Use of Imaging Devices and Machine Learning Software to Assist in Autonomous Vehicle Path Planning

DESIGN DOCUMENT

Team 3

SmartAg: Client

Joseph Zambreno: Advisor

Souparni Agnihorti: Meeting Facilitator

Ashley Dvorsky: Document Manager

Eric Himmelblau: Webmaster

Fahmida Joyti: Test Master

John Orefice: Communications Lead

Bowen Zhang: Hardware Maintainer

Team Website: sdmay18-03@iastate.edu

Version 1: Revised October 15th, 2017

Table of Contents

List of figures/tables/symbols/definitions

1 Introduction

1.1 Acknowledgement

We would like to express our gratitude Department of Electrical and Computer Engineering who offer EE 491 class, and because of this, we have an opportunity to work together, communicate together, and learn together. Thank you to the Dr. Joseph Zambreno who is teaching this class.

Second, the project team is thankful to Smart Ag who propose this project to our senior design class. We would also like to thank Mark Barglof and Thomas Antony from Smart Ag who are leading us to do this project, and our adviser Dr. Joseph Zambreno who gives us advice for our project.

1.2 Problem and Project Statement

We will be using object detection and depth perception to ensure that tractors driving according to a path planning algorithm do not hit unexpected obstacles.

We are using a stereo camera system and an Nvidia GPU housed in the tractor. We will detect and recognize an obstacle (using object detection algorithm), classify it, and find its latitude and longitude to use as an input to the path planning algorithm allowing our vehicle to circumvent the obstacle efficiently.

1.3 Operational Environment

This product will be working on mid-sized farms. It is exposed to all conditions including rain, dirt, snow, but in moderation. If the weather is to the point that the conditions are not safe for the tractor or that the human observer would not be able to watch and effectively judge if they should deploy the emergency stop button this product will not be used. The exact details of the environment may vary, but the goal of this system is to allow the tractor to adjust accordingly.

1.4 Intended Users and uses

The target audience for this project is farmers, specifically farmers who have other time commitments which require their attention. It greatly reduces the time spent by farmers on the farms for harvesting as the autonomous tractors can do it for them. It will also reduce costs and increase outputs for the farmers.

1.5 Assumptions and Limitations

1.5.1 Assumptions:

- The Convolutional Neural Network provided by the client is working as intended.
- The data acquired by the stereo camera system is accurate.

- Our tractor can see in at least 10 meters in front of them., otherwise an emergency stop will be deployed.

1.5.2. Limitations:

- The additional cost for the camera sensors is limited to $1000
- It will need to be powered by the tractor electrical system.
- Lack of a pre-defined dataset to train our model. We need to do image acquisition to get the data before we can start analysing the data.
- Testing limitation: Training of our model would be done during harvest season but we will be unable to test the model we create on the same conditions because the testing will be done during winter.

1.6 Expected End Product and Deliverables

At the end of the product, we should have a fully functional object detection system that can identify any object that is within the scope of our dataset or any object that looks similar to another object in the dataset.

In addition, we should also have a depth perceptive algorithm that can find how far an object is relative to the tractor.

2 Specifications and Analysis

2.1 Proposed Design

Our current design is to use dual cameras to capture the environment of an autonomous tractor. The video feed from these cameras is fed into a convolutional neural net which identifies and classifies objects of importance (for example fences or ditches). The location of these objects in the images is used by a distance determination system which compares the differences in the stereo images to determine the displacement from the tractor. This distance is combined with the tractors current GPS coordinates to determine the GPS location of the object. This information is then delivered to the path planning algorithm and added as an obstacle for the tractor to avoid. These computations are performed on an Nvidia GPU to allow everything to occur in real time.

There are three requirements for this project:

1. Capture video and perform object classification with a Convolutional Neural Network framework.
2. Calculate the position of objects relative to the tractor.
3. Convert to GPS location and upload to path planning system

2.2 Design Analysis

We have begun doing "hello world" type proofs of concept for various pretrained neural nets including darknet, dark flow, and mobile-SSD. It is fairly simple to interface these neural nets with OpenCV which makes it simple to connect them to the video feed and will allow us to do further analysis of the image after object classification has been performed. While they all seem to be fairly accurate, they perform very slowly. Running on a CPU, they average under 1 frame per second, and we have had trouble getting any of them to run successfully on a GPU. This primarily seems to be caused by compatibility issues between the Tensor flow backend in Python and newer versions of Nvidia GPU libraries. Regardless, SmartAg has tested dark flow on the GPU which will be used in production and still only achieved around 4-5 frames per second. This performance will not be able to satisfy our requirement that this run in real time, so we will continue to investigate different neural nets. Additionally, we have found that using smaller, more selective, weight files helps to improve performance at the cost of a loss in accuracy, so we are investigating a balance between the two. If we are still unable to achieve real time performance, we will investigate using libraries in C, rather than Python,

The second major part of this project is the distance calculation. Initially our client intended to use a single 360° camera for video capture, but were also open to the idea of using stereo cameras. After investigating methods for distance determination using single and stereo cameras, we have found that it will be far easier to implement using a stereo camera system. This information, combined with problems that SmartAg has had using the 360° camera, is causing us to focus on the stereo camera system over the 360° camera. Combining accurate stereo distance calculations with a GPS which is accurate to under two inches will allow for precise GPS coordinates to be provided to the path planning algorithm.

3 Testing and Implementation

3.1 Interface Specifications

This project consists of two hardware interfaces and two software interfaces. The hardware interface consist of interfacing between the stereo camera and image recognition system and interfacing between the GPS and distance determination system. The software interfacing consists of interfacing between the image recognition system and distance determination system and the distance determination system and the path planner.

The interface between the camera and the image recognition system will be achieved using OpenCV. OpenCV has library calls which we can use to use the capture the video feed. Once we capture the video feed we can use the image recognition system to detect and classify the objects in front of the autonomous tractor. This information will then be available to OpenCV allowing us to do distance determination calculations.The image recognition system will give information about the object in the form of JSON array which we can pass to the distance determination system . The distance determination system will send the information to the path planner created by our client using a restful api call. The connection between the GPS and the distance determination still needs to be determined.

3.2 Hardware and software

We will be writing automated tests using the Python UnitTest module. It is a common testing framework provided by Python and has similar behavior like Junit in Java. This will allow us to determine correct behavior for parts of our system. Also we will be using unittest.mock which will allow us to replace parts of our system with mock objects. Hence, we will be able to test our distance determination system without relying on the image recognition system as we can use mock objects instead. However, these tests will only allow us to catch defects in early stage of our development stage. To test the accuracy of our model we will need to train our model first then use a validation set to further refine our model and then use a test set to test the accuracy of the model generated by our image recognition system.During this phase we will be relying be on the Nvidia TX2 provided by our client as it would significantly speed up the computations performed by our image recognition system . Also during the testing of interfacing between the GPS and distance determination system we will have to use the GPS provided by our client . Likewise, we will have to use cameras to determine if the cameras are interfacing correctly with the image recognition system.

3.3 Process

To start testing our project, it made sense for us to try experimenting with the different frameworks and determine which would be the best fit for our image recognition system.We started by trying to interface the various frameworks suggested by our client with the webcam in our laptop using OpenCV. For preliminary testing purpose we tried to use the GPU in our laptop which is slightly faster than the Nvidia TX2 GPU provided by our client. While we were able to use the frameworks to identify various kind of objects, we have been not be able to configure the framework to use the GPU instead of our CPU. Hence, we have been getting a frame rate of only 5 frames per second instead of a rate of 15 frames per second we would be getting if we used the GPU instead. This is a difficulty that we intend to resolve so that we can determine correctly the ideal framework we should be using for our project and focus on more important tasks such as training and refining our neural network model and implementing our distance determination system.

3.4 Results

So far we did not we not have much results as we recently divided our group tasks, but in terms of determining what our client expects from us has progressed well. This is because our client had worked on detecting humans and knows the kind of difficulty that might arise while working on projects that require the use of neural networks. We have acquired around 200 images so far and we will be creating more images by modifying our existing images such as changing the lighting of the image which we can use to train our model. We have been getting ourselves familiar with various neural network frameworks by reading the research papers associated with them and doing small "Hello World" experiments to compare the appropriate framework to use for our project. We have also determined that the quality of pictures taken by 360 degree cameras would not be very usable for our model and opted to use stereo cameras instead.

4 Closing Material

4.1. Conclusion

Our group will be working with SmartAg to use image recognition software to detect obstacles that would commonly be found in farming fields. Once an object is detected we will return the location information to the path planning algorithm to avoid said obstacle and then also store the gps coordinates to the map to avoid said obstacle in future path plans.

As a team we have researched extensively the different options for making this project a success. We compared the benefits of 360° cameras, lidar sensors, stereo cameras to collect environmental data and we have decided that the costs of the lidar sensors wouldn't make them a feasible option, and the 360° camera had too many issues with tilt shifting and unwrapping affecting the quality and usability of that image.

We decided to use OpenCv as our image recognition software as that is industry standard, and had more resources available to us including our client team. OpenCv also interacts seamlessly with our neural network options including darkflow, darknet, and mobile SSD.

4.2. References

1. Allanzelener. "Allanzelener/YAD2K." *GitHub*, 2 July 2017, https://github.com/allanzelener/YAD2K.
2. Balancap. "Balancap/SSD-Tensorflow." *GitHub*, 10 Apr. 2017, https://github.com/balancap/SSD-Tensorflow.
3. "Feed the World Smarter." *Smart Ag*, https://www.smart-ag.com/.
4. Thtrieu. "Darkflow." *GitHub*, 24 Sept. 2017, https://github.com/thtrieu/darkflow.

4.3. Appendices