SSC20-WP2-07: Design of Modular Star Tracker Software

Luca Merlo Paula Soares

Computer Engineering Department, California Polytechnic State University, San Luis Obispo, CA 93407, luca.mpsoares@gmail.com

Abstract

The current CubeSat attitude sensor landscape is bimodal. One end of the spectrum is occupied by solar angle sensors, magnetometers, and rate gyros. However, these low-cost sensors offer insufficient performance for missions that need higher pointing accuracy. Star trackers occupy the other end of the spectrum, providing much greater pointing knowledge accuracy. Unfortunately, the commercially available star trackers are prohibitively expensive for most modest-budget CubeSat missions. To address this problem, the Cal Poly CubeSat Lab (CPCL) decided to implement a star tracker targeted at filling the gap between commercial options and coarse sensors.

Project Requirements

The star tracker has a list of requirements it must fulfil before it can be used in CPCL's spacecrafts:

- It must be able to run on CPCL's flight computer.
- It must use less than 10Mb of storage space.
- It must use less than 16Mb of RAM.
- It must be able to acquire an attitude during a state of "lost in space."
- It must have better accuracy than CPCL's solar angle sensor.
- It must be compatible with other CPCL's software.
- It must be able accept input from different kinds of cameras.
- It must run in C or C++.
- It must provide a quaternion containing the rotation information from camera inertial frame to earth centered inertial frame and a time stamp.

About the Author

Space, the final frontier. These are the voyages of the Cal Poly CubeSat Lab's spacecraft. Their continuing mission: to explore strange new worlds. To seek out new life and new civilizations. To boldly go where no one… Oh wait, wrong star trekker.

I am a Cal Poly computer engineering alumni and I am currently working full time at Micro-Vu as a firmware engineer. I am extremely passionate about space exploration, which lead me to join CPCL. During my 4 years at college, I had the opportunity to work on many CPCL's missions as a member of the software team. I decided to work on this project as my senior project.

Process Architecture

The Star Tracker process was divided into 6 distinctive stages: reader, filter, finder, camera operations, identifier, and quaternion calculator (QUEST Algorithm[1]). The reader, filter, and identifier utilize object-oriented design to allow the star tracker to easily swap between different algorithms and drivers. Furthermore, this modular design allows for maintenance and expandability.

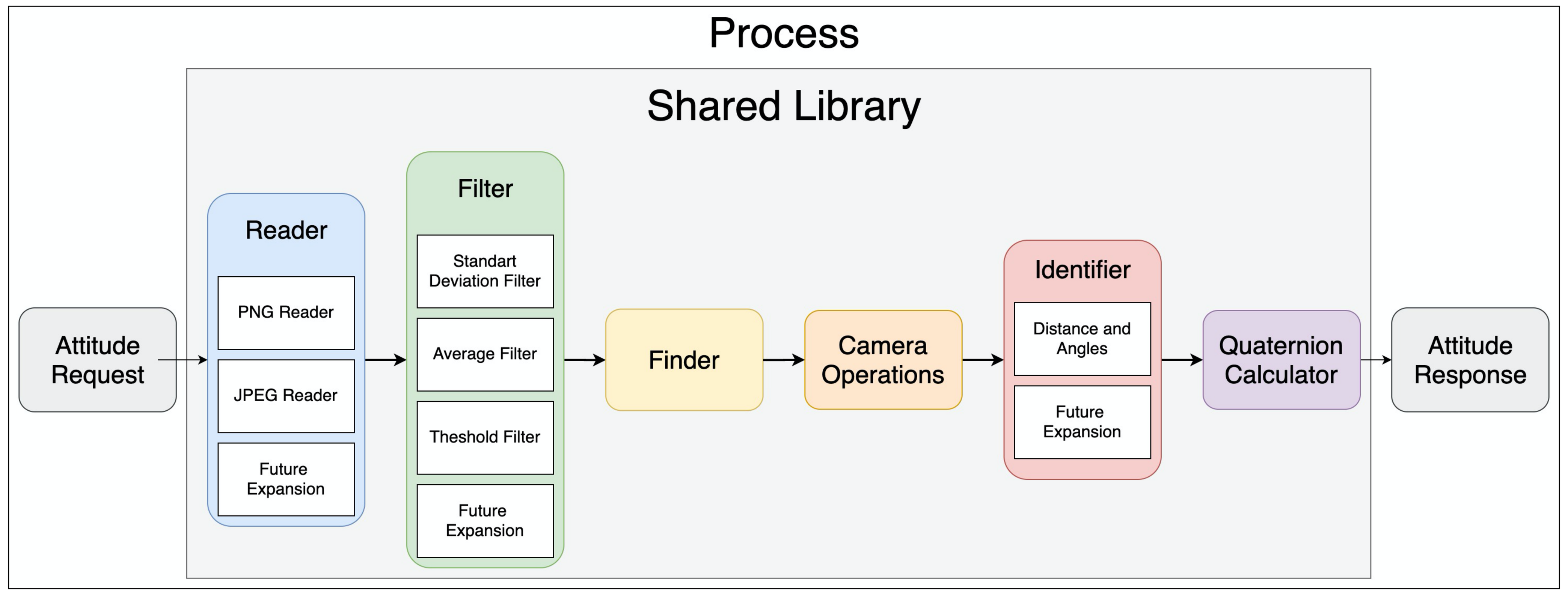
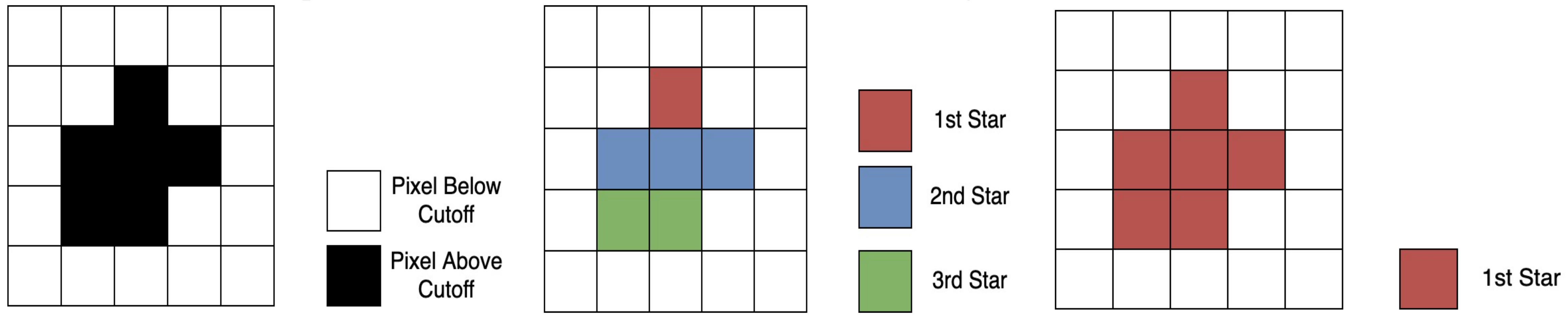


Figure 1: Shows diagram of the process' architecture and information flow.

[1]Anton H.J. de Ruiter, Christopher J. Damaren, and James R. Forbes, "Spacecraft Navigation," in *Spacecraft Dynamics and Control*, 1st Ed. Canada: Wiley, 2013, ch 25, sec 2, pp 467-476.

Star Centroid Calculation

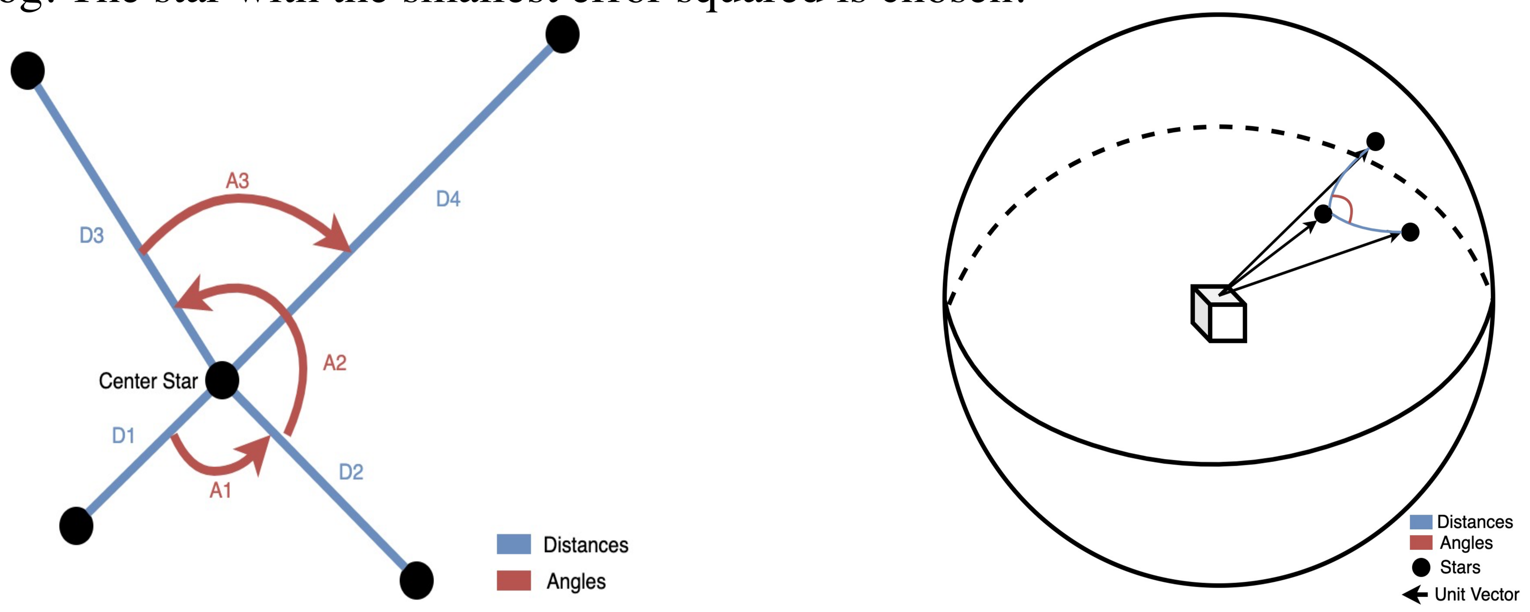
To locate the stars in the picture, that star tracker calculates a pixel cutoff value based on the standard deviation and average values of pixels in the image. After that, it groups up the pixels into star objects, first horizontally, then vertically. Finally, a center of mass calculation is performed to determine the star location in the image.



Figures 2, 3, and 4: Shows diagram of the steps used in the star finding algorithm.

Star Identification Method

After locating the stars on the image, correcting for lens and camera distortions, and transforming the 2D coordinates into 3D coordinates, the star tracker attempts to identify the center star. This is done by calculating the distance from the center star to its neighbors and the angle between the lines they form. Then, these values are compared with a custom catalog based on the Harvard Revised Catalog. The star with the smallest error squared is chosen.[2]

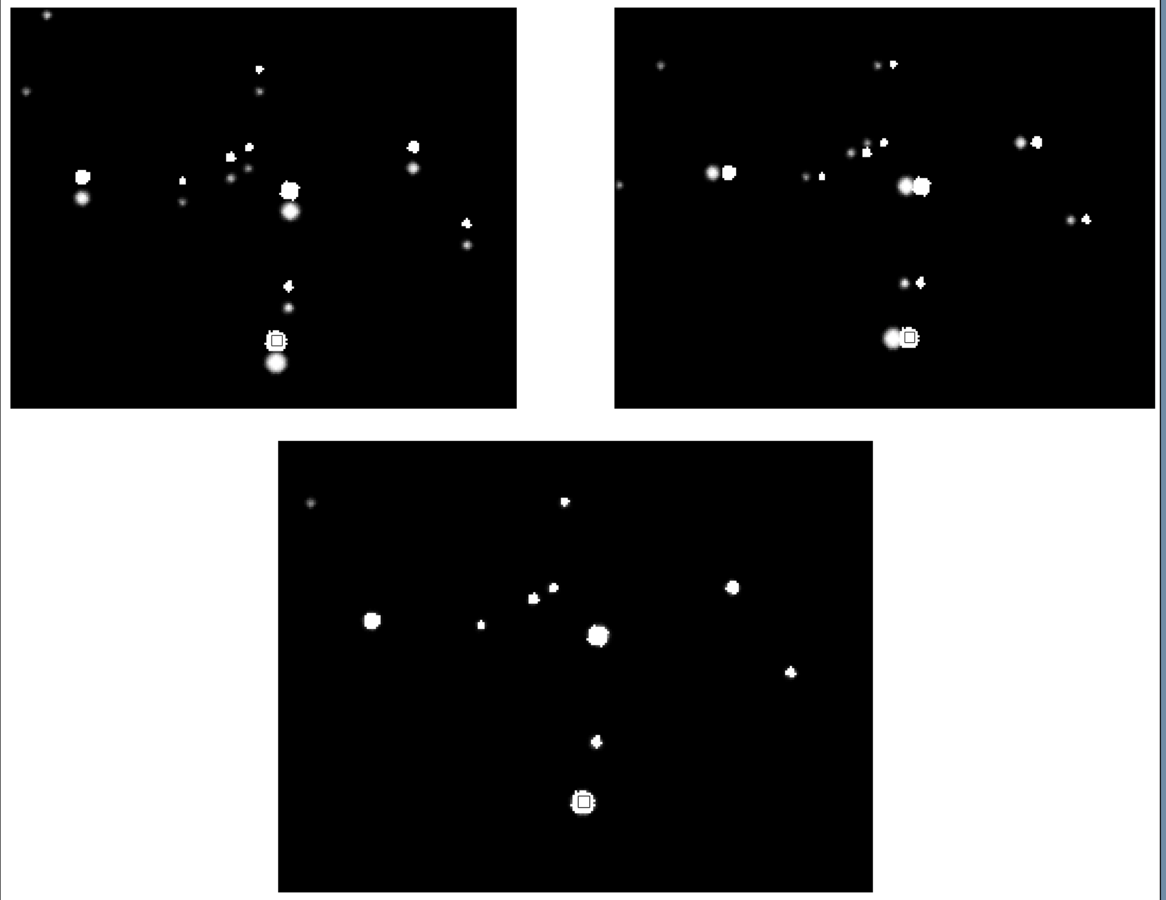


Figures 5 and 6: Shows the identification algorithm in 2D and 3D respectively.

[2]Gustavo Baldo Carvalho, "Levantamento de Técnicas de Identificação de Estrelas e Desenvolvimento de um Ambiente de Simulação e Testes para Análise de seus Desempenhos em Aplicações Espaciais," M.S. thesis. Instituto Nacional de Pesquisas Espaciais. Instituto Nacional de Pesquisas Espaciais, São José dos Campos, Brasil, 2000. INPE-INPE-8307-TDI/766

Testing

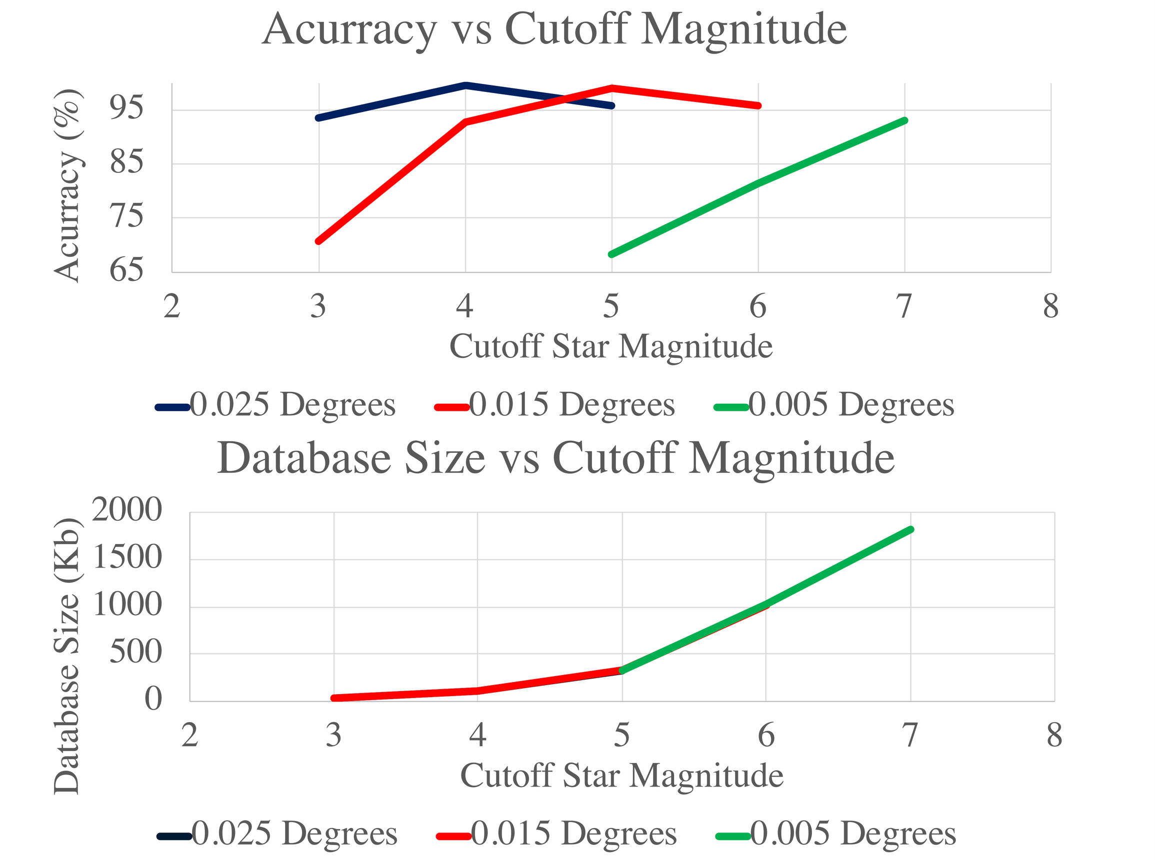
To test the star tracker a python script was used to generate images of stars. To ensure that it was generating correct images, they were compared with Dominic Ford's In The Sky database. The images below contain the star HR 2618 generated by both the python script and In The Sky database overlaid together.



Figures 7, 8, and 9: Shows the stars generated by both programs overlaid.

Testing Results

These generated images were inputted into the star tracker and the accuracy at correctly identifying stars was recorded as well as the generated database size.



Figures 10 and 11: Shows the results of the tests. Each color represents a specific field of view per pixel.

Conclusion

CPCL's star tracker shows that it is possible to create a simple, fast, modular, and small star tracker. Its modular architecture not only allows for easy maintenance and updatability but also enables it to utilize most cameras already in use by CubeSats. More importantly, it bridges the gap between low costs and expensive off the shelf star trackers.