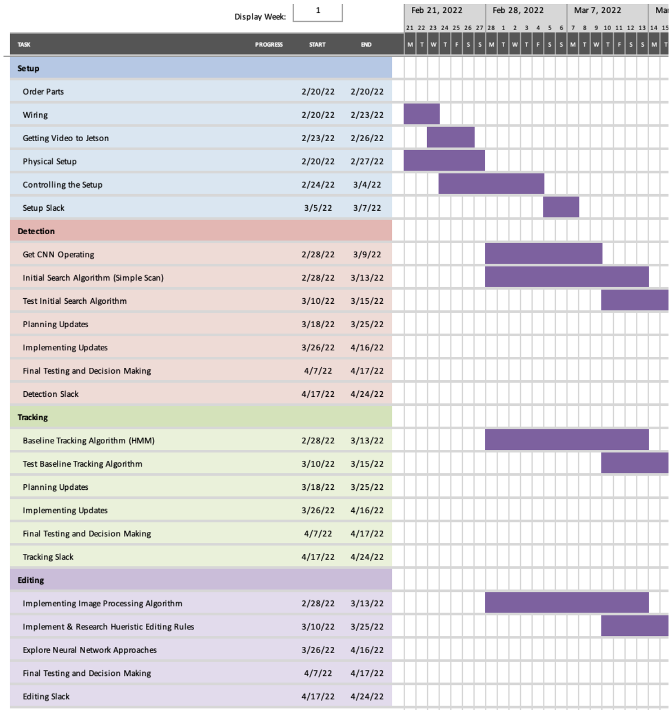
C. *Tests for Editing Requirements*

The editing section of the system uses a library of image processing algorithms which are inspired by image editing rules. We distort the images and then apply our image editing algorithms. We then show human testers pairs of images and ask them to distinguish between pairs of professional pictures and ones clicked by our robot to test how effective the image editing algorithms are.

VIII. Project Management

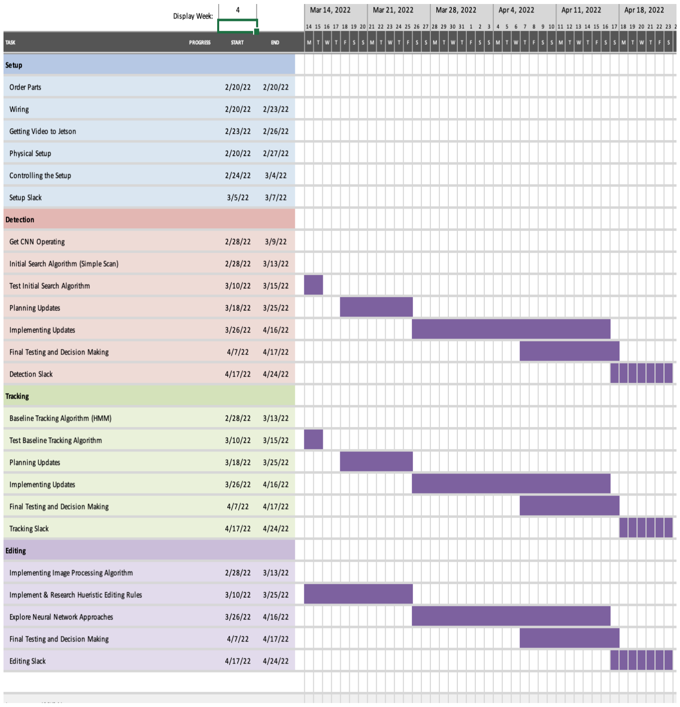
This section describes the project schedule, team member responsibilities, bill of materials and risk mitigation plans.

A. *Schedule*



TASK	PROGRESS	START	END
Setup			
Order Parts		2/20/22	2/20/22
Wiring		2/20/22	2/23/22
Getting Video to Jetson		2/23/22	2/26/22
Physical Setup		2/20/22	2/27/22
Controlling the Setup		2/24/22	3/4/22
Setup Slack		3/5/22	3/7/22
Detection			
Get CNN Operating		2/28/22	3/9/22
Initial Search Algorithm (Simple Scan)		2/28/22	3/13/22
Test Initial Search Algorithm		3/10/22	3/15/22
Planning Updates		3/18/22	3/25/22
Implementing Updates		3/26/22	4/16/22
Final Testing and Decision Making		4/7/22	4/17/22
Detection Slack		4/17/22	4/24/22
Tracking			
Baseline Tracking Algorithm (HMM)		2/28/22	3/13/22
Test Baseline Tracking Algorithm		3/10/22	3/15/22
Planning Updates		3/18/22	3/25/22
Implementing Updates		3/26/22	4/16/22
Final Testing and Decision Making		4/7/22	4/17/22
Tracking Slack		4/17/22	4/24/22
Editing			
Implementing Image Processing Algorithm		2/28/22	3/13/22
Implement & Research Hueristic Editing Rules		3/10/22	3/25/22
Explore Neural Network Approaches		3/26/22	4/16/22
Final Testing and Decision Making		4/7/22	4/17/22
Editing Slack		4/17/22	4/24/22

(a)



TASK	PROGRESS	START	END
Setup			
Order Parts		2/20/22	2/20/22
Wiring		2/20/22	2/23/22
Getting Video to Jetson		2/23/22	2/26/22
Physical Setup		2/20/22	2/27/22
Controlling the Setup		2/24/22	3/4/22
Setup Slack		3/5/22	3/7/22
Detection			
Get CNN Operating		2/28/22	3/9/22
Initial Search Algorithm (Simple Scan)		2/28/22	3/13/22
Test Initial Search Algorithm		3/10/22	3/15/22
Planning Updates		3/18/22	3/25/22
Implementing Updates		3/26/22	4/16/22
Final Testing and Decision Making		4/7/22	4/17/22
Detection Slack		4/17/22	4/24/22
Tracking			
Baseline Tracking Algorithm (HMM)		2/28/22	3/13/22
Test Baseline Tracking Algorithm		3/10/22	3/15/22
Planning Updates		3/18/22	3/25/22
Implementing Updates		3/26/22	4/16/22
Final Testing and Decision Making		4/7/22	4/17/22
Tracking Slack		4/17/22	4/24/22
Editing			
Implementing Image Processing Algorithm		2/28/22	3/13/22
Implement & Research Hueristic Editing Rules		3/10/22	3/25/22
Explore Neural Network Approaches		3/26/22	4/16/22
Final Testing and Decision Making		4/7/22	4/17/22
Editing Slack		4/17/22	4/24/22

(b)

Fig. 4. Our Gantt chart for (a) Feb 21 to March 20, and (b) March 20 onward

Our schedule divides the Setup, Detection, Tracking and Editing secretions of our system. Our plan highlights finishing the physical setup section by the end of spring break. Similarly, we plan to finish the preliminary algorithms for the detection, tracking and editing phases right after the physical setup.

This gives us time to test the initial algorithms and plan as well as implement updates following this phase. This section of our schedule is shown below and also includes a designated portion of slack which gives us integration time and time to refine anything that is needed.

B. *Team Member Responsibilities*

We divide the responsibilities by different sections of the project as outlined in the schedule. Justin is the Image editing lead and plays a role in the Physical Setup too. Sid is the Detection and search lead for the project and is responsible for the electronics setup. Fernando is in charge of the Pretrained CNN Setup and also plays the role of the Tracking and photography lead.

C. *Bill of Materials and Budget*

Included in appendix.

D. *Risk Mitigation Plans*

The critical risk factor we identified for the project is the ability to detect animals accurately, in the defined time frame.

If we are unable to detect the animal while it is in the related environment, we will not be able to capture a photograph of the animal since to the system, it does not exist. Therefore, detecting the animal with a bounding box around it is a pivotal aspect of the project.

To mitigate the risks associated with this, we plan to use a pre-trained CNN model which already has established validity. We consider multiple possible models and evaluate the performance of each on our use case before making a decision. By doing so, we identify the best possible detection algorithm for our nature photography robot. In case we face possible failure with this due to an inaccurate model, we plan to use a slower and more accurate CNN model. Similarly, too much error in detection can be resolved by using a combination of more representative datasets while training our model.

IX. Summary

The stand-alone nature photography robot uses computer vision to click pictures of animals and edit them. The system employs two cameras for faster results and division of labor during the capturing phase. The system also features the use of servo motors to allow us to scan every direction and hence cover more area. The system is capable of detecting, tracking and clicking pictures of animals found in the environment which significantly reduces the need for long hours of monotonous human labor to capture these sparse animal appearances. The system also features an editing software which uses a library of image processing rules to edit the clicked pictures before outputting them to the user. This means the photos we click will be edited which reduces the need for further human labor and capital since editing is an expensive and time consuming skill to learn.

References

[1] https://www.nationalgeographic.com/photography/article/paid-content-take-a-photo-save-a-species-the-power-of-wildlife-photography

[2] https://mashable.com/article/robots-wildlife-photography

[3] https://digital-photography-school.com/photographers-are-robots-coming-for-your-jobs/

[4] https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks

[5] https://www.rei.com/learn/expert-advice/photo-editing-basics.html

[6] https://towardsdatascience.com/top-10-cnn-architectures-every-machine-learning-engineer-should-know-68e2b0e07201

[7] https://lila.science/datasets/wcscameratraps#:~:text=This%20data%20set%20contains%20approximately,by%20the%20Wildlife%20Conservation%20Society.

[8] https://github.com/Divyanshupy/MobileNet-Object-Detection

[9] https://arxiv.org/pdf/1412.7725.pdf

Description	**Manufacturer**	**Use**	**Quantity**	**Price**
8MP Pan Tilt Zoom PTZ Camera	Arducam	Real time image processing	2	$94.99
2 DoF Pan Tilt Digital Servo Kit	Arducam	Rotation and movement for tracking/detection	2	$89.99
Multi Camera Adapter module V2.2	Arducam	Connect the cameras to the Jetson Nano	1	$49.99
Shipping				$40
Total				**$460**