Quadcopter Delivery Verification

Project Proposal

ECE 445 SP2016

TA: Luke Wendt

Team Members: Sachin Weerasooriya - Electrical Engineering

Raymond Hoagland - Computer Engineering

Rahul Joshi - Computer Engineering

February 8th, 2016

Table of Contents

1.0 Introduction

1.1 Statement of Purpose:

When you order a package that is shipped via ground, it is either left at your door or you must be present to sign off on the package. Services like Amazon PrimeAir look to speed up delivery time with drone shipping, which claims to be able to come to your door in 30 minutes. Their current delivery model has a shipment loaded onto a drone which takes off, converts to a plane, flies to the vicinity of the landing local, converts back to a drone, and lands on a marker put out by the recipient. This drone then deposits the package and returns to the factory for its next package. Here lies a major flaw; in the event that something valuable is being shipped, it would be desirable for their to be a confirmation that someone is available to pickup the package that is being delivered. We aim to solve this by creating a drone that will wait for customer verification before the drone is delivered.

1.2 Objectives

1.2.1 Goals:

- Design an embedded system on a drone to authenticate delivery with customer
- Determine proper landing location when in close proximity of customer
- Wait until notified by customer to drop off package or leave with package if customer doesn't make contact

1.2.2 Functions/ Features:

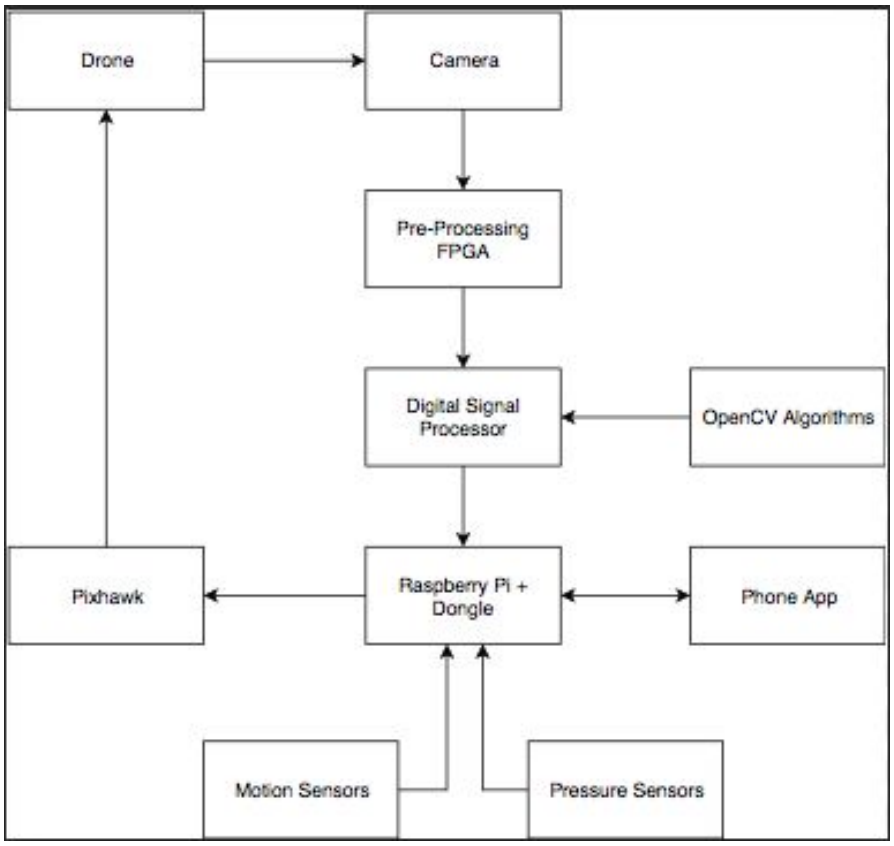
- Communicate with end-user through app when landing location has been found so that user may acknowledge delivery
- Identify and land in zone signified by fiducial marker
- PCB containing
- Observe surroundings for potential hazards to package using motion sensors
- Takeoff to evade any detected threats and light up red LED
- Have green LED light up when customer confirms package delivery
  (will not be implementing mechanics to drop off package as this is out of our scope)
- Takeoff with package if contact with customer is not made

1.2.3 Benefits:

- Enables speed of drone delivery but security of standard ground signature authorized deliveries for valuable packages
- Cost-effective automated delivery system

2.0 Design

2.1 Block Diagram



2.2 Block Diagram Description

2.2.1 Drone: This is the main component of the project. The drone will have a camera, an FPGA, a Digital Signal Processor (DSP), Raspberry Pi, Dongle, motion sensors, and force sensors attached to it.

2.2.2 Camera: There will be a 180 degree camera mounted vertically on the drone. This means when flying straight, there will be 180 degree view from the ground to exactly above the drone. This allows us to save weight by using only one camera in order to look down when landing and look ahead when scanning the surroundings for potential threats.

2.2.3 FPGA: In order to process the image signal, it may be necessary to do preprocessing on the image signal via an FPGA to ensure that all the image processing is done in real time.

2.2.4 DSP/OpenCV: The DSP will have the OpenCV algorithms ported to do the required image processing. OpenCV is an open sourced library with functions to accomplish real time computer vision.

Using these algorithms, the DSP will run a Proportional-Integral-Derivative (PID) controller in software to generate error signals to send to the Pi for processing.

2.2.5 Raspberry Pi + Dongle: Once the drone has reached the designated drop off point, it needs to notify the recipient. The raspberry pi and dongle will send a notification to the recipient's phone app to indicate arrival. The dongle will also receive confirmation that the recipient is ready to pick up the package, indicating that the drone can leave. The Pi will also receive the error signals generated from the DSP. Depending on these error signals, it will generate signals to send to the Pixhawk to order throttle power for each rotor. The Pi will also receive signals from the motion sensors and force sensors to adjust the throttle power as well. If the motion sensors are triggered, the drone will take off. If the force sensors - on the bottom of the drone legs - are triggered, the drone will power down.

2.2.6 Pixhawk: This is an open source controller used by our drone. It allows for a connection to a Raspberry Pi so that the Pi can execute commands to control drone operations.

2.2.7 Phone App: The phone app receives confirmation that the drone has arrived and allows the user to send the drone a confirmation signal that they are in the vicinity and ready to pick up the package.

2.2.8 Motion Sensors: There will be one motion sensor placed on each side of the drone. These 4 sensors will monitor each side of the drone for potential movement. In the event that movement is detected, a signal will be sent to the Pi to take off. The type of sensor being used is a basic PIR motion sensor.

2.2.9 Force Sensors: There will be one force sensor placed on the bottom of each drone. Image processing will be used to an extent to determine the height of the drone. Once it is known that the drone is only a few feet off the ground, the drone will slowly descend the final few feet until the force sensors are triggered on the ground. At this point, the drone will power down.

3.0 Requirements and Verification

3.1 Requirements and Verification Table

Requirements	Verification	Points
Voltage Source (Battery Pack) ● Meet all power requirements for DSP, Raspberry Pi, and camera modules for at least 5 minutes of flight time.	1. On drone landing, force sensor triggers LED and motor completely stops within 2s delay. 2. Draw expected load from battery pack and ensure can last at least five minutes, trigger failsafe landing if battery critical.	**5**
Camera Modules	1. Measure latency between change in environment and corresponding drone movement to ensure data is	**10**

● Supply image stream to DSP for processing at 5fps or greater to ensure adequate data to sample	being processed quickly	
FPGA ● Filter images for DSP processing use	1. Check output from FPGA on the DSP and ensure FPGA can send correct signals based on proper filtering of image	**5**
DSP ● Process images from camera modules and send through image processing software (OpenCV) for analysis	1. Provide static images simulating FPGA output and ensure DSP outputs anticipated signals for Pi.	**25**
Image Processing Algorithm ● Using output from image processing as inputs to a PID controller implemented in software. This will generate error signals to output to Pi.	1. See whether drone successfully arrives within designated landing zone of 5ft +/- 2.5 ft above ground within a 50 square foot landing region 2. See whether drone successfully takes off to safe hovering altitude (50 ft) in the event that someone approaches drone while it is hovering in landing zone	**20**
Raspberry Pi ● Process error signals from DSP to generate signals for each motor to adjust course appropriately ● Process signals from force and motion sensors ● Also responsible for sending push notification to user app over WiFi and receiving user confirmation	1. Move drone in all four directions 10m without input from the manufacturer provided 3G controls. 2. Manually trigger GPIO pin input for destination arrival, send text message to verify correct logic triggered. 3. Ensure LED switched on when user confirms delivery.	**20**
App ● Push notifications to indicate drone arrival ● Send signal to Pi to authorize package delivery	1. Manually trigger drone push notification, ensure app displays notification. 2. Click button to acknowledge package delivery, ensure LEDS on drone for package receipt triggered.	**5**

Motion Sensor ● Monitor surroundings for potential threats approaching the drone	1. Start with stationary object 35m away. Move in 5m increments until verify LED on drone for threat detection triggered. 2. Ensure drone has reached 50t off ground within 5 seconds of LED trigger.	**5**
Force Sensor ● Fixed on landing stilts of drone ● Send signal to shut off power when the drone lands	1. Apply increasing levels of force in increments of 5 Newtons 2. Ensure once target force is reached LED is triggered.	**5**

3.2 Tolerance Analysis

The major hardware component of the project that will most affect the outcome of this project is the accuracy and reliability of the PIR sensors. It is vital that the range that these sensors work accurately is relatively far away. Ideally, the sensors would work up to distances of 25 feet. With this range, there would be plenty of time for a sensor to detect motion, send a signal to the Pi, and fire up the motors so that the drone takes off in time to avoid the threat. In the event that the sensors cannot accurately tell that a threat is approaching until closer (i.e. 15ft) there is a chance that the assailant could get to the drone and steal the package before the drone is able to take off.

In addition, another problem that we could run into is receiving false positives from the PIR sensors. A false positive isn't necessarily catastrophic because it doesn't effect anything critical to completion of the mission; the drone would take off to a safe hovering height but would come down in the event that a confirmation message is received from the user. With that being said, it is still undesirable because it would use up a considerable amount of power in the event that the drone is forced to hover for any length of time. Depending on the frequency we see false positives during testing, we may consider adding 2 sensors on each side of the drone to add redundancy into the check.

4.0 Cost and Schedule

4.1 Cost Analysis

4.1.1 Labor

Name	Hourly Rate	Total hours invested	Total
Raymond Hoagland	$50	200	$25000
Rahul Joshi	$50	200	$25000
Sachin Weerasooriya	$50	200	$25000
Total			$75000

4.1.2 Parts

Item	Cost
Flytrex Sky Quadcopter	$549
C6000 DSP + ARM Processor OMAP-L138	$48.50
DS32EL0124SQE/NOPB FPGA	$17.22
180degree Fisheye Lens 1080p Wide Angle Pc Web USB Camera	$45
PIR Motion Sensor (4)	$9.95 ($39.8)
Force Sensor (4)	$7.95 ($31.8)
Raspberry Pi	$0 - already own
WiFi Dongle	$0 - already own
470 Ohm Resistor (4)	$0 - already own

LEDs (4)	$0 - already own
Total	$734.44

Total

Name	Hourly Rate	Total hours invested	Total
Raymond Hoagland	$50	200	$25000
Sachin Weerasooriya	$50	200	$25000
Rahul Joshi	$50	200	$25000
Grand Total			$75000

4.2 Schedule

Week	**Task**	**Member(s)**
Week 1	Finalize Proposal	All
	Finalize drone and camera purchase	Sachin Weerasooriya
	OpenCV Installation/Look into Ardu-Copter source-code	Rahul Joshi
	Research on open-source drone systems	Raymond Hoagland
Week 2	Design Review	All
	Finalize microprocessor purchase	Sachin Weerasooriya
	Computer Vision research--libraries to use, algorithms,etc.	Rahul Joshi
	Drone Systems research--encodings, range, etc.	Raymond Hoagland
Week 3	Begin developing code for DSP.	Sachin Weerasooriya
	Analyze current signals used to	Rahul Joshi

	control drone.	
	Either recreate current signals or determine how to create them	Raymond Hoagland
Week 4	First test flight to ensure quadcopter can be properly controlled using generated signals	All
	Continue work on analyzing feed using DSP	Sachin Weerasooriya
	Continue work on Raspberry Pi signal generation	Rahul Joshi
	Begin integration between DSP signal and Raspberry Pi	Raymond Hoagland,
Week 5	Begin work on Lidar detection	Sachin Weerasooriya
	Begin developing user application	Rahul Joshi
	Begin work on Fiducial sensor detection	Raymond Hoagland
Week 6	Mock Demo	
	Second test flight, check progress of DSP algorithm	All
	Finalize Lidar detection skeleton	Sachin Weerasooriya
	Finalize app skeleton	Rahul Joshi
	Finalize Fiducial sensor detection skeleton	Raymond Hoagland
Week 7	Integration and prepare for third test flight	All
	Begin detection for sound	Sachin Weerasooriya
	Integrate user application with Raspberry Pi communication	Rahul Joshi
	Integrate motion sensor for hovering position	Raymond Hoagland

Week 8	Third test flight, ensure drone can find landing and navigate to it	All
	Optimizing Control Loop	Sachin Weerasooriya
	Work on core functionality of phone app	Rahul Joshi
	Work on Raspberry Pi communications with app	Raymond Hoagland
Week 9	Fourth test flight, ensure hover standalone behavior.	All
	Force sensor for landing	Sachin Weerasooriya
	Finalize App	Rahul Joshi
	Finalize Raspberry Pi Communications	Raymond Hoagland
Week 10	Final test flight, ensure drone can land and hover when motion detected	All
	Finalize Force sensor PCB	Sachin Weerasooriya
	Drone controls given analysis of Force sensor output	Raymond Hoagland
	Force sensor output and analysis on Raspberry pi	Rahul Joshi