Autonomous Vehicle Processor

DESIGN DOCUMENT

Dec1710
Rockwell Collins - Josh Bertram
Philip Jones, Joseph Zambreno
Team Leader - Alexander Orman
Communications Leader - Sean Jellison
Webmaster - Chris Kelley
Key Concept Holders - Lixing Lin, Evan Lambert
Scribe - Lucas Ince
dec1710@iastate.edu
http://dec1710.sd.ece.iastate.edu/

Revised: 03/2017, Ver. 1.0

Contents

1 Introduction

1.1 Project statement

The purpose of this project is to utilize an onboard system in realtime to provide information to flight systems regarding location of objects in a drone camera's view. To accomplish this we use a pre-trained neural network to identify complicated objects and then a series of other, simpler methods to extract additional information such as distance away from the drone.

1.2 Purpose

This software will utilize a system's GPU to be able to more accurately and efficiently apply deep learning concepts to embedded real time system processing. It will be able to detect and analyze objects and provide relevant information about it.

The use of a neural network operation with real time data on onboard hardware has a wide range of applications beyond this project. While many military applications come to mind, there are plenty of civil uses as well to better the world. A good example of this would something like Amazon's drone delivery service. If they could utilize this system, it could save time and money for the company by avoiding objects and ensure more happy customers by putting packages in more convenient and secure locations.

1.3 Goals

When finished, we want our system to be able to be installed onto any autonomous system and be able to provide accurate and useful information about the surroundings. This is intended for use on remote controlled drones, but this could also be applied to autonomous vehicles to detect objects near the vehicle or surveillance systems to detect intruders. The ability to detect a generic object has a multitude of uses already, but our system will be able to be trained for specific purposes. Additionally, the ability to easily the network to detect novel objects.

Lastly, here is a list of tentative features we hope to achieve in order of under taking:

- Simulation data collection
- Distance and relative object orientation calculation
- Geographical position detection based on landmarks
- Object trajectory predictions
- Multi Camera and alternative camera support
- Wingmate and other aerial object tracking

2 Deliverables

We aim to provide software that can be installed onto an embedded board for use in autonomous aircraft. The software will be able to detect landing strips, runways, known airports, and be able to analyze additional flying objects. It will also be able to track objects and point out landmarks or significant structures.

Specifically, these items will be released to the client upon completion:

1. Source code for application developed, unless otherwise specified we(the students) may share and reuse any and all source code.
2. A PDF guide explaining installation and usage of source code aforementioned
3. A powerpoint explaining the project
4. Any videos generated during this project regarding the project.

3 Design

Conventional Computer vision: We researched standard computer vision techniques, and determined that standard methods were slow and unadaptable. We were looking for a system that could be updated in the future to determine multiple different object types. We also wished to limit the amount of computation necessary.

Neural Networks: We decided to use a neural network to solve the problem because of its lightweight approach to solving complex tasks. We are using a convolutional neural network to limit the level of computation needed for the image processing. We are avoiding recurrent networks to simplify the program also. .

3.1 System specifications

Initially, use NVIDIA tutorials to implement some type of visual object recognition or similar task and use a mockup environment in the lab. (Toy airplanes on sticks, etc.) As confidence grows, Rockwell Collins can potentially provide some video footage from one of our drone flights this spring or summer. The specific objective of the visual system can be negotiated; examples might be runway detection, detection of another nearby aircraft, etc.

We will not receive any outside data besides what is collected by the camera. We will need to generate any required information through image processing.

Any goals not stated above are choices the design team have chosen to implement and are assumed about the project, such as image processing speed and quality:

- Board: Jetson TK1 board mounted on a aerial platform
- Camera: See3CAM_10CUG_M
- Machine Learning Framework: TenserFlow
- See functional and nonfunctional requirements

3.1.1 Non-functional

- Embedded system must fit inside the drone
- Must be able to process an image at least once per second from onboard camera

- Ideally, target goal is 30-60 FPS

3.1.2 Functional

- Runable on the embedded system.
- Can detect basic objects.
- Must be able to take in images from a camera
- Be able to locate key features in an image
- Be able to identify known objects or features from 400ft - 600ft

3.1.3 Standards

As of yet we have not encountered a need to have a standard to follow or set up. As a general rule we will follow established good programing practices for C, C++, and Python. Theses are the main languages used for the training and of our neural network , and the post processing.

3.2 PROPOSED DESIGN/METHOD

As a team, we have decided to implement a machine learning/deep learning approach to this problem. We plan to utilize a convolutional neural network to identify objects such as runways, airports, and other planes/drones. A camera will provide input for the network and any pre-processing for the network. The network would use that input to determine what type of object is in the frame. The network output would be sent for post-processing which could be used for detecting the orientation of an object, the distance of an object, and detecting other friendly drones.

3.3 DESIGN ANALYSIS

We have a basic camera setup with Tensorflow. We are still trying to order our specific camera for this project. We have done some basic neural network implementation for other objects but need to collect our runway and airport data for the project specific implementation.

4 Testing/Development

4.1 INTERFACE SPECIFICATIONS

Discuss any hardware/software interfacing that you are working on for your project. This section is decided by team advisor/client.

N/A

4.2 HARDWARE/SOFTWARE

Indicate any hardware and/or software used in the testing phase. Provide brief, simple introductions for each to explain the usefulness of each.

Tenserflow:

Jetson TK1:

OpenCV:

See3CAM_10CUG_M : 1.3 MP Global Shutter Monochrome Camera with 720p HD and support for trigger modes.

a. USB 3.0 device with USB 3.0 Micro-B connector. Our Standard camera board will have USB 3.0 Micro-B connector.
b. Supported OS: Windows 10, Windows 8, Windows 7 & Linux
c. Pixel size:3.75µm x 3.75µm

4.2 Process

As of now we have only been using theory and a test Tenserflow neural network on a Python script. All tests have been manual tests feeding in images to the network. We have set up scripts to automatically gather test photos from the simulator and have found datasets online.

5 Results

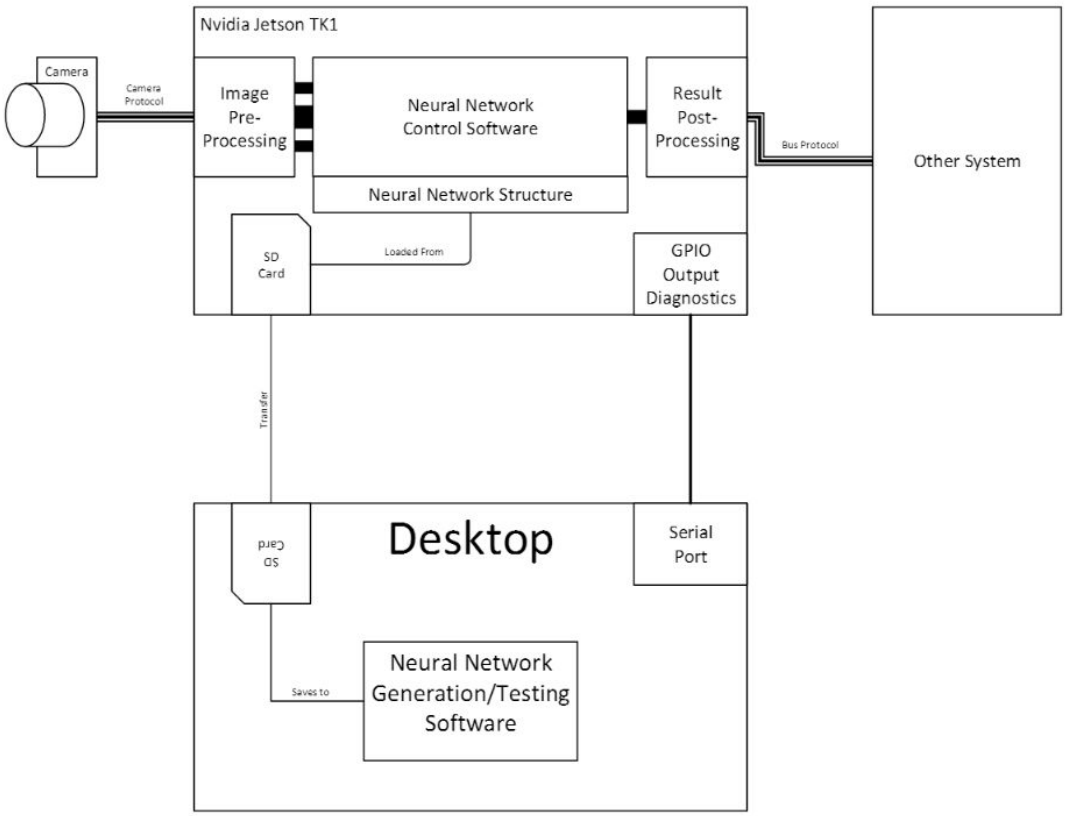
List and explain any and all results obtained so far during the testing phase. Include failures and successes. Explain what you learned and how you are planning to change it as you progress with your project. If you are including figures, please include captions and cite it in the text.

6 Conclusions

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

7 References

List any references used in the document. These are an essential part of your review so far.



8 Appendices

If you have any large graphs, tables, or similar that does not directly pertain to the problem but helps support it, include that here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc. PCB testing issues etc. Software bugs etc.