AprilTag Journal

# AprilTag C Code

Broke apart the algorithm into the distinct steps as seen in the main paper. This is so it will be easier down the line to start and end at any point.

Created visual outputs for each step.

Organized all the code to run from a single solution with a single executing project. Work still needs to be done to split apart the different variations of the code. Maybe have them loaded as DLLs instead of static libraries??? This will require me to dig a little deeper on DLLs but is not the focus of my graduate studies.

1/5/2021

I figured out a way to split the code via a single solution and project. I realized that a lot of the code is the same so instead of copying it, I have a single baseline code. Any steps that need to be changed or excluded can be done so manually with its own code and step process. Now, ever different flow will have to implement their own “Demo” and “Extraction” processes. The “Demo” will control how images get loaded and some of the default parameters. The “Extraction” will control the actual AprilTag algorithm, what images to load, from where, and where to save them. This process may leave out specific steps or swap out steps.

Implemented the cordic algorithm in C. There are some noticeable errors and changes within the algorithm although the end tag is still detected.

Timing analysis is coming soon. Everything is in place except some changes in the looping to get more accurate results. I need to figure out a way to export these times in a better manner (maybe a csv file?).

# Matlab/Simulink

Currently working with a different version of the apriltag code (ravventag) to test various algorithms as it is the only stable code working right now. This does not matter as the matlab algorithms will be translated into C code for verification. Then, that code will be turned into a Simulink model, verified again via simulation, and uploaded to the FPGA for live testing. Once each component is working, they will all be strong together.

1/19/2020

Finished, verified and ensured atan2 cordic co-routine works!

# Pynq/FPGA