UAV Detect and Avoid

Lindsey Raven
University of California, Davis
1 Shields Ave.
Davis, CA 95616
laraven@ucdavis.edu

Christopher Bird
University of California, Davis
1 Shields Ave.
Davis, CA 95616
ckbird@ucdavis.edu

Angela Tobin
University of California, Davis
1 Shields Ave.
Davis, CA 95616
ahtobin@ucdavis.edu

Shalmali Joshi
University of California, Davis
1 Shields Ave.
Davis, CA 95616
shajoshi@ucdavis.edu

ABSTRACT

Our project description changed and grew over time as all good projects do, and in the end we were able to settle on and complete our objective. Our unmanned aerial vehicle (UAV) had a camera, LIDAR sensor, and FPGA on it which read in a video feed, determined the location of an object and flew towards it by signaling the flight controller. Once it was within five feet of the object, it would stop and activate an avoidance sequence. Our FPGA and its components correctly analyzed the video feed and gave the correct signals to the flight controller as we were able to demonstrate in several videos. Unfortunately the UAV itself was not up to the task and would drift off course due to winds, weight balance, and small GPS issues. The image processing and coordinate locating was done mostly in the software components of the project. Those coordinates were sent to the FPGA to determine the next update to the flight controller through the use of the GPIO pins. Our hardware speedup was minimal, but if we had had more time we would have done some of the image processing on the FPGA itself.

1. ALGORITHM

The overall aim of this project was to integrate an image processing based collision detection system with a quadcopter's native flight controlling system. This was accomplished by analyzing individual frames of a video camera feed to identify a target and then signal the quadcopter to move so the target is centered in the frame. The quadcopter is then signaled to continue on its forward path until a LIDAR sensor detects that there is an obstacle within close range of the quadcopter and triggers an avoidance sequence.

We decided to utilize image processing algorithms to detect an object in the path of a the quadcopter. This is accomplished by using a camera connected to an integrated ARM processor and FPGA board called the Cubic board. This board was designed by Bo Zhou from Altera as an alternative to the De1-SoC boards used by the class. It allows us to be able to run a LINUX operating system that gives us a large amount of flexibility in implementing our algorithm while also utilizing the speed and efficiency of an FPGA. We developed an algorithm that captures frames from an onboard video camera and processes the frame pixel by pixel to determine its contents. The algorithm searches for a specific symbol of a pre-determined shape and color and identifies it within the frame. It then determines if the quadcopter needs to change its course based on the location of the symbol within the frame. Our system is entirely autonomous and protects the quadcopter without the need for human intervention.

2. IMPLEMENTATION

2.1 Image Processing

Specifically, our system uses the OpenCV image processing library for C++ to efficiently cycle through each image and process the pixel data in the frame. We run two modifications to the image to determine the location of the symbol. We chose to use a large orange-colored plus sign on either a white or black background as the symbol for our algorithm to detect. We changed the background color depending on the background of the environment we were testing in. The first image modification converts the color image to a grayscale image, and then determines if each pixel value is above a specific threshold value. If so, the pixel is turned white; otherwise it becomes black. The result is an image that isolates the white background of the symbol we are looking for. The second modification takes the original full color image and compares the red-green-blue (RGB) values of each pixel to a set threshold to determine if they are the color orange or not. The result is a black and white image where orange colored objects are white and everything else is black. The two black and white images are then combined to result in an image that only contains the orange plus sign symbol.

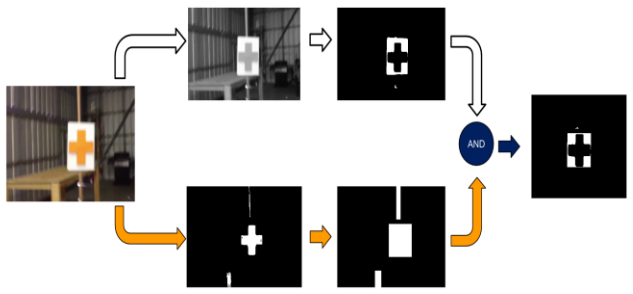

Figure 1. Process of filtering image

2.2 Coordinate locating and signaling

The location of the center of the plus sign within the frame is then calculated and analyzed to determine if a signal needs to be sent to the UAV to change its course. The system maintains the symbol in the center of the frame until it determines the object is close enough to need to avoid it. It communicates with the flight controller of the quadcopter by sending high and low signals on three general purpose input/output (GPIO) pins that are wired between the two boards. Using a combination of the three pins allows us to have 8 signals that the UAV flight controller can interpret as a command to go left, right, forward, back, up, down and to hover.

2.3 LIDAR

In addition to using the data from the camera, our system also receives data from the quadcopters flight controller that indicates distance to the object using a LIDAR sensor. We worked with two graduate students who were using the HUGO quadcopter with a LIDAR sensor to interface with our project. We worked with them to develop signals that both systems would use to interpret the data from the LIDAR. We received different signals depending on the distance to an object from the quadcopter. The flight controller uses two GPIO pins in combination to indicate the distance the LIDAR sensor detects an object in front of the quadcopter and communicate that to the system on the Cubic board. This is a feature added to our system to increase its accuracy in determining when to avoid an object.

2.4 Hardware/Software Interface

The hardware/software interface was done using Qsys modules through Quartus. We implemented modules that would activate GPIO pins on the cubic board to receive LIDAR data on two pins and to output signals to the flight controller on 3 pins. The standard modules that were generated were then modified to work with the Cubic board as it had pin assignments that were not standard in Quartus. The main issues with this was getting the correct pins activated to use for the LIDAR input and the flight controller output. These pins had to be designated specifically as either input or output pins while generating the component. The other part of the hardware/software interface was the camera we used. We decided to use an IP Camera that would be able to connect to our board using Ethernet instead of a USB port. This was done because Ethernet is a faster mode of transportation than USB and so that we did not have to deal with porting USB drivers onto our board. The camera connects to our board via a standard Ethernet cable and a crossover adapter that allows us to read the data from the camera without connecting to an Ethernet switch.

2.5 Memory Management

Memory management was mainly done through the Linux hosted operating system. By using offsetting calculations to determine the register values and address values of key parts in memory, we can begin to store and read images read in through the camera. Using OpenCV mats in conjunction with these memory addresses allows us to easily and quickly access different versions of the image in order to process the two separate thresholds. This means that we can manipulate the original image to process with one method and run another image processing method on the same image without hassle. If we had finished our hardware speed up of the image processing, we would be able to run both of these functions at the same time, and cut our processing time in half.

Table 1. Output signals

Output Pins			Function
0	0	0	Hover
0	0	1	Forward
0	1	0	Up
0	1	1	Down
1	0	0	Left
1	0	1	Right
1	1	0	<not used>
1	1	1	Avoid collision

3. RESULTS

Our system worked very well under our test conditions. It was very accurate in finding the target and determining its coordinates within the image frame. It was also very accurate at sending the right signal to the UAV’s flight controller in order to move the UAV. This was achieved through design decisions we made that balanced accuracy and processing speed.

In order to smooth the movements of the UAV and ensure only one signal was being sent at a time, we intentionally lowered how often we calculated different frames and their coordinates. However, once we captured a frame, we had to analyze it as quickly as possible so the sway and movement of the UAV while hovering didn’t throw it off. This ensured that we would be as accurate as possible in determining the location of the target, however, we would not be processing the video at a fast speed due to a lower sampling rate. If we had more time we would have cut the extra time off our processing speed, but nevertheless, it took less than half a second to analyze each new frame and send the signal to the flight controller. By optimizing our code had we been given more time, we would have been able to cut the image processing time in half.

Aside from the speed at which our code ran, the rest of the code worked to perfection. With an accuracy of approximately 90%, our code accurately determined the correct cross location and sent the correct signals to the flight controller. This accuracy proves both our implementation was correct, but also that the only roadblock remaining was the speed up.

4. LESSONS LEARNED

We have learned many lessons throughout working on this project as previously stated, but the final lessons for us to take from this is that getting very high accuracy with a powerful image processing library isn't as difficult as getting that image processing to be faster and more efficient in terms of space and complexity.

The conceptual technical lessons we learned from this project number much higher than the specific data lessons learned. Mainly, we should have spent most of the beginning of the project detailing the specifications of what needed to be done, and what the exact project specifications are. Since we did not nail down exactly what our project was until we were underway, getting our plan approved by multiple different advisors, mentors, and professors took much more time than it needed to. In order to fix this for next time, we would come up with a solid design plan, send it to the required leaders and professors early on. Based on their feedback, we would revise our plan until there was a solid plan that everyone agreed to before starting to work. If we had done this, we would have had more time for hardware speedup and testing.

In terms of advice for future students doing a similar project, we would recommend setting strict deadlines for different parts, and parallelizing your work more efficiently. Instead of waiting for one member to be done with a part before starting the next, figure out which pieces can be done at the same time and do that. If we had been more efficient about this we could have finished much faster. Finally, be sure to understand the full scope of the project, and be aware that coordinating logistics for a special project is very difficult especially if you plan on pleasing multiple parties. We attempted to coordinate with four group members, two professors, two teaching assistants, two Altera mentors and two graduate students, along with university affiliates for access to testing locations. We learned a lot about the value of getting things clarified early on and in writing. The TAs and professors were very helpful in guiding our group when we were struggling with defining the project and working out the implementation details.

To 'improve' this project, we would recommend having clear outlines set for independent projects that must be followed to aid the students in deciding how to attack their project specifications. If all of the requirements had been in writing, the group would have been able to make sure that their project meets those requirements before even attempting it.

5. CONTRIBUTION BREAKDOWN

OpenCV Image Processing Code- Lindsey (60), Chris (30), Angela (10)

Hardware/Software Interface – Angela (60), Chris (30), Shalmali (10)

Compiling Code - Lindsey (90), Chris (10)

Powerpoint presentations, reports- Shalmali (55), Chris (20), Angela (20), Lindsey (5)

Research – Chris (25), Lindsey (25), Angela (25), Shalmali (25)

Presenting - Chris (25), Lindsey (25), Angela (25), Shalmali (25)

Coordination with outside parties - Shalmali (65), Lindsey (25), Angela (5), Chris (5)

In our group, Chris was the strongest in FPGA resources and hardware. Angela was strongest in logic design, flow of control, and moral support. Lindsey was strongest in image processing code and solving compiler issues, and had incredible dedication and perseverance. Shalmali was strongest in coordinating the group's meetings with all the guides, creating and editing presentations and other documents, and keeping each of the parts of the project together.

6. REAL WORLD APPLICATION

In recent years the United States has been increasingly reliant on unmanned aerial vehicles to complete tasks that are too dangerous or impossible for humans to do. While the technology to fly and control UAVs is fairly well established, keeping the UAVs safe in flight is still an issue that needs to be solved. An advantage our system has is that it is autonomous and runs without human intervention once initialized. Therefore any sudden obstacles that appear near a UAV can be dealt with without relying on the reflexes of a human operator. Our implementation runs while the quadcopter is in an autonomous mode itself where it flies along a preset path and deviates as determined by the image processing algorithm.

While our system is specific to the conditions we have set for its implementation, it serves as a proof of concept that a larger scale version of our system can work. Utilizing computer vision - based algorithms, a system that uses visual feedback can be implemented on larger-scale unmanned aerial vehicles. For UAVs with existing camera or visual feedback sensors, implementing our algorithms would eliminate the need to add additional sensors that detect the UAV's environment. For systems that do not already have a camera or visual feedback sensor, implementing our system would be a precise way to identify obstacles in the path of the UAV to a more degree that would not be possible with other sensors.

7. ACKNOWLEDGMENTS

We would like to thank our mentors who have guided us throughout this project. We would like to thank Professor Soheil Ghiasi, Professor Leo Liu, Andrew Chang, Mohammed Motamedi, Minjie Zhu, Samuel Cheung, Bo Zhou, Terry Barrette, Professor Stephen Robinson, and Intramural Sports and Facilities.