# Multi-Sensor Passive Ranger

Pedro A. Alvarez Fernandez, Taylor W. Whitton, Ilina Sunkara, John R. Haynes III

### Dept. of Electrical Engineering and Computer Science and CREOL, University of Central Florida, Orlando, Florida, 32816-2450

### *Abstract* — This paper outlines the design and testing of a multi-sensor passive range finder utilizing infrared search and track (IRST) algorithms. A calibration is done of a stereo visible camera system and an infrared sensor to undistort the respective systems. It is then demonstrated that through several algorithms, the cameras can detect, track, and range objects traveling towards it.

### *Index Terms —* Infrared, stereo vision, ranging, IRST, CNN.

### Introduction

In most range finding systems, engineers take advantage of active techniques to determine the range of an object from the observer. Examples of such active range finding methods include radar, sonar, and laser-based approaches. While these methods are highly accurate, they do not appeal to every use case. Since these methods involve producing a signal, the object that is being ranged can detect that it is being ranged. For certain applications, such as military applications, this is not always ideal. For situations where such a property is not desired, we produced the Multi-Sensor Passive Ranger (MSPR).

While active range-finding systems send out signals, passive rangefinders do not. Rather than relying on these output signals to gauge the distance, passive systems rely strictly on inputs to the system. As a result, any given object being tracked by the system has no way of detecting that it is being tracked. Again, an effective use case of this technology is in military applications where the element of surprise is valued.

Our design is composed of set of subcomponents: the stereo system, the IR camera system, and the power system. The power system is responsible for providing the correct voltage and current to each of the components in our system, namely the Nvidia Jetson Nano board we chose as our microprocessor, and the fan that will keep the board and other components cool. This system is composed of a 120VRMS to 12V AC-to-DC converter, and a DC-to-DC converter converting 12V DC to 5V DC. This is necessary since the Jetson Nano runs on a 12V input, while the fan runs on a 5V voltage input.

The IR camera system is composed of the IR camera (FLIR TAU 2), and all the code required to derive the range of a given object from the IR camera. On each frame that the camera reads, several steps need to take place to extract the range estimate. First, the frame needs to be pre-processed to remove any lens distortion. Next, the location of the object will need to be extracted from the image to give a region of the image that can be processed. Once the object has been located, the region in which the object is present needs to be parsed using the apparent surface algorithm – where we search for pixels greater than a threshold intensity and obtain a pixel count. This count can be translated to an approximation of the distance of the object in the image.

The stereo system is composed of the two Arducam IMX477 visible light cameras and the code required to derive the range of a given object from the IMX477’s. On each frame captured by the set of cameras, the system will first preprocess the images to grayscale to reduce the computational complexity of the later stages of processing. Next, the system will use the two images to compute a disparity map (mapping of the distances between objects in a set of stereo images). The closer the object, the higher the disparity. Finally, the range will be extracted utilizing this disparity map.

Both the stereo camera system and the IR camera systems are connected to and powered by the Nvidia Jetson Nano microprocessor board. A key component of our design is that these two systems should interact with one another, to detect inconsistencies between range measurements. To accomplish this, after both systems have completed their range computations, the final values will be cross-referenced, and should a large difference exist between the two values (>5%), the readings will be deemed inconsistent at that point in time.

Our goals with the project are to produce a system that can accurately find the range of an object in a static scene. The range estimation accuracy should be ±10% of the true value. The time to differentiate the object from the background should be done within 2 seconds. Additionally, we should be able to perform this range computation at least 7 times per second. We should also be able to track any object in the frame at the same rate. We feel that working within these bounds will allow us to realistically create a system that is usable in the field while also being achievable within the scope of this class.

5)

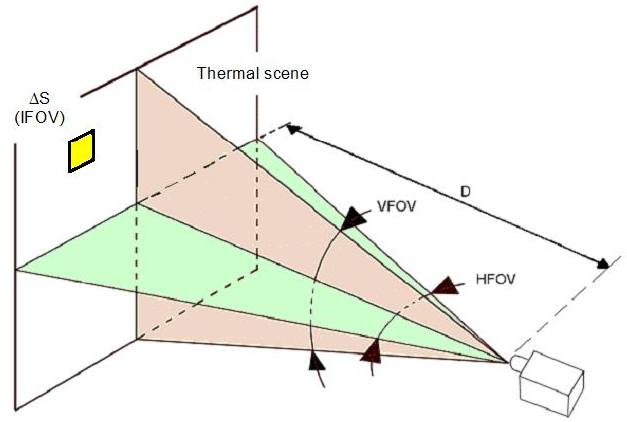
### II. Specifciations and Standards

The engineering specifications we had were aimed at the accuracy of the calculated range and the speed of the calculation. The estimation speed is going to be more dependent on the hardware, so we set a goal of 1 estimation per second as a baseline. We wanted to be as accurate as possible while also being able to get the estimated value close to the real value, so we set a goal of the estimated range being within 10% of the true range. These 2 goals go against each other because we could get a better estimate with more time, but we want to give as close to a live value as possible. Another goal was to differentiate the target within one second of the system activating and when a new object appears in the field of view. This is so then when something comes into view of the camera, it starts tracking it right away. We had some stretch goals we would have liked to achieve like a disparity image and velocity estimations but lacked the time to implement them into our design.

Some of the standards we must follow include IPC-2220 (IPC-2221), UL 1642 as well as the standards set for the FLIR Tau 640, the Arducam MINI and the Jetson Nano. IPC-2220 and IPC-2221 are standards regarding printed circuit board design, and UL 1642 regards lithium ion batteries, which was a stretch goal of ours to make the system more portable. The standards set for the cameras are about safety and possible electromagnetic emissions. The Jetson Nano is compliant with the Federal Communications Commission.

### III. Theory

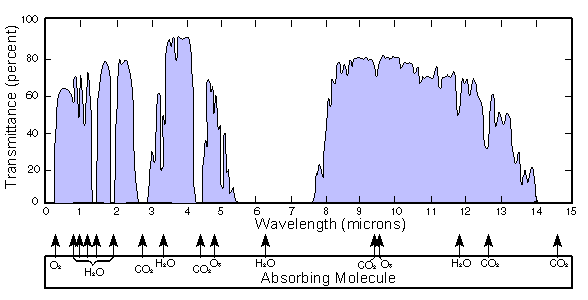
In this section we will provide some concepts that will be helpful in understanding the inner workings of our project. We will introduce infrared (thermal) imaging and sensors, stereo triangulation, lens distortions and calibration, and power design



*A. IR imaging and sensors*

Infrared cameras operate at different ranges of the electromagnetic spectrum than visible cameras do. The wavelength (color) range labeled as infrared is from 0.9 um – 14 um, and this is the typical range an infrared sensor can detect. Thermography is the process by which a detector gets radiation from a scene and converts it into an image of thermal distinction. The light from the field is incident onto a lens that focuses onto a detector, which is connected to electronics and software to analyze the information and display the images and signals [1].

The specific detector deployed in IR cameras is a focal plane array (FPA) of micrometer size pixels which can detect IR wavelengths. After undergoing specialized radiometric calibrations, software, and computational methods, the temperatures can be estimated at each pixel. Based on the application at hand, we need to select the right portion of the IR EM spectrum by considering the atmosphere between the camera and the object of interest. The transmission spectrum is illustrated in figure 1, and it can be concluded that the 7 um – 13 um range is best for diffusive atmospheric scenarios such as gas, fog, or smoke



**Fig. 1.** Transmission spectrum on the infrared range for different atmosphere effects. Permission pending.

Other factors considered for IR cameras, are field of view (FOV), resolution and frame rate. The FOV determines the spatial area of the scene that will be captured by the camera and is given in degrees. The further away from the camera, the larger the area that can be seen, this can be seen in figure 2. There is also an instantaneous field of view (IFOV) which is the detector pitch (size) divided by the effective focal length (EFL) of the camera. The IFOV is the smallest feature in the FOV that can be identified at some range.

**Fig. 2** Illustration of FOV and IFOV.

*B. Stereo Triangulation*

Stereoscopic cameras utilize stereo disparity to calculate depth in a similar fashion to the way that human vision is able to perceive depth using the horizontal separation between the eyes [2]. This process is termed stereopsis and stereo cameras are able to simulate this by employing geometric triangulation through taking two different images with separate cameras [3]. **Figure 3** shown below depicts the setup of the two cameras in relation to the objects being ranged along with the image plane shown using the focal lengths of the cameras. By measuring the disparity between specific pixels correlated to objects between the two images, the depth of the object itself can be calculated [3]. With the help of computer aided image processing, any two corresponding frames from these cameras can be jointly processed to create a disparity map. This would visually indicate objects that are closer and further from the camera system by representing them in different colors. The basic principle behind creating these maps is that the disparities should be larger for closer objects and smaller for further ones.

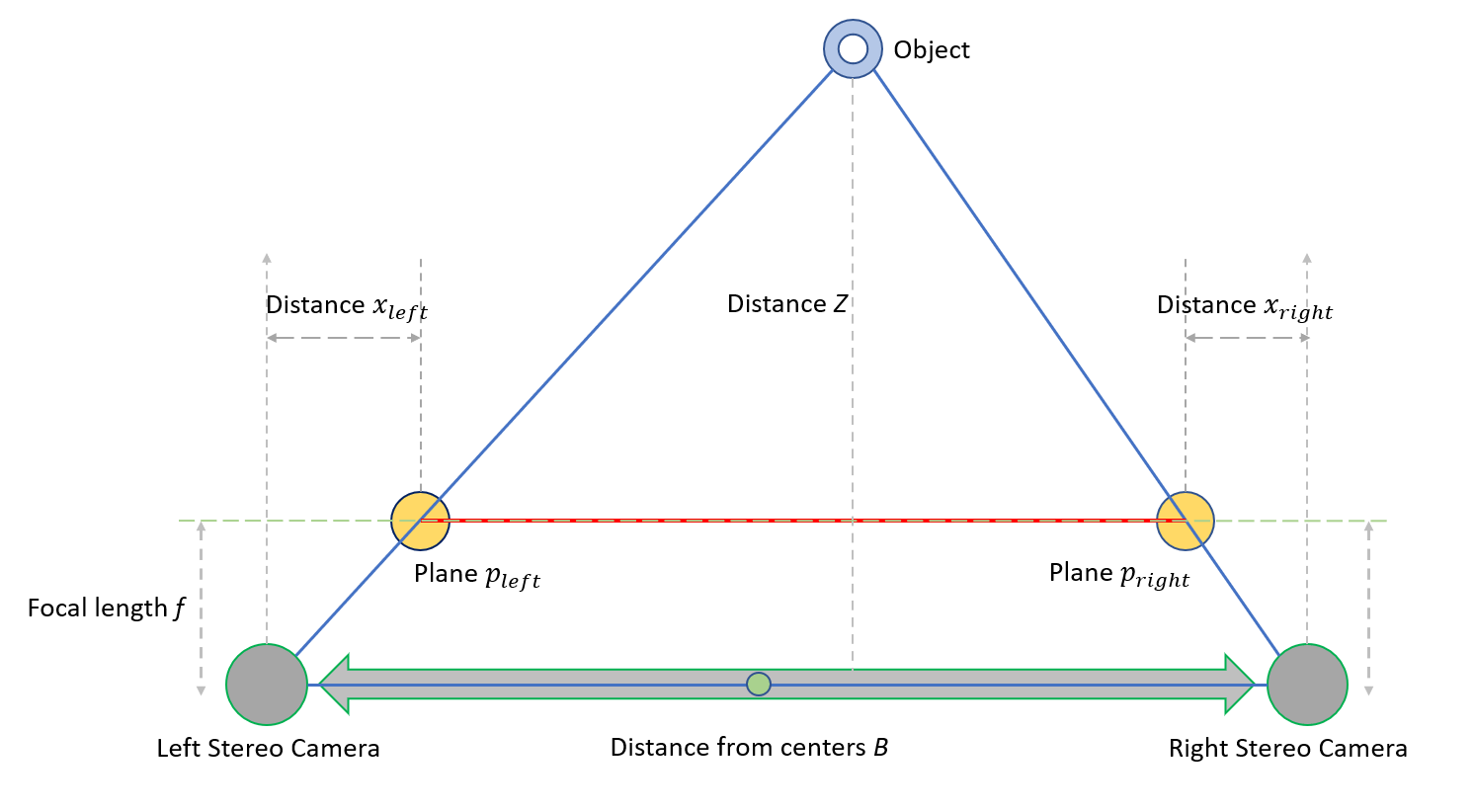


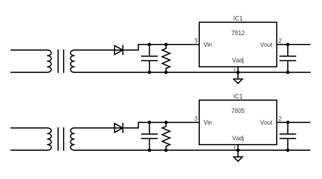
Fig. 3 Stereo Camera setup with labeled distances

Stereo depth measurements can also be taken using both passive and active techniques. Passive systems rely solely on light that is already present within a scene and typically work well in any illuminated conditions, whether it be through natural or artificial light sources. While passive stereo systems do not typically perform as well as their active counterparts in low light scenes or with non-textured objects, passive ranging is a more cost-effective method and works well for the purposes of our project. The active version of a stereo system would involve the use of a structured light source between the two cameras and would also be more useful for low lighting conditions. However, in sufficient lighting conditions, an active system would fundamentally work the same way as a passive system [8]. Also, since a primary objective in this project is to determine range without the risk of detection, introducing an active light source would not be favorable to our design.

*C. Power Design*

This first power design is 2 separate conversions. The entire design goes from 120V AC to 12V DC and from 12V DC to 5V DC. The input would go into a rectifier, which turns the voltage into DC at the capacitor. The input current goes through the transformer and goes to the MOSFET. The MOSFET gate is driven by the flyback to turn the DC current on and off, turning it into AC current. The Flyback knows when to do this because it senses the output voltage from the transformer. The output is 12V DC from the output diode at the output capacitor. This then goes into a voltage regulator, which steps the voltage down to 5V from 12V. See our complete documentation for a complete picture.

The second design shown in figure 4 uses voltage regulators instead. The input from the wall goes through a transformer, which transformed the 120V AC into around 12V AC. This then goes through the rectifier, turning it into a DC voltage at the capacitor. This output around 21 volts in testing. This is because the transformer I used was rated for 115V AC to 12V AC. One voltage regulator has an input of 21 volts and goes down to 12 volts, while the other regulator goes down to 5 volts. The 12V would power the fan while the 5V would power everything else.



**Fig. 4** Power Design 2

### IV. Design Components

*A. IR Camera: FLIR Tau 640*

The IR camera used for this project is the FLIR Tau 640 core which was loaned by the AFRL for the duration of this project. This compact and rugged camera core has the option for interchanging lenses, is designed for demanding integrations like handheld imagers and airborne devices. The resolution is 2.267 mrad with a pixel pitch of 17 um which is good for detecting objects at about 200-300 m. FOV is 90° x 69° with a 13 mm lens wich is desirable for detecting objects at wide angles. This has tradeoff with the resolution as the larger FOV will come with lower resolution. While this is a “slow” camera with a frame rate is 7.5 Hz, this is a good speed for image processing. To interface with the camera, we were also supplied with a VPC module which powers and controls the camera with a USB cable and outputs analog video using an MCX coaxial cable.

*B. Visual Cameras: Arducam MINI 12.3MP IMX477*

The visual cameras used in this system are both Arducam MINI 12.3MP IMX477 models. They each have a 3.9 millimeter focal length and 75 degree field of view along with a maximum resolution of 4056 x 3040 pixels. While the preferred specifications would have included a 90 degree field of view to match that of the infrared camera, it was difficult to find cameras that matched this description. Since the stereo system involves two cameras spaced a certain distance apart, it was decided that a 75 degree field of view for each camera would be suitable. The MIPI ports that these cameras utilize are useful in terms of compatibility with the chosen microprocessor. Once we narrowed down our selection to the NVIDIA Jetson Nano, it was also possible to filter our camera options by compatibility with the controller. The 2-lane MIPI connection was also preferred over the 4-lane option due to constraints on the Nano [5].

### V. DESIGN AND IMPLEMENTATION

*A. Spatial calibration*

In order to determine range using the stereo visual cameras, they must first be calibrated. To accomplish this, we used MATLAB’s stereo camera calibration toolbox to find specific calibration parameters of the camera system. To facilitate this process, we designed a python program that would quickly take pictures with all three cameras simultaneously at the push of a button and file them into separate folders labeled ‘left’, ‘right’, and ‘IR’ in relation to each camera. For the purposes of calibrating the stereo cameras, only the images related to those cameras were used. Once we were satisfied with images we had taken of our custom checkerboard, the process of loading the images into the MATLAB calibration toolbox was simple. The toolbox was able to find calibration parameters based on the detected locations and reprojections of the checkerboard corners in each image.

Once the calibration parameters were found, the images were able to be rectified correctly. This means that the left and right images would be slightly adjusted so that the pixels in each image in the horizontal direction would align correctly with each other. Calibrated and rectified images are able to produce a much clearer disparity map, as the pixels between images are already aligned to a certain degree.

The computation of a disparity map employs a process called block matching, which involves searching for corresponding pixels between specified objects in the two images. Rectified images would align pixels in the horizontal direction, making the block size that would need to be searched through smaller than that of an unrectified image. To correctly match the blocks between the two images, the block on the second image is shifted across the search block one pixel at a time. At each point, a similarity score is calculated between the shifted block on the second image and the original block on the first. After iterating through the entire search block, the block with the highest similarity score is found and the center pixel at that block is returned. The difference between the horizontal locations of the corresponding pixels in the right and left images represents the disparity of that object [48].

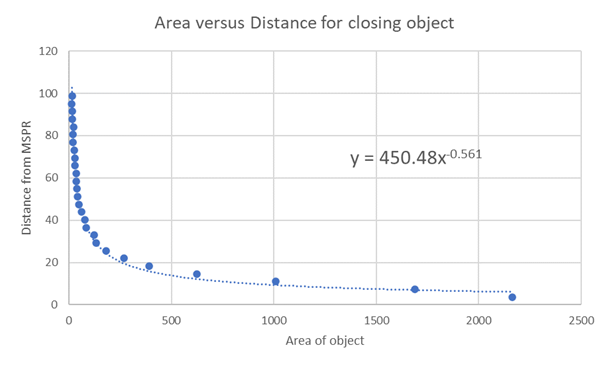
For our actual implementation we were able to create a disparity map by taking an image from each camera as an input. By using calibration parameters taken from running MATLAB’s stereo calibration toolbox on a set of checkerboard images, we were able to perform rectification along a common plane in order to more easily match the pixels. We then undistorted these parameter matrices and were able to remap the physical images onto them. These steps are taken primarily to ensure our system is calibrated correctly.

Once this is completed, the images are able to be combined into a disparity map using python’s OpenCV library. The disparity map is able to show disparities between pixels corresponding to different objects and possesses intensity values that either increase or decrease accordingly. Since these values are not yet able to represent any specific measurement, the disparity map can be converted into a depth map using the stereo triangulation equation. This is represented by (focal length \* baseline distance) / disparity, where the focal length is represented in pixels.

*B. IR ranging*

For ranging purposes, we look at two cases of object detection. The first case is when the object is really far away and is considered to be a point source. This case is called the point source case, and uses the Inverse Square law, which says which says that irradiance (H) decreases as (1/R^2), with R being the distance away from the point source. When only a handful of pixels are seen of the object, it is considered a point source and only the intensity is needed for range finding

The second case is called the extended source case. This case is useful for moving objects that take up a multitude of pixels on the camera. Some initial information is required for both cases, such as object location in the image plane. In this case we also need to know the changes in range. The equation in this case is 𝑅=−2𝐴 1/(𝑑𝐴/𝑑𝑅). Which is -2 times the area (in pixels) divided by change in area due to change in range. A modified version of the apparent algorithm was used by adding the we know what the object size is already. If this is known, a function fit can be done by taking pixels areas after moving specific range amounts. In this case the object was moving towards the camera and the area versus range was plotted in Figure 5. A power power function was the best fit for this scenario and is the function used to calculated range of extended targets.



**Fig. 5.** Area vs distance for Toyota Tacoma test object

*C. Software Project Structure*

The software component of this project is critical in producing a functional piece of equipment. Consequently, it was vital that we build a system that is extensible, maintainable, and efficient. To accomplish this task, we took heavy advantage of object-oriented design principles to model each of the components in the system. We created modules for each component of the system, and within these modules, we built classes and methods that operate within that system.

In our system, we created classes that wrap around the various libraries that we used to extract frames from the three cameras in our system. The IRCamera and StereoCamera acted as wrappers around OpenCV, allowing us to quickly and easily access the methods that we required to operate on the frames that the cameras captured. In the case of the IR camera, the IR camera captures each frame and saves the frame on the object on each iteration. Once the frame has been saved, we created methods to operate on that saved frame, saving us from the trouble of having to keep a separate variable in the main function to store frame data.

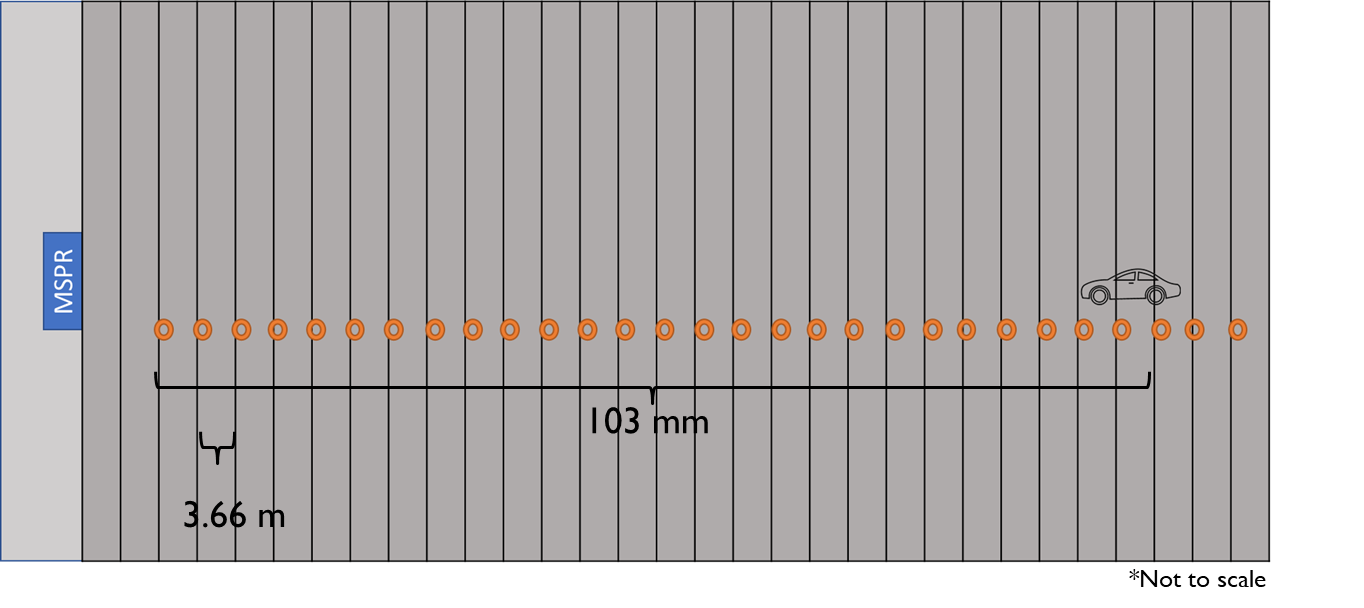
In the case of the stereo cameras, it made sense for us to abstract even further. The StereoCamera class served as a wrapper around the basic OpenCV functions. However, since we had two stereo cameras, we created another class called StereoSystem that unified two StereoCamera classes into one cohesive object. From this StereoSystem class, we could capture frames from both StereoCamera objects simultaneously, and store their data on the StereoSystem object. Additionally, this gave us a central location for us to call our stereo vision algorithms on both frames through the forms of methods implemented on this class.

Once we obtained the frames, using the methods found in both these classes, we needed to be able to perform operations on the frames. We decided that the best course of action was for us to separate the algorithms into two separate modules, one for the IR camera and the other for stereo vision. This separation of concerns allowed us to work collaboratively without intruding on the work of the others within our team. The individuals responsible for working on the IR camera were assured by this design that any changes that they made were localized to the IR camera, and likewise with the stereo system. This design minimized Git merge conflicts and allowed us to rapidly develop and test our own systems.

Within the IR camera module, we created several submodules that each served a purpose on the overall calculation of the range from a given frame. First, the trt.py module was responsible for applying Nvidia’s TensorRT GPU-driven inference library with Google’s “ssd-mobilenet-v1” open-source convolutional neural network trained on the COCO dataset to obtain a bounding box around a given object in a scene. We filter out any objects that we are not interested by the class ID’s generated by “ssd-mobilenet-v1”, namely anything that isn’t an automobile or a human. Once we have our set of objects, we iterate through each detected object and pass the box coordinates to the apparent\_surface.py module.

The apparent\_surface.py module is responsible for applying the apparent surface algorithm within the bounds of the box that TensorRT placed around a given object in the scene. The algorithm iterates through each pixel in the box and compares the intensity of the pixel with a threshold value that we set to 200 out of 255 for the 8-bit intensity system. Since our image is a BGR image, we first average each layer’s intensity to generate a single intensity score. If this intensity is greater than our threshold, the algorithm increments a counter variable that is fed into the equation we derived to relate pixel count to object distance. The output of this equation is drawn to the screen as the z-axis in our real-time GUI.

Simultaneously, the stereo system processes each frame that it receives. To accomplish this task, we also make use of the TensorRT module, TRT. Since we have two frames rather than one, we need to detect the object in both frames to triangulate an averaged position in both frames. Consequently, we run Google’s SSD over both the left and right captured frames on each iteration of the program’s runtime loop. Should we detect the object in both frames, we extract the bounding box of the object that has the highest confidence score in both the left frame and right frame. We then take these bounding boxes and average their position to get a centralized location of the object. This centralized location is then passed into the stereo\_ranging.py submodule, which applies a disparity mapping algorithm to compute the range when given the averaged bounding box.



Once we have the range values, we then must draw this range output to the GUI as well as the bounding box around the object. In the case of the IR camera, we can draw the range of multiple objects, however due to the limitations on our hardware, we limited ourselves to only ranging one object in the stereo system. To accomplish this task, we take the x and y-coordinates from the TensorRT object detection result, and the z-coordinate from the range finding algorithms. Using the x and y-coordinates as well as the width and height derived from the TensorRT algorithm, we use OpenCV’s rectangle method to draw a box on the frame captured in each class. We use OpenCV’s putText method to write the x, y, and z positions to the GUI and use the x and y positions as the anchor position in the frame for the text.

After completing the execution of the object-tracking and range-finding algorithms, we finally display the images to the user. To accomplish this, we use OpenCV’s imshow methods to draw the frames to the screen. We produce separate windows for the IR camera and the stereo cameras to ensure that it is easy for the user to differentiate between the range readings that the program computes. The main mspr.py module is responsible for the runtime loop of the program in addition to calling all the methods that need to occur on each iteration of the runtime loop both subsystems.

### VI. Demonstration (results)

*A. Testing procedures*

For testing the system, we chose to take measurements during desirable weather conditions. The MSPR was used on the last floor of a parking garage at nighttime after the concrete had cooled off. A car was on for a while so that the motor is the hottest thing in the scene. The car drove towards the camera from afar at intervals of 3.66 m and measurements where recorded. The schematic is shown in figure 6.

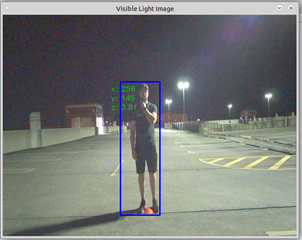
**Fig. 6.** Physical test set-up

*B. Power Design*

For the power system, design 1 and 2 failed for different reasons. Design 1 failed because the capacitance seen by the input was too high, causing the circuit breaker to trip. This can be solved by adding a 1 to 1 transformer at the input to make the capacitance seen by the input to be lower, reducing the current flow once the system gets plugged in. The second design failed because there was never enough current to start up the Nano and keep it on consistently without the voltage regulator overheating, causing the regulator to do a thermal shutdown.

*C. Stereo System*

With a working depth map, we were able to find range using various methods. In our final implementation, we were able to use the object detection algorithm on both image feeds and found the range on an area directly in between where the images were found on each image in the depth map. To find the range, we took the average of all pixels within the bounding rectangle and ignored any values that went outside of a certain threshold. The x and y values on the **Figure 7** represent the coordinates of the object within the image while the z value represents the range.



**Fig. 7** Stereo Camera System Working

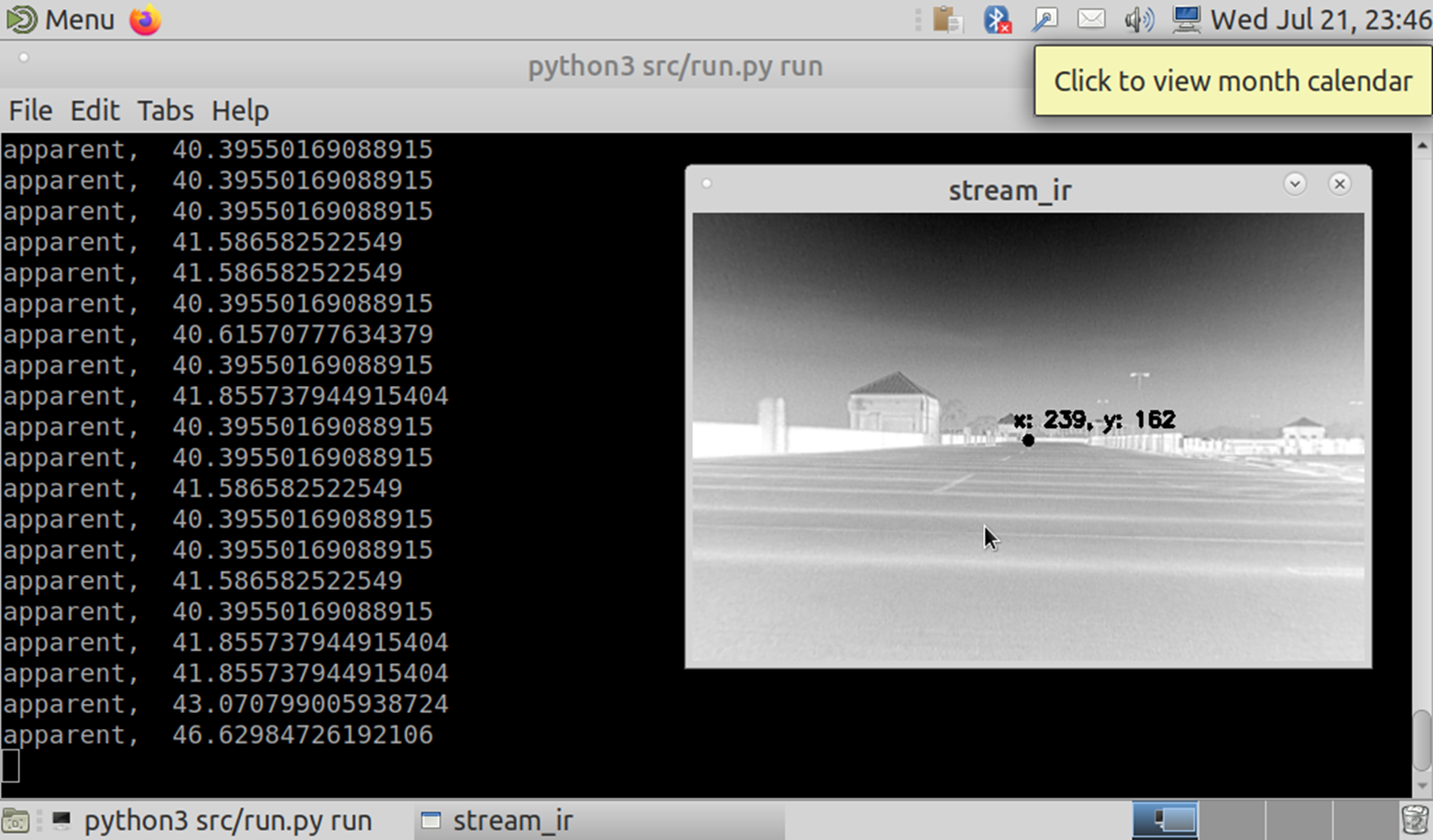
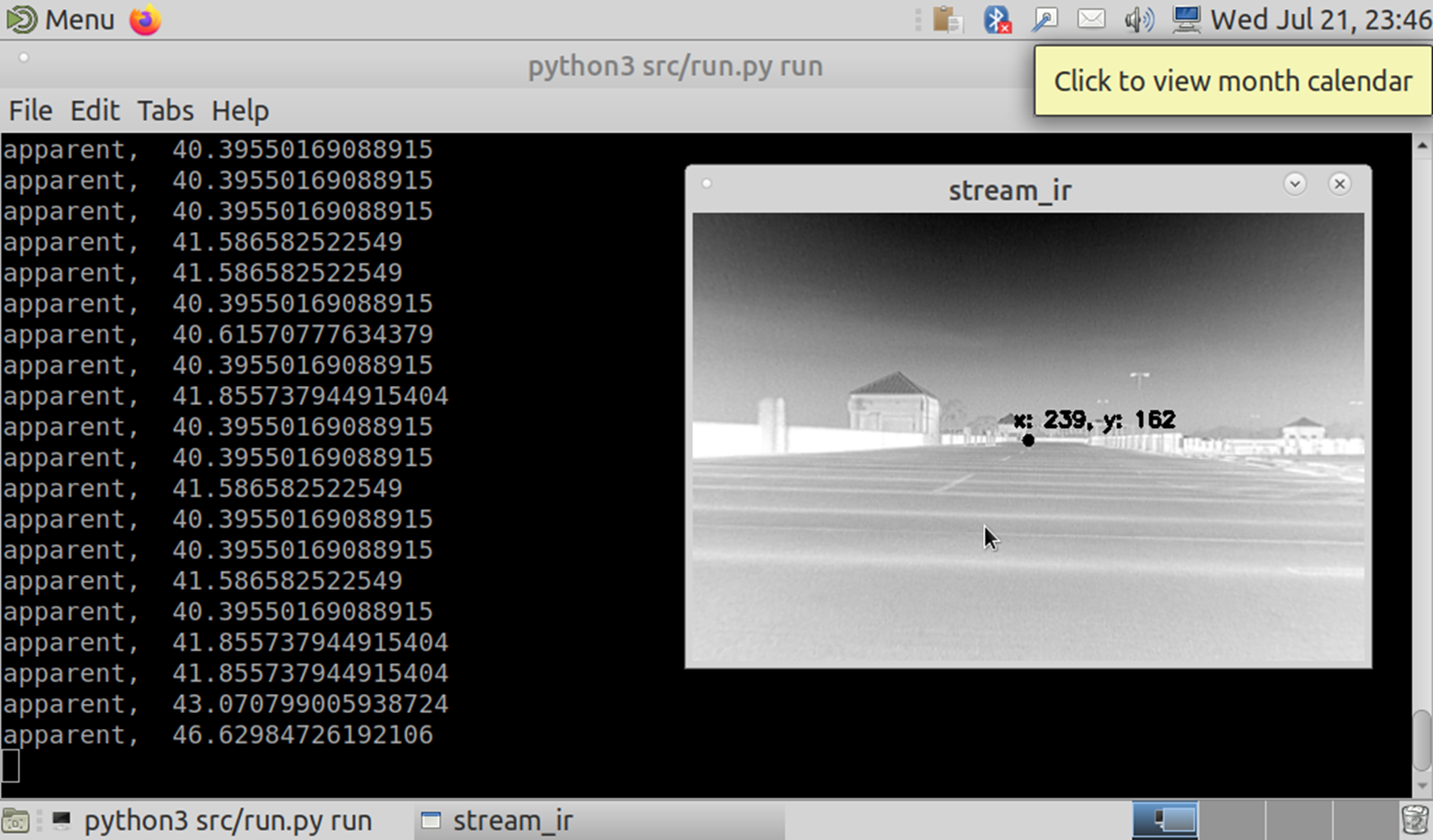
*D. IR Ranging*

The set of images in figure 8 show the results of the spatial calibrations for IR camera. The image on the left shows a potent barrel distortion, while the right is effectively undistorted. This process was applied for every frame of image acquisition.

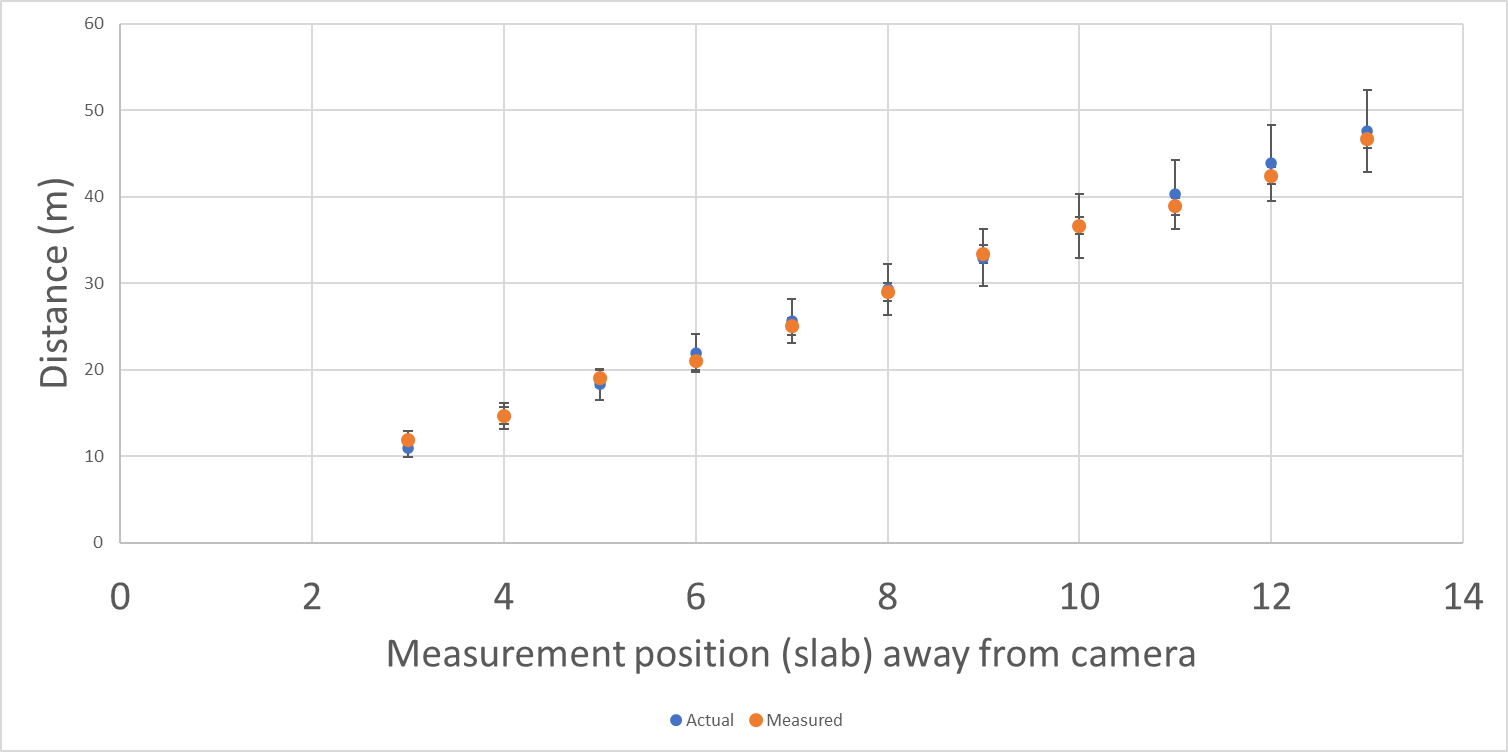
|  |  |
| --- | --- |
| Left: Distorted | Right: Undistorted |

**Fig. 8.** Distorted and undistorted image of FLIR Tau 640 f = 13 mm.

Below in figure 9 there is a sample image of the car that is really far away and is detected, giving the coordinates in the image plane as well as the range based on the apparent surface algorithm. A set of measurements was collected at different ranges and is displayed in figure 10. As seen in figured 10, the measured ranges are within 10% of true values for the highest range able to be measured being ~ 48 m from the cameras. This is because of the object detection and identification difficulties. As long as the object can be detected and tracked within the scene, a fairly accurate determination of range can be made. While the range estimation is fairly quick, this is only because of the simplicity of this system being on microcomputer. We were not able to work with the most high-resolution images. With stronger hardware, the results from this work will improve exceptionally.



**Fig. 9** Toyota Tacoma test object ranged at 46 m.



**Fig. 10.** Actual vs Measured distances for Toyota Tacoma test object

### VII. Conclusion

This system was created to determine range passively with minimal risk of detection. While active systems are typically a more precise alternative in measuring physical distance, the objective of creating a covert device led us to research other means and to design a fully passive system. This included the use of one long wave infrared camera along with two stereoscopic visual cameras. The IR camera is able to determine the range of a moving object in both ideal and non-ideal lighting conditions while the stereo visual camera system is able to determine the range of static and moving objects with sufficient lighting.

The cameras are aligned and connected to the microprocessor in a specially designed housing made to fit and secure all the necessary components for the system. Certain outputs and inputs of components also had to be accounted for in case there was no direct connection from that unit into the Jetson Nano. While the visible range cameras contain compatible MIPI port connections, the IR camera is connected through USB 3.0 after being converted from RCA. The system is also powered by a specific power system that is designed to supply power to all the associated components.

A large part of the design in this system is also in the programming related to the processing of the image feeds from the cameras. The IR camera software design includes a motion tracking algorithm along with programs to calculate range based on specific emissivity related equations. The stereo visual camera code on the other hand creates disparity maps from individual frames and calculates depth based on disparity between pixels in a static scene. An overarching setup for the code was also designed and structured beforehand along with interfacing code to connect components and display results in an organized manner.

This project was developed extensively through interdisciplinary collaboration and teamwork. Many aspects of the design relied on a combination of knowledge in the fields of optics, computer science, and electrical engineering. Each member of our group had more specific contributions to different parts of the overarching system which came together to form our final project. Overall, we hope that our work can prove to be both accessible and valuable to the field of passive range finding and that it could have the potential to be incorporated into a larger system with the purpose of discrete tracking.



Overall, our system was able to meet our specification of being able range objects within 10% of their actual distance value. The highest distance we were able to measure with the infrared camera was up to 48 meters and we were able to implement two different methods for object recognition in this system. The stereo system was more reliable within the 10 meter range and was able to show range values on stationary objects. This system also worked better in well-lit lighting conditions. Both systems would work more accurately with higher resolution images, however, due to constraints related to processing power on our microprocessor, it would have been too computationally expensive to implement this.

### Acknowledgement

Special thanks to Steve Marlow and Darrel Card from the Airforce Research Labs for all the advice and loaning the IR camera to us. Thanks to KnowBe4 for nurturing our computer engineer’s programming skills and being supportive during this process. Lastly, thank you to CREOL for all the equipment that was borrowed from them.

### Biography

Pedro Alvarez Fernandez is pursuing a bachelor’s degree in Photonic Science and Engineering, planning to graduate in August 2021. Pedro will begin his career to contribute in the field of Optics and Photonics after graduation



Taylor Whitton is pursuing a bachelor’s degree in Computer Engineering. He expects to graduate in August 2021. He is currently employed full-time as a junior software engineer at KnowBe4 following an internship opportunity over summer 2020.

Ilina Sunkara is a Photonic Science and Engineering student at the University of Central Florida. She will be pursuing a Master’s Degree in Imaging and Light in Extended Reality following the completion of her Bachelor’s Degree.



John Haynes is a senior in Electrical Engineering at the University of Central Florida and will be receiving his Bachelors of Science in Electrical Engineering in August 2021. John will pursue a career in digital circuit design after graduation.

### References

[1] FLIR. *The Ultimate Infrared Handbook for R&D Professionals*. Available from: <https://vault.flir.com/file/asset/18320/original?token=8df9f33c-ebf4-43ce-897e-4fdeb4672a18>

[2] Eric Carmi, J.V., *Geometrical Calculations with Stereo Vision*. 2015, California State University: Sacramento.

[3] Anantharam, P. *Disparity Map Computation in Python and C++*. 2020; Available from: <https://pramod-atre.medium.com/disparity-map-computation-in-python-and-c-c8113c63d701>.

[4] *Raspberry Pi 4 Model B specifications.* Raspberry Pi

[5] Arducam MINI High Quality Camera with M12 mount lens, 12.3MP 1/2.3 Inch IMX477 HQ Camera Module for Jetson Nano