PROJECT PROPOSAL

HawkEyeSD

An autonomously navigating Scan-Drone that creates 3D-Models of real-world objects and optimizes them in real-time

Lukas Mutter

28.05.2018

CONTENT

[Summary 3](#_Toc515955281)

[Introduction 4](#_Toc515955282)

[Needs/Problems 4](#_Toc515955283)

[Design theory 5](#_Toc515955284)

[Procedures/Scope of Work 5](#_Toc515955285)

[Goals/Objectives 6](#_Toc515955286)

[Next Steps 6](#_Toc515955287)

[Features 7](#_Toc515955288)

[Endorsements 8](#_Toc515955289)

[Appendix 8](#_Toc515955290)

[Legend 8](#_Toc515955291)

# Summary

The Nvidia Jetson HawkEye Scan-Drone is designed to be operated by untrained persons using a standard android smartphone, allowing them to create high-quality 3D-models of objects without any human work. The user points the camera of the smartphone towards an object of interest and does the selection via the touchscreen. The drone then begins to autonomously orbit around the object and creates a polygon optimized model of the object whilst using the obtained data to navigate. The obtained polygon model can then be downloaded by the user via the USB-port or Wi-Fi.

This project is open source. If you like the project and you want to develop new skills by just doing great stuff: This is a great opportunity to learn cutting edge technology. Feel free to download the project files (see [Next Steps](#_Next_Steps_1)).

# Introduction

Hello, my name is Lukas Mutter. I live in Germany and I am currently studying robotics, artificial intelligence and parallel computing.

I want to give you a little explanation of how this project came up:

It all started in 2015, when I was working for a water-power-plant manufacturing company in Germany as a measuring technician. When I started the apprenticeship as mechatronics technician at the same company in 2012, I thought that I would be the guy that actually designs and builds mechatronic systems. I quickly realized that this was not the case. After I finished my apprenticeship, I immediately went to the Abendgymnasium to be allowed to study. The delivered education was of such a bad quality and, for me, totally useless knowledge, so I decided to finish this school in 2 years and take the Fachhochschulreife (FH) degree.

At this point I knew that a German university is not the right thing for me, so I did some research about studying at Berkeley or MIT. But no matter how I calculated it: I simply had not enough money. But that did not stop me from reaching my goals. I started to take online courses at different online platforms like Udacity or Edx and started teaching myself.

Then, in September 2017, when I was taking a robotics course, Nvidia offered me a Jetson Tx1 development kit for a special price. Unfortunately it was catching some dust, waiting for the right idea to come up…

# Needs/Problems

I love playing video games but I was bored of the available games so I started to make my own game within the “Unreal engine”.

But there is one major problem that every game developer has: 3D assets!

3D assets are 3D models used by game developers. Every object in a video game has some sort of polygon mesh which represents its shape. Traditional methods of creating 3D assets include hand sculpting with CAD software, 3D scanning and photogrammetry. All of these techniques require excessive human work to get usable results. Whether it is cleaning the model from unwanted ‘noise' or reducing the polygon count to optimize its memory usage, special knowledge and a lot of time are required.I quickly realized that there is big potential. Not only can the time needed to produce those assets be reduced. It can also be simplified to allow people with little or no experience in CAD to produce AAA assets.

# Design theory

The drone’s main board is fully able to control the drone without the Jetson module. The on board FMS controls the ESC’s via PPM.

Sensors can be attached through the expansion connector. The on-board Sensors act as a backup.

The FMS computes the drone’s theoretical position and tries to hold it. It also handles maneuvers as landing etc. The theoretical position from the FMS gets constantly stream to the Jetson module. The Jetson module then streams instructions to the FMS. The Jetson module is in direct control of to lower resolution cameras (mounted at the sides of the drone, facing forward) and one high resolution camera (mounted at the front of the drone). The fixed difference in position of the two lower resolution cameras allows the computation of depth. However, this is an extremely complicated computational task. Therefore a “neural net" has been chosen to do the job. It is trained through the “Unreal engine”. This training this one of the most critical parts of this project.

After successful training, the drone will (hopefully) be able to recognize specific features such as edges, corners, curves, colors, etc. and keep track of them in space. This allows to significantly reduce the computational task of the next frames. The high resolution camera is then used to upscale the depth resolution in certain areas of interest.

The specific features get saved and verified while the drone orbits around. This is done by saving each feature with a “probability” that gets updated in future frames. After reaching a specific save threshold the feature gets permanently saved to the finished model and is ignored in next frames, to further reduce the computation. When there are no unsaved features within a certain threshold (e.g. only features with a probability of fewer than 10%) and the saved features can close the model, the drone hovers to its start point and completes the model by removing and closing the ground plane.

# Procedures/Scope of Work

The scope of this work includes the actual **drone motherboard** with **integrated Flight Management System** and **ESC**s on the hardware side and a **trained “neural net”** and an **android app** on the software side.

The FMS and the ESCs can be updated through the Jetson module. A charge-control/BMS connector is also routed on the mainboard.

The ESCs consist of a modified reference design from Texas instruments (TIDA-00643) which has been modified to ensure programmability and connectivity.

# Goals/Objectives

One of the main goals is to adapt the PX4 firmware to the motherboard. This allows the drone to do all the processing needed to navigate on-board.

The next main goal is to get all the cameras up and running. A device-tree has to be written for the Jetson module. This allows the module to load the corresponding drivers.

The third main goal is to write the drivers that allow communication between the Jetson module and the FMU. At this point the drone is ready to fly.

Now the fun begins! A “neural net” has to be written. This is probably the most challenging part of the project, a process of trial and error. It has to be capable of handling all the computer-vision in real-time whilst outputting instructions for the FMS.

The final goal is the training of the “net”. Without proper training, the “net” will never work as intended. No matter how good it actually is!

After the drone’s part of the project is ready, it is time to design an android app that allows interfacing the drone. Due to the fact that developing android apps is pretty easy nowadays, this is more of an objective.

# Next Steps

This project is still in progress, but you can download it on [my Github page](https://github.com/FinestArcadeArt/Jetson-HawkEyeSD). If you want to contribute to the project, you can do this via Github. Detailed information about the project status will be accessible there.

The project’s progress:

* Modifying and manufacturing some prototype ESCs
* Designing a prototype
* Manufacturing a prototype PCB
* Writing the FMS firmware and the Jetson module’s device tree
* Testing of the FMS
* Ordering some custom camera modules from China
* Writing the ROS code for the Jetson module
* Testing of the camera dedicated functions
  + Optional second prototype
* Designing the AI that handles the polygon creation/reduction
* Training and optimization
* Designing an android app

# Features

The mainboard is designed to work with both of Nvidia’s current modules, the Tx1 and the Tx2. It has been designed using Nvidia’s developer board as reference design.

a Depending on which board is used, the following features are different. a

Nvidia Jetson Tx1

* NVIDIA Maxwell ™, 256 CUDA cores GPU
* Quad ARM® A57/2 MB L2 CPU
* 4 GB 64 bit LPDDR4 25.6 GB/s memory
* CSI2 D-PHY 1.1 (1.5 Gbps/Lane) camera interface

Nvidia Jetson Tx2

* NVIDIA Pascal™, 256 CUDA cores GPU
* HMP Dual Denver 2/2 MB L2 + Quad ARM® A57/2 MB L2 CPU
* 8 GB 128 bit LPDDR4 59.7 GB/s memory
* CSI2 D-PHY 1.2 (2.5 Gbps/Lane) camera interface

a Both modules mate with the 400-pin connector on the mainboard.

Mainboard

* 32bit STM32F427 Cortex® M4 core with FPU
* 32 bit STM32F103 failsafe co-processor
* ICM-20948 9-axis Motion Tracking device
* MS5611-01BA altimeter sensor
* Sensor expansion connector
* 2 dual lane CSI2 camera connectors
* 1 quad lane CSI2 camera connector
* Onboard camera power rails
* Jetson controlled MCLK
* Camera I2C multiplexer
* 4 ESC connectors
* Charge control connector
* Solder points for power source
* USB 3 Type-C
* Power monitoring
* Dedicated configuration EEPROMs for the board itself and the camera interposer module (onboard)
* Configuration EEPROMs on each of the camera modules

# Endorsements

Names and addresses of individuals and companies who support and endorse the project will be listed here. Your support to the project will be honored depending on your contribution.

# Appendix

You can find additional information about the Nvidia Jetson module on [Nvidia’s official developer page](https://developer.nvidia.com/embedded/develop/hardware).

# Legend

FMS Flight Management System

BMS Battery Management System

ESC Electric Speed Controller (motor controller)

PCB Printed Circuit Board

Abendgymnasium German evening college-preparatory school

CSI Camera Serial Interface

AI Artificial Intelligence

ROS Robot Operating System