INSTITUTE OF CONTROL AND COMPUTATION ENGINEERING
FACULTY OF ELECTRONICS AND INFORMATION TECHNOLOGY
WARSAW UNIVERSITY OF TECHNOLOGY

MASTER OF SCIENCE THESIS

STAR-TRACKER PROGRAM FOR CUBESAT SATELLITES

Szymon MICHALSKI

Supervisor:
prof. dr hab. inż. Ryszard Romaniuk

Warszawa 2017

Abstract

Last years directed space industry towards small satellites. Many countries which did not have possibility to enter this branch of industry before now create their own solutions. The goal of this work is to create fully functioning star-tracker software eligible to be used in future satellites as Polish solution of determination of satellite attitude towards Earth. Work contains also description of individual parts and variants of solutions connected with star-tracker.

Streszczenie

Ostatnie lata ukierunkowały przemysł kosmiczny na małe satelity. Wiele krajów, które wcześniej nie miały możliwości wejścia w tę gałąź przemysłu, teraz tworzą własne rozwiązania. Celem niniejszej pracy dyplomowej jest stworzenie w pełni działającego oprogramowania star-trackera nadającego się do wykorzystania w przyszłych satelitach jako polskie rozwiązanie problemu określania orientacji satelity względem ziemi. Praca zawiera też opis poszczególnych części i wariantów rozwiazania problemów związanych ze star-trackerem.

Contents

Acronyms

ADCS Attitude Determination and Control System.

GPS Global Positioning System.

LEO Low Earth Orbit.

LIS Lost-In-Space.

List of Symbols

ϕ Euler angle, roll.

θ Euler angle, pitch.

ψ Euler angle, yaw.

$\boldsymbol{R}(\cdot)$ Rotation matrix using Euler angles.

$\boldsymbol{q}$ Unit quaternion.

q_0 Scalar part of unit quaternion.

$\boldsymbol{q}_{vec}$ Vector part of unit quaternion.

$\boldsymbol{Q}$ Quaternion matrix.

v General Euler angle.

$\boldsymbol{n}$ Unit vector.

$\boldsymbol{I}$ Identity matrix.

$\boldsymbol{S}(\cdot)$ Skew symmetric matrix.

$\boldsymbol{R}_n^b$ Rotation matrix representing a rotation from n to b.

$\boldsymbol{M}$ Least squares estimate of rotation matrix.

$\boldsymbol{r}$ Known directional unit vector in the NED frame.

$\boldsymbol{b}$ Known directional unit vector in the BODY frame.

1 Introduction

Stars were used for navigation already ages ago. Together with technological development, spread of sailing and seafaring people started to use more advanced tools helping to more precisely estimate position of ships on sea. In XVIII century new tool was created, named sextant. This device helped to quite precisely calculate angle between the line ship-horizon and ship-star, what let to estimate position.

Nowadays, hundreds of years later, thanks to other technologies like for example Global Positioning System (GPS) stars are not necessary any more for travelling. As it is commonly known, GPS technology is based on satellites, and today those satellites need stars for determining their attitude towards Earth.

With the miniaturization of electronics and batteries new type of satellites was invented: CubeSat micro-satellite. They have now greater sensory and processing power possibilities, previously found only in larger satellites. However CubSats are still behind in terms of attitude determination.

1.1 Motivation

The goal of this work is to make fully operational star-tracker software, that could be used on Cubesat satellites. Such program could be used on space missions and could start Polish state-of-the-art technology in growing space technology sector. There is already existing prototype of on-board computer for such CubeSat satellite, which together with this work should create full star-tracker device: hardware + software.

1.2 Outline of thesis

This thesis consists of several chapters. Here they are shortly summarized:

Chapter 1 serves as introduction to this thesis and describes the motivation and goal of this work. It also describes the background of the topic.

Chapter 2 describes all the important foundations for the fully understanding given work.

Chapter 3 is the main part of this thesis. It describes how the star-tracker program works and goes through detailed comparison of different approaches.

Chapter 4 describes the created prototype of star-tracker in Python language.

Chapter 5 talks about the implementation of star-tracker on the existing prototype of on-board computer.

Chapter 6 describes how the finished program is performing.

Chapter 7 contains conclusions about this work and created star-tracker program.

1.3 Related work

There exists a number of works connected to topic of star-tracker. Many works are cited later in this thesis as they describe important parts of algorithms. There are just a few works dealing with whole start-tracker program, not just parts of it, and nearly no works about whole program together with implementation on the real equipment. The related work can be divided into few sections:

- Works describing CubeSats in general: Swartwout [1] and Heidt et al. [2].
- Algorithms for calculating star centroids: Liebe [3], Samaan et al. [4], Knutson [5], Azizabadi et al. [6], Lindh [7] and Zhang et al. [8].
- Star identification (which star in image corresponds to which star in on-board catalogue) divided by design:
  - Planar Triangle - Cole and Crassidis [9],
  - Spherical Triangle - Cole and Crassidus [10],
  - Pyramid - Mortari et al. [11],
  - Geometric Voting - Kolomenkin et al. [12],
  - Grid Algorithm - Padgett and Kreutz-Delgado [13],
  - Brightness Independent - Dong et al. [14],

  - SP-Search - Mortari [15],
  - use of neural networks:
    * Miri and Shiri [16],
    * Lindblad et al. [17],
    * Li et al. [18],
  - separately discussed k-vector algorithm for faster search:
    * Mortari [19],
    * Mortari and Neta [20],
    * Mortari and Rogers [21].
- Attitude Determination divided by design:
  - AIM (Attitude estimation using Image Matching) - Delabie [22],
  - QUEST - Shuster [23],
  - QUEST improvement - Cheng and Shuster [24],
  - Extended Quest - Psiaki [25],
  - EQUEST - Rinnan [26],
  - Singular Value Decomposition - Juang et al. [27],
  - Optimal Image Matching - Delabie [22]
  - summarized description of attitude estimation algorithms:
    * Markley and Mortari [28],
    * Hall [29],
    * Tappe [30].
- Works on hardware for star-tracker Azizabadi et al. [6], Gąska [31], Felikson et al. [32].
- Works on full star tracker but without implementing on real device/ simulations only: Kandiyil [33], Huffman et al. [34] and Diaz [35].
- Full star-tracker with device/on real device Jalabert et al. [36], Lizy-Destrez and Mimoun [37], Rose [38], Mortari and Romoli [39] and Cannata et al. [40].

Many of those works will be mentioned and cited later in this thesis as they are crucial for this work and describe the ways how star-tracker should be designed. All those main algorithm types, like finding star centroids or star identification, are also described here in full detail in next sections.

1.4 Cubesat

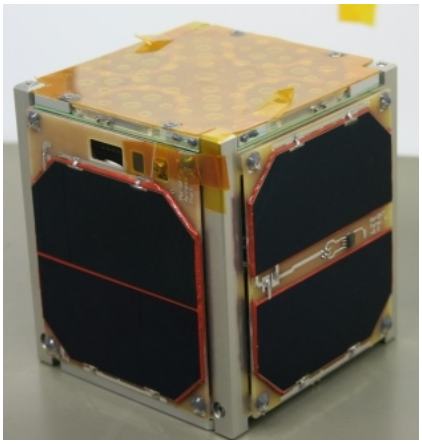
In recent years there has been the development of micro-satellites called CubeSats. CubeSat is a standard created in 1999 at the California Polytechnic State University [2], which is used for low-cost micro-satellites. CubeSats are measured in units. Most are 1U, 2U and 3U. CubeSat 1U has a size of 10 cm on each edge and the maximum weight of 1.333 kg, while the CubeSat 2U is 20x10x10 cm and can weigh up to 2.666 kg.

The advantage of the standard is primarily reduction of the price of elevation of such satellites into orbit. Their popularity is evidenced by the fact that the percentage of satellites weighing less than 10 kg has increased to about 60% of all satellites, and only CubeSats make up about half of the small satellites that were launched in last years [1][41]. Till now there were around 800 CubeSats launched [42].

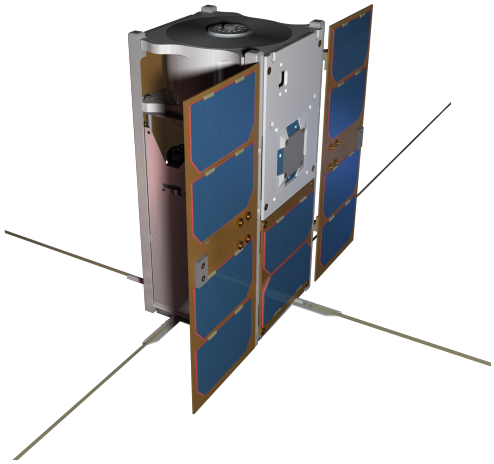
There exist wide range of CubeSat examples. Mostly they are scientific and students' satellites, for learning, experimental and science purposes. There are also few Polish examples: PW-SAT (1U) and PW-SAT 2 (2U) designed by students of Warsaw University of Technology, and Lem (8U) and Heweliusz (8U), both made at Polish Academy of Sciences. PW-SAT and PW-SAT 2 were used for learning and experimenting with new technologies, like deorbitation sail, solar sensor, etc. [43]. Scientific satellites Lem and Heweliusz were built as part of BRITE programme - joint programme of Austria, Canada and Poland. The goal of this programme is to research mechanisms of energy transportation and angular momentum which happen inside hottest stars[44].

Typical CubeSat consist of many different components. Quite often it has antennas and radios for communication with mission centre on Earth, Electric Power System (EPS - subsystem responsible for managing electric power in the satellite), Attitude Determination and Control (ADCS - subsystem responsible for analysing data from sensors and taking decision about trajectory, etc.), magnetometer, star-tracker, sun sensors, payload processor with mission instructions, etc. CubeSats are quite small, hence they usually do not have any propulsion system. Figure 2 shows the components of CubeSat on example of NASA's Interplanetary NanoSpacecraft Pathfinder In a Relevant Environment (INSPIRE).

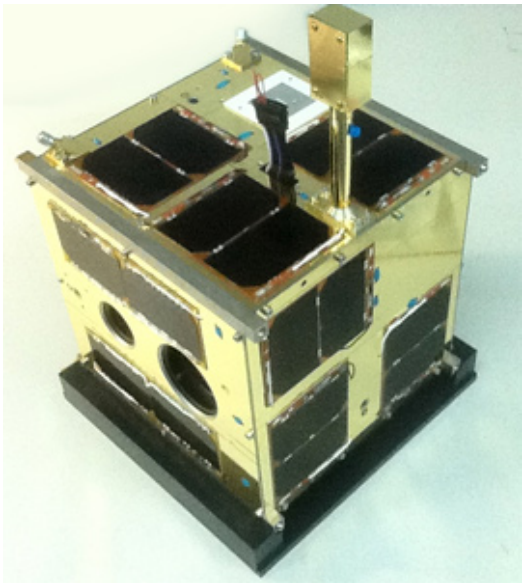
Costs of such CubeSat satellite on example of PW-SAT 2: 120,000-



(a) PW-SAT [45]



(b) PW-SAT 2 [43]



(c) BRITE-PL (Lem, Heweliusz) [46]

Figure 1: Example of Polish CubeSats.

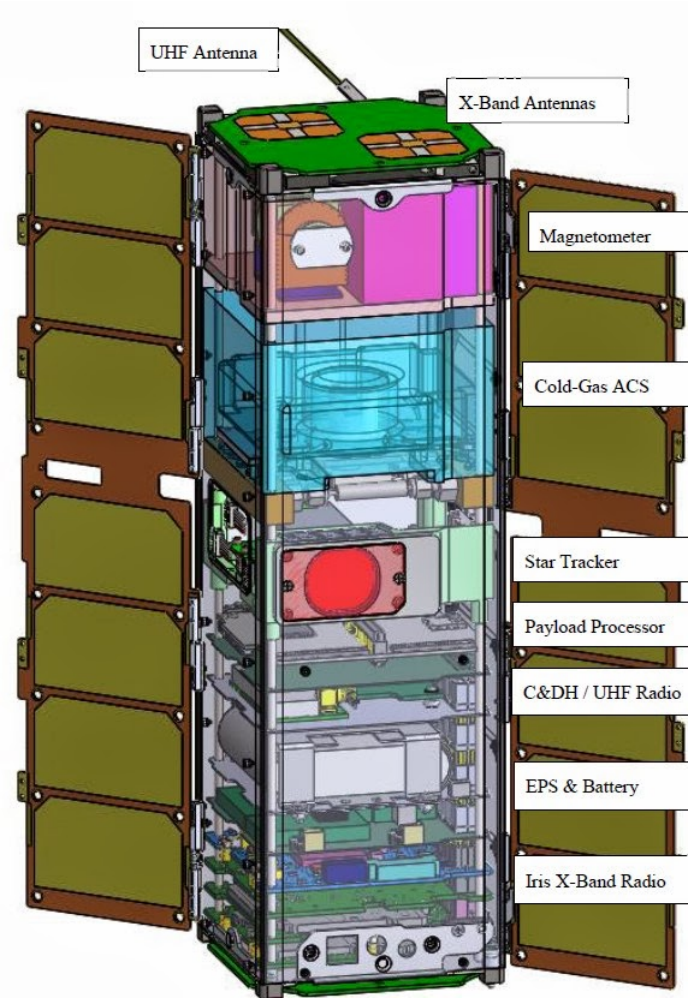


Figure 2: CubeSat build - INSPIRE, Image [47]

200,000 EUR for putting the satellite on the orbit and round 70,000 EUR for designing and building the satellite itself [48].

1.5 Means of attitude determination

Attitude of spaceships must be usually stabilised and controlled for various reasons. It is necessary for satellite antenna to be pointing towards Earth for the proper communication, to intelligently control the heat by using the effects of cooling and heating of the shadows and sunlight, as well as to navigate: manoeuvres must be performed in the right direction.

Attitude is determined between two coordinate systems (where one is reference system) and defines by what angles the coordinate system connected with the researched object has to be shifted in order to cover the reference system. Devices such as planes and satellites have so called Attitude Determination and Control System (ADCS), which controls attitude of object relative to an inertial reference frame or another entity (the celestial sphere, certain areas, the nearby objects, etc.).

Currently, attitude determination of CubeSats is limited mainly to the sun sensors, magnetometers and measurements of inertia. The following table describes the accuracy of different sensors. Unquestionable winner here is the star-tracker, which, due to its quality and uniqueness of the constellations is ideal for navigation.

1.5.1 Inertial Measurement Unit

An Inertial Measurement Unit (IMU) detects linear acceleration using one or more accelerometers and rotational rate using gyroscopes, sometimes they also include magnetometers. They are commonly used in planes, spacecrafts and many consumer devices like smart-phones. The biggest disadvantage of IMU is accumulated error. Even small error that is accumulated with time brings device to think it is in different location than it actually is. Usually the errors are corrected via use of Kalman filter, which corrects IMU errors using other sources (i.e. GPS).

1.5.2 Sun Sensors

Sun sensors typically consist of few sensors oriented perpendicularly, which can tell the angle of the sun. They can give the information how the satellite is oriented towards the sun.

1.5.3 Star-Tracker

Star-tracker is quite sophisticated device, which consists of camera, processor and memory for database. In the database is catalogue of stars, built basing on data from Earth's observatories. Simply put star-tracker, when in orbit, takes picture of space - stars. Then processes the image to recognize stars in it and compares with data in catalogue or with previous image. When stars are identified star-tracker estimates the attitude of satellite towards Earth. The main advantage over other means: great accuracy and star-tracker can find its location again after satellite was restarted or lost in space.

1.5.4 Horizon Sensors

Earth horizon sensors use infra-red radiation from the Earth's surface via use of bolometer. The device detects when infra-red signal was first detected and then lost. The time between is used to determine the Earth's width, what later can be used to determine roll angle. In other words device detects the contrast between the cold of deep space and the heat of Earth's atmosphere. Problems of this sensors are the fact that Earth is not perfectly circular and sensors detect infra-red in the atmosphere, which varies depending on the latitude.

1.5.5 Magnetometer

The device actually consists of three magnetometers - one for measuring Earth's magnetic field in each direction. Measurements can be compared with known magnetic field given by International Geomagnetic Reference Field model. The problem with this sensors is that they are susceptible to noise and outside of Earth's orbit it is completely unpredictable.

1.5.6 Global Navigation Satellite System

Nowadays nearly everyone has personal device with Global Navigation Satellite System (GNSS). They are more commonly known by their implementation by various countries: Global Positioning System (GPS) by USA, GLONASS by Russia, Galileo by European Union, and in close future BeiDou by China. It is possible to use GNSS to determine spacecraft's attitude. This is possible when at least four GNSS satellites are visible by spacecraft, what provides three-dimensional position. Even though it gives quite good accuracy, but it decreases with the altitude. GPS satellites are located on Medium Earth Orbit (MEO), around 20,200 km above sea [49]. It is possible to navigate spacecraft between Low Earth orbit (LEO - 160 to 2,000 km) and Geosynchronous Orbit (GEO - 35,786 km) [50]. For missions beyond Earth's orbit GNSS navigation is however not useful. Example of GNSS usage for navigation in space is International Space Station (ISS), which uses GPS. [51]. There are however works that claim it could make using GPS systems possible for attitude determination up to Moon attitude in the future [52].

1.5.7 Conclusions

The above means of attitude determination are usually used together with the use of Kalman filter, to compensate errors of one sensor with the other. However none of the means is more accurate than star-trackers. Sun sensors can provide very accurate measurements, but can only operate in sunlight. For low-Earth orbit (LEO), even 30% of the orbit can be done in the darkness. Magnetometers are small and can give accurate measurements if properly calibrated. Their drawback is the limited knowledge of the magnetic field and electromagnetic interference due to highly integrated construction of CubeSats. Microelectromechanical gyroscopes are small enough to fit into a CubeSats. However, they suffer from sudden movements, and could not maintain the correct measurement of the 15-minute period of the eclipse orbit LEO. GPS navigation is accurate, however only on Earth's orbit. To be truly competitive and reliable platform, CubeSats must provide the correct determination of attitude. The best way to meet this goal is via using star-trackers. The typical accuracies of different sensors are available in Table 1.

Sensor	**Accuracy** (less is better)
Inertial Measurement Unit	0.001°/hr to 1°/hr
Sun Sensors	0.005° to 3°
Star Sensors	0.0003° to 0.01° (1 arc sec to 1 arc min)
Horizon Sensors	0.05° (GEO) 0.1° (LEO)
Magnetometer	1.0° (5000km alt) 5.0° (200 km alt)

Table 1: Sensor Accuracy Ranges. Adapted from Hall [29] and Larson and Wertz [53]

1.6 On-board computer

This section describes the on-board computer (OBC) which was done as part of other thesis and is the basis for this work. Due to no existing of-the-shelf CubeSat's on-board computer with navigation module, it was designed by Gąska [31]. The OBC can work in two modes: managing CubeSat's systems with navigation module (star-tracker) or as navigation module only. Figure 3 shows the mentioned OBC, while Table 2 describes the OBC's technical parameters.

The possible tasks that can be realized by OBC are: operating on-board satellite's modules, navigation in space, auto-diagnosis, carrying out the mission.

Orientation estimation requires much more hardware resources and more power demand than other tasks executed by OBC. The OBC was designed to be able to calculate estimated attitude 2 to 5 times per second. Algorithm uses data from optic sensor and star catalogue that is placed in internal Flash memory of OBC.

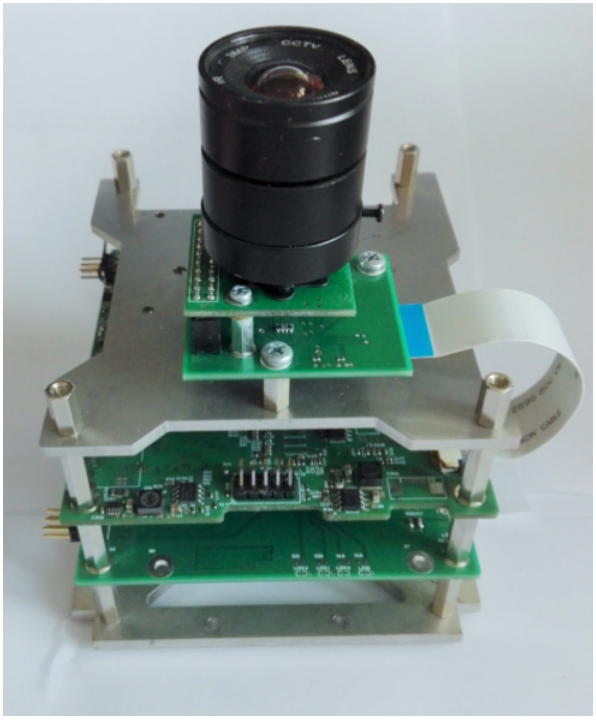


Figure 3: On-board computer designed by Gąska [31].

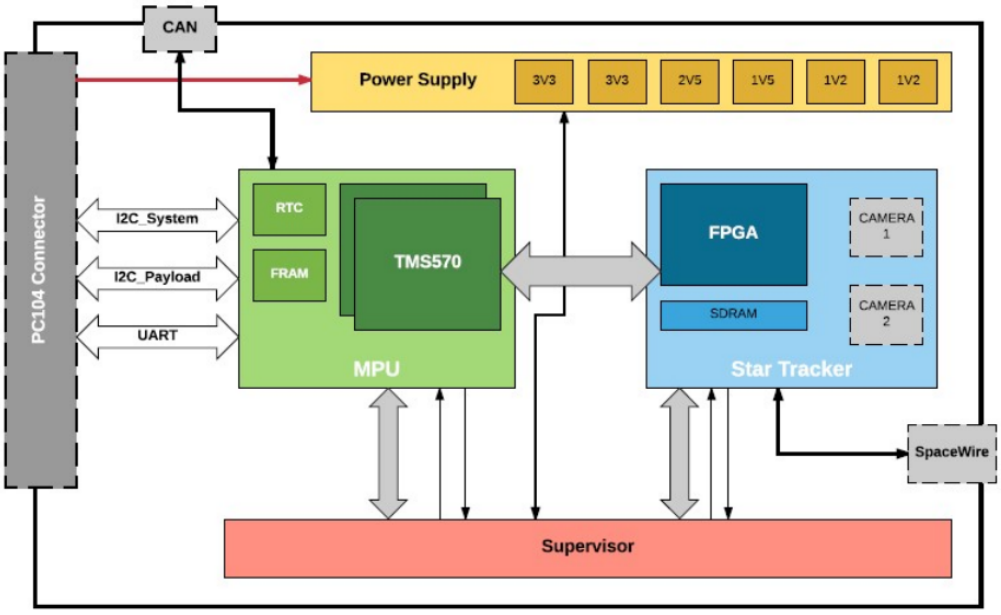


Figure 4: OBC schema[31].

1.6.1 The design of OBC

The OBC is built with:

- two redundant main micro-controllers (TMS570) which:
  - provide communication with other satellite's modules via I2C, UART and CAN interfaces,
  - read from external Flash memory the scheduled tasks and execute them,
  - obtain current time from the external RTC,
  - communicate with supervisor via UART interface,
  - store current mission status and tasks in FRAM external memory,
  - communicate with FPGA via SPI interface; they read the results of the work of built-in algorithms and can download the compressed image from the camera,
  - have the ability to change the internal configuration of the FPGA,
- programmable FPGA:
  - performs data acquisition from two cameras,
  - places the received image in the external SDRAM,
  - executes star-tracker algorithm,
  - supports SpaceWire[1] interface,
- SDRAM - memory of DDR SDRAM type for temporal storage of received image frames from both star-tracker and secondary cameras
- Supervisor - a micro-controller that acts as supervisor. Its tasks are:
  - detecting the suspension of the active main micro-controller and thus disconnecting it from the on-board computer subsystem, activating the redundant main micro-controller,
  - resetting micro-controllers,
  - transferring information about the power budget to the active micro-controller,

[1]Interface used in space devices, designed for fast data transfer between satellite's modules. The maximum transmission rate is 400 Mbps.

  - voltage measurement on power lines,
  - receiving data from current consumption measurement systems,
  - switching off the corresponding inverter in case of detecting overvoltage or power consumption excess.
  - turning on or off camera modules,
  - temperature measurement in key PCB places,
  - taking control of the I2C bus connected to the communication module to retrieve the current embedded software for the main micro-controllers; updating firmware in the main micro-controllers (TMS570),
  - communication with FPGA via UART interface,
  - reporting on the status of the on-board computer,

- Power Supply - subsystem responsible for providing a number of stable voltages powering OBC electronics, equipped in special integrated circuits responsible for measuring current on the individual power lines,
- Camera 1 - star-tracker image sensor module with the lens; the data from this camera is used by star-tracker program,
- Camera 2 - additional image sensor module for taking pictures and recording short films,
- FRAM - non-volatile ferroelectric memory that stores sensitive data of micro-controller software,
- RTC - real time clock providing the current date and time.

1.6.2 Main processing module

The main processing system is build with Texas Instrument's TMS570LS3137 with two Cortex-R4F cores in lockstep mode - both cores execute the same instructions and the results are compared; if they are different, the instructions are repeated. Both cores are operating in FreeRTOS, but can be easily moved into SafeRTOS. Table 3 describes hardware resources of this micro-controller and Figure 5 show schema of this module.

	Designed OBC module
Architecture	Two independent micro-controllers with Cortex-R4F core
Frequency	180MHz
FPGA	Cyclone III EP3C25Q240
RAM	128MB
Flash memory	16MB + 2MB
FRAM	256KB
RTC	Yes
CAN interface	1x
I2C interface	2x
SpaceWire interface	possibile to implement
Image sensor interface	2x (star-tracker camera and additional camera)
Solutions increasing resistance to radiation	Yes

Table 2: OBC's technical parameters. Adapted from Gąska [31].

Resource	Amount
Flash memory	3 MB
RAM	256kB
Frequency	160Mhz
FlexRay	1
CAN bus	3
SPI	3
I2C	1
UART	2
ADC canals	24

Table 3: Resources of Texas Instrument's TMS570LS3137. Adapted from Gąska [31].

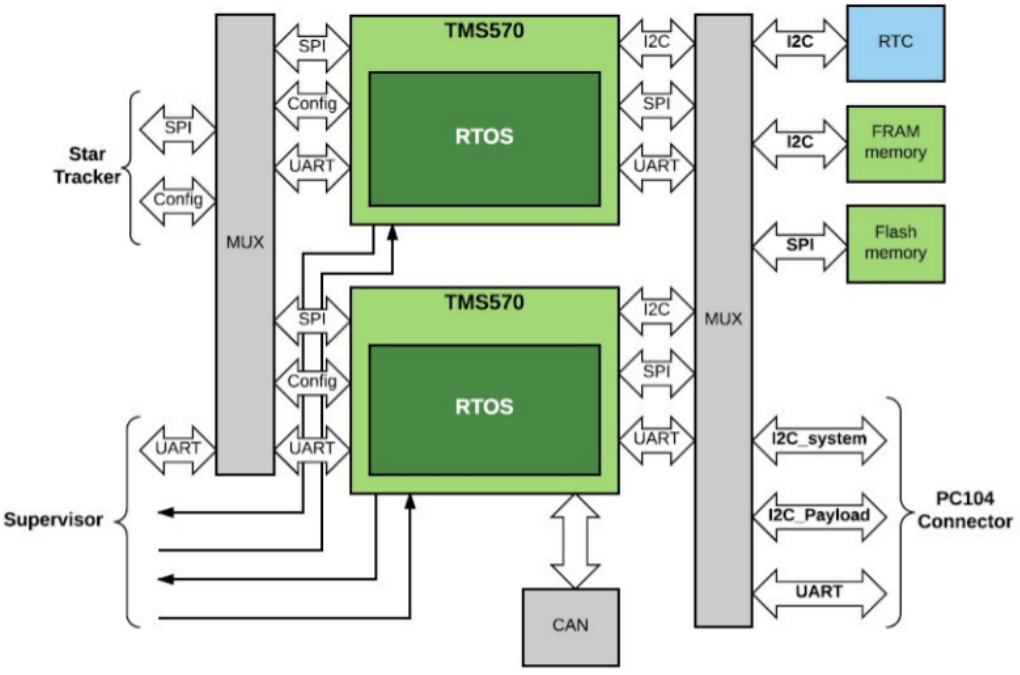


Figure 5: OBC main processing system schema[31].

1.6.3 Supervisor module

The supervisor unit is build with ATmega128 micro-controller, what is visible in Figure 6.

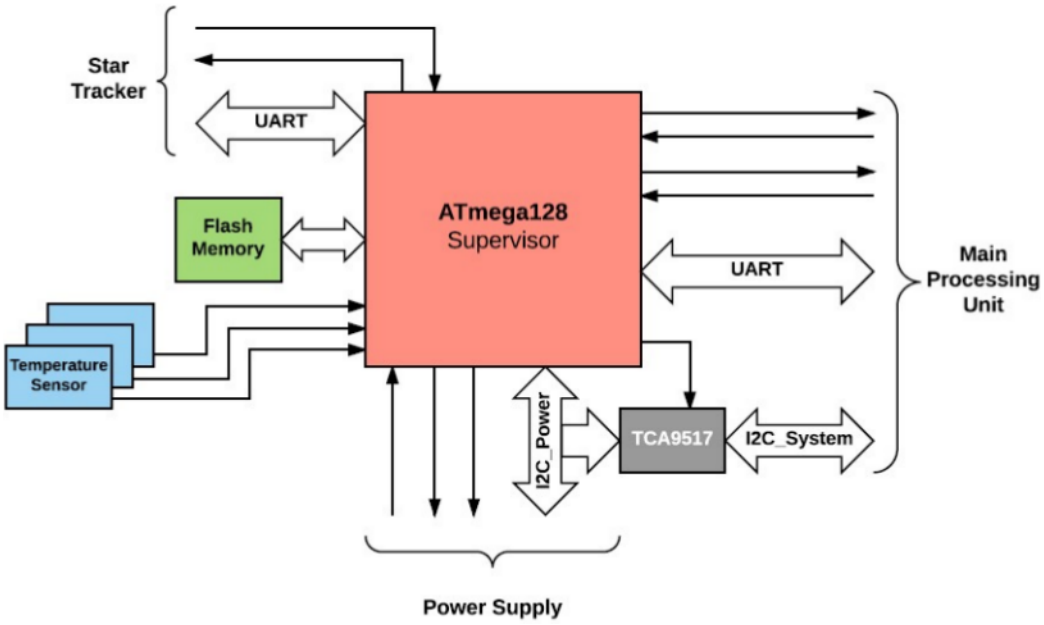


Figure 6: OBC supervisor system schema [31].

1.6.4 Star-tracker module

The FPGA used in star-tracker module is Altera's EP3C25Q240. Table 4 describes the resources of the mentioned FPGA module. In this FPGA 32-bit soft-processor NIOS II was implemented in Fast version. It can work with 175MHz frequency and achieve 195 DMIPS. The elements generated together with processor are:

- memory protection unit (MPU),
- external interrupt controller,
- 2kb cache memory,
- 4kb instruction set memory,

- address space up to 4 GB,
- 6-step pipeline,
- JTAG interface.

Resource	**Amount**
Logical elements	24624
M9K blocks	66
RAM	0.06Mbit
Hardware multiplying blocks	66
PPL	4
I/O lines	148
Differential pairs	43

Table 4: Resources of Altera's EP3C25Q240. Adapted from Gąska [31].

Another element of star-tracker module is SDRAM memory. By using two Micron's two parallel 64MB DDR SDRAM MT46V64M8 in total 128MB was achieved, where temporary image frames from cameras are stored.

The last part are cameras. Here only the star-tracker camera module is described, because the other camera is not important for this work. The star-tracker camera is based on CMOS MT9D111 sensor. It contains built-in JPEG compressor and consumes little energy. It is equipped in parallel interface with 8-bit bus for data transfer and I2C interface for configuring basic image parameters and many other sensor options, i.e. JPEG compression, coding type, etc. The Table 5 describes the most important sensor parameters.

Parameter	Value
Format	1/3.2”
Resolution	1600x1200 pixels
Max frame rate	15 fps
ADC resolution	10 bits
Output data format	YCbCr, 565RGB, 555RGB, 444RGB, JPEG 4:2:2, JPEG 4:2:0
Additional pros	Built-in JPEG compressor

Table 5: Most important parameters of MT9D111 sensor. Adapted from Gąska [31].

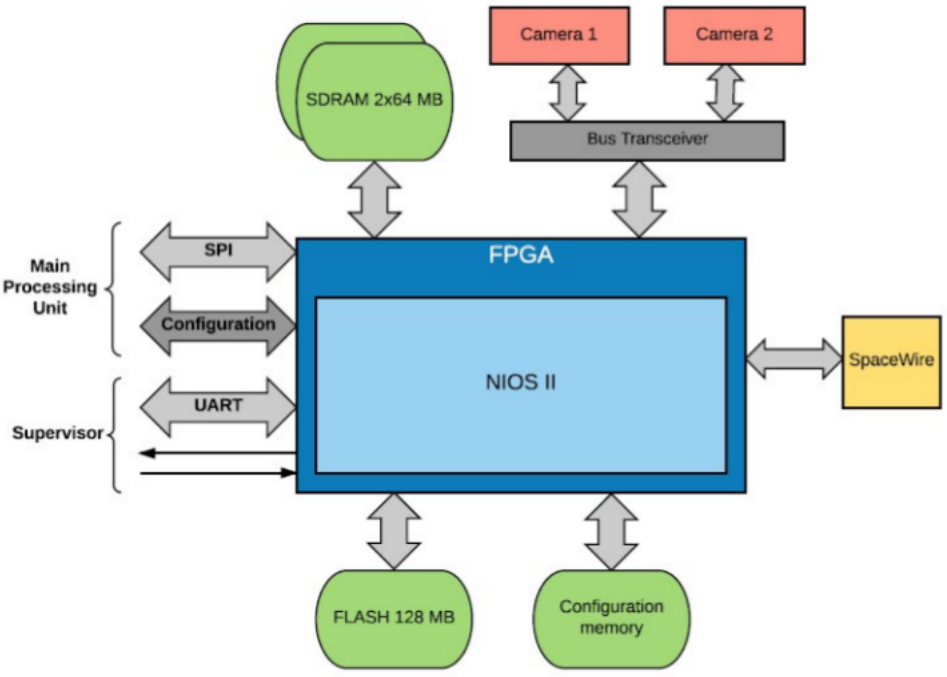


Figure 7: OBC star-tracker system schema [31].

2 Preliminaries

2.1 Earth's orbits

Most of spacecrafts' missions take place on Earth's orbits. There exist a variety of different classifications of orbits. Here will be presented only a few necessary to understand the topic.

Centric classification:

- geocentric - orbit around Earth
- heliocentric - orbit around the Sun
- Lunar orbit - orbit around the Earth's moon

Orbital period classification:

- geosynchronous - period of rotation is equal to one sidereal day (23 hours, 56 minutes, and 4 seconds) in the same direction as Earth. This results that satellite stays in the same meridian, but can move in the North-South axis. The altitude of such orbits is 35,786 km above sea level.
- geostationary - Special case of geosynchronous orbit. A geostationary orbit stays exactly above the equator. For the observer on the surface the satellite stays always at the same point in the sky. Most commercial communication, broadcast and Satellite-Based Augmentation System (SBAS - system complementing GNSS) satellites use geostationary orbits.
- semi-synchronous - orbit has an orbital period of ½ sidereal day. The example here are orbits of GPS satellites.

Altitude classification:

- Low Earth orbit (LEO): altitudes from 160 to 2,000 km. Used among others for ISS, Earth observation and spy satellites, some communication satellites like Iridium phones network, Hubble Space Telescope. Every object on altitude below 160 km will quickly loose it's altitude and it's orbit will decay.
- Medium Earth orbit (MEO): altitudes from 2,000 km to 35,786 km. Used for navigation, communication, space environment science. Most commonly used is the altitude approximately 20,00 km, because it has orbital period of ½ sidereal day. It is used for example by GPS. Other GNSS satellites' orbits are 19,000 km for GLONASS and 23,222 km for Galileo.
- Geosynchronous (GSO) and Geostationary (GEO) orbits - altitude approximately 35,786 km.
- High Earth orbit: altitude above 35,786 km. Orbital periods are longer than 1 sidereal day.

The mentioned orbits are visible to some extend in the Figure 8. Of course this is just a small variety of possible orbits, however describing more is not necessary for the purpose of this work.

2.2 Coordinate frames

Describing attitude is not that simple outside Earth's surface. Earth rotates and moves around the Sun, therefore it is necessary to understand means needed for correct attitude description. Below are described a few important frames important for understanding this work.

2.2.1 ECEF frame

Earth-Centered, Earth-Fixed frame is visible in Figure 9. Its x axis points towards intersection between Greenwich Meridian and equator, while its z-axis points along the rotation axis of the Earth. The y-axis completes a right handed orthogonal coordinate system. The origin of the frame is at the center of the Earth. This all means that the coordinates move together

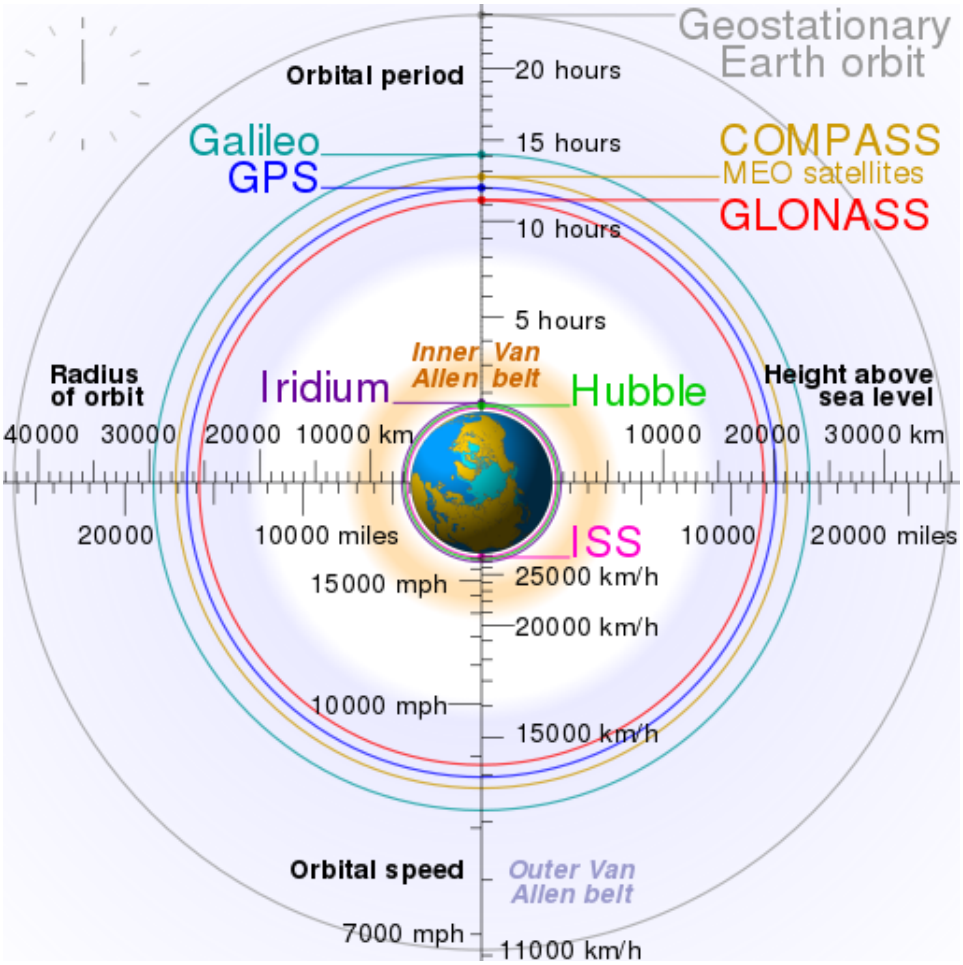


Figure 8: Earth's satellites navigation orbits [54]

with Earth's rotation. It is good frame for representing objects on Earth's surface, but not for objects in space.

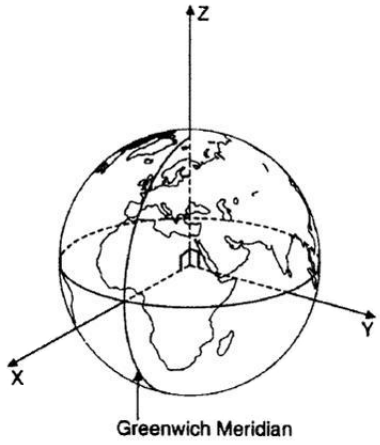


Figure 9: ECEF frame, Image from Larson and Wertz [53]

2.2.2 ECI frame

The Earth Centered Inertial frame is similar to ECEF, with the difference that it is inertial - it does not follow Earth's rotation, but stays the same despite the planet's rotation. The origin of the frame is at the center of the Earth. It is much better frame for representing objects in space. Larson and Wertz [53]

2.2.3 NED frame

The North East Down frame is represented in Figure 10. Its z-axis points downwards, perpendicular to the tangent plane of Earth. The x-axis points towards true north and the y-axis points East. The NED frame is an inertial frame - not dependent on Earth's rotation.

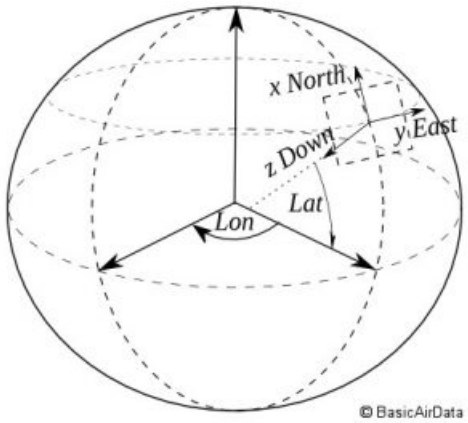


Figure 10: NED frame [55]

2.2.4 BODY frame

In this frame the origin is the centre of the spacecraft. The moves and rotates with the spacecraft. The x-axis points forward from spacecraft, the y-axis points to the right side and the z-axis points downwards. The frame is represented in Figure 11.

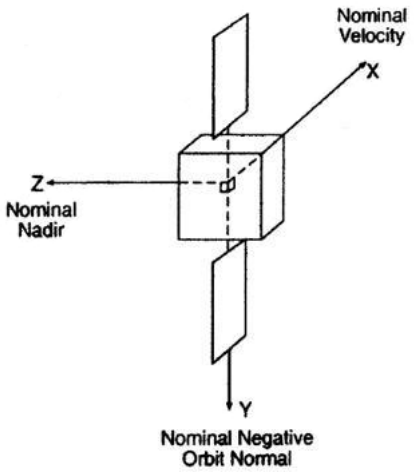


Figure 11: BODY frame, Image from Larson and Wertz [53]

2.3 Space environment?

2.4 CCD vs CMOS?

2.5 Attitude representations

Attitude is not only were one object is located relative to other, but also how it is rotated. It is possible to represent attitude in few ways. Here are described two most commonly used for such case: Euler angels and quaternions. Quaternions are used in all presented estimation methods.

2.5.1 Euler angles

Euler angles were described by Leonard Euler in 1776 [56]. They are used for representing an orientation of an object. To fully understand the orientation between two frames it is necessary to have three parameters: one angle for the rotation around each axis. Those angles are called roll, pitch, yaw typically written as ϕ, θ and ψ respectively. The Euler angles are often used for the definition of rotation matrices about the x, y and z-axis. The equations 1, 2 and 3 describe those rotation matrices.

$$\boldsymbol{R}_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{1}$$

$$\boldsymbol{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2}$$

$$\boldsymbol{R}_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

2.5.2 Quaternions

In 1843 Sir William Rowan Hamilton described quaterions Hamilton [57] and in 1845 multiplication of quaternions to describe rotations were published by Arthur Cayley Cayley [58]. Quaternions consist of four elements: three give the coordinates and fourth describing angle of rotation Courant and Hilbert [59]. A quaternion can be described as a four-dimensional vector:

$$\boldsymbol{q} \coloneqq \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{4}$$

where real part can be described using a rotation angle v, as follows:

$$q_0 = \cos(v/2) \tag{5}$$

$$\boldsymbol{n} = \frac{\boldsymbol{n}}{||\boldsymbol{n}||} \tag{6}$$

The imaginary part can be written as a vector:

$$\boldsymbol{q}_{vec} \coloneqq \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = [\boldsymbol{n} \sin(v/2)] \tag{7}$$

where $\boldsymbol{n} = \frac{\boldsymbol{n}}{||\boldsymbol{n}||}$.

2.5.3 Advantages of quaternions

Quaternions are easier to use in calculations, have more compact notation and errors easier to handle than in matrix representation. Also normalization of quaternions is easier and cheaper computationally than normalization of matrices. The most important advantage of quaternions over Euler angles is being safe from gimbal lock. Gimbal lock is the loss of one degree of freedom

that occurs when axes of two of three gimbals are driven into a parallel configuration. This could lead to real disaster Shoemake [60].

Quaternions show many advantages, and due to possible spins of CubeSat, quaternions are the only logical choice to be used for attitude estimation methods.

2.5.4 Wahba's problem

In 1965 Grace Wahba posed the problem of finding the proper orthogonal matrix A that minimizes the non-negative loss function[61]

$$L(A) = \frac{1}{2}\sum_{i}^{n} |\boldsymbol{b}_i - \boldsymbol{A}\boldsymbol{r}_i|^2 \tag{8}$$

where $\boldsymbol{b}_i$ are unit vectors in body frame, $\boldsymbol{r}_i$ are the corresponding unit vectors in a reference (NED) frame and n is the number of observations.

If the vectors are error-free and the true attitude matrix A_{true} is assumed to be the same for all the measurements, then $\boldsymbol{b}_i = A_{true}\boldsymbol{r}_i$ for each i and the loss function $L(A) = 0$ for $A = A_{true}$.

3 Star-tracker program

Star-tracker program is designed to determine the attitude of the satellite with the image of the stars made by the satellite camera. Before the start of the mission star catalogue is generated, based on the star catalogues obtained from the observatories on Earth and uploaded to the computer on-board the satellite.

After the start, the satellite is released from rocket's tank in space and has no idea where it is. Then the star-tracker on the satellite enters into Lost-In-Space mode (LIS) and analyses the current image of the stars of the camera, and then searches the corresponding result of stars in the on-board star catalogue database. If it finds an entry in the database, the satellite goes into tracking mode. This means each next calculation of the orientation of the satellite attitude is going to be based on a comparison of the current pictures of stars with the preceding ones. If it cannot find the result in the database, this action is repeated from time to time, until a match is found in the database and the program will go into tracking mode.

Of course LIS does not happen only at the beginning of the satellite's flight, but can also result from many causes, e.g. a satellite which is for a long time in the darkness may discharge its battery, and during the next entry into sunlit zone will turn on and look again its attitude. Another case is when satellite becomes lost, although once found the orientation and follow her. This happens, because the successive results are based on preceding ones and even the smallest mistake will grow until the satellite will not be able to correctly calculate their attitude. It may also happen that the satellite will rotate around its axis so fast that the program would not keep up with the processing the image and calculations. In this case, the corresponding other satellite's systems should take an action reducing his spin, but this is not part of this work. It is showed in simple conceptual diagram - Figure 12.

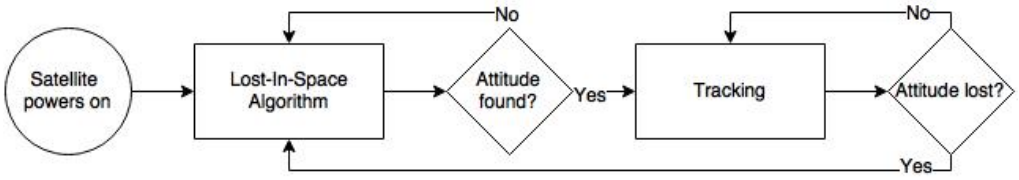


Figure 12: Startracker conceptual algorithm diagram

Each part - LIS and tracking - consists of smaller algorithms. Lost-In-

Space at the beginning of reading an image from the camera, thresholds the image. Thresholding is basically throwing away parts of the image which do not exceed the threshold brightness, making just bright enough stars taken into account. Next it calculates centroid (center of mass) of selected stars, identifies these stars using one of the possible methods (here Planar Triangle, described later) and searches the directory on-board technology (k-vector). If the database does not contain corresponding stars, the algorithm returns to the input state and starts analysing the next image. If it manages to find the corresponding stars, the program determines the attitude of the satellite (by what angle is satellite shifted comparing to the referenced object - Earth or previous image) and enters tracking mode. Tracking mode is in large part similar to the LIS. The algorithm also first analyses the photo, selects and calculates centroids stars, identifies them, but does not search for stars in the directory board. In this algorithm are compared two images, one following after the other, and on this basis the current attitude is calculated.

Generally star-tracker is divided into three main partsMcBryde and Lightsey [62]:

- recognizing stars on the image and converting the data into list of star vectors by calculating star centroids;
- identifying which star vector represents which real star in catalogue. This is done by comparing star vectors from the image with data in star catalogue, which is generated before space mission;
- estimating the attitude by calculating the displacement between two frames.

3.1 Centroid - star recognition

The whole star-tracker program is based on very precise calculations. For this reason, the calculation based on the position of the pixels only can give incorrect results. It is necessary to calculate the position of stars with an accuracy exceeding pixels. This is why the calculation of the star centroids is necessary.

The first step is the determination of stars' position in the plane of the image. If focused star pictures are recorded, the image of each star will fall

only on one or two pixels, and most likely the saturate these pixels, resulting in a pixel level accuracy. Many star-trackers are doing deliberately blurred images, in order to spread the photons on a larger number of pixels, which allows the algorithm for calculating the centroid in sub-pixel accuracy.

After the registration of such image, the centroid of the star is found similarly to the centroid weight points array, with a few differences. Firstly, instead of weight light intensity is used. Secondly, the intensity of light is usually normalized by the pixels around the star in order to filter out glare or noise. The resulting outcome is a series of two-dimensional coordinates on the photo plane with the starting point at the center of the image. This allows the system to the coordinates of the stars can be easily converted into unit vectors in the next step.

The algorithm requires specification of the light intensity threshold (to select the brightest stars) I_{thresh} and the size of the Region of Interest (ROI) a_{ROI} in pixels. These values are adjusted to manipulate the performance of the algorithm. For example, the higher the value of I_{thresh} is more resistant to noise, but can miss some real stars in the picture. Similarly, a large value a_{ROI} means a more exact value of centroid, but the algorithm can see one star there, where are actually two within a short distance of each other. The centroiding part is about trade-off between noise resistance and star recognition, and also between recognizing few close stars as one and recognizing one big star as a few. Please note that a_{ROI} must be a positive odd number for the proper functioning of the algorithm.

The idea of how to calculate such centroids is adapted from McBryde and Lightsey [62] and described below:

1. For each pixel at image coordinates (x,y) with light intensity $I(x, y) > I_{thresh}$, the ROI is defined as the square of pixels with side length a_{ROI} and bottom-left corner at (x_{start}, y_{start})

$$x_{start} = x - \frac{a_{ROI} - 1}{2} \tag{9}$$

$$y_{start} = y - \frac{a_{ROI} - 1}{2} \tag{10}$$

$$x_{end} = x_{start} + a_{ROI} \tag{11}$$

$$y_{end} = y_{start} + a_{ROI} \tag{12}$$

2. If $x_{start} < 0$ or $y_{start} < 0$, discard the pixel and return to previous step with the next pixel.

3. Find the average intensity value of the border pixels I_{border}

$$I_{bottom} = \sum_{i=1}^{x_{end}-1} I(i, y_{start}) \tag{13a}$$

$$I_{top} = \sum_{i=2}^{x_{end}} I(i, y_{end}) \tag{13b}$$

$$I_{left} = \sum_{j=1}^{y_{end}-1} I(x_{start}, j) \tag{13c}$$

$$I_{right} = \sum_{j=2}^{y_{end}} I(x_{start}, j) \tag{13d}$$

$$I_{border} = \frac{I_{top} + I_{bottom} + I_{left} + I_{right}}{4(a_{ROI} - 1)} \tag{13e}$$

4. Normalize all the non-boarder pixels, yielding normalized light intensity matrix $\tilde{I}$

$$\tilde{I}(x, y) = I(x, y) - I_{border} \tag{14}$$

5. Calculate centroid locations (x_{CM}, y_{CM}) and brightness B

$$B = \sum_{i=x_{start}+1}^{x_{end}-1} \sum_{j=y_{start}+1}^{y_{end}-1} \tilde{I}(i, j) \tag{15}$$

$$x_{CM} = \sum_{i=x_{start}+1}^{x_{end}-1} \sum_{j=y_{start}+1}^{y_{end}-1} \frac{i \times \tilde{I}(i, j)}{B} \tag{16}$$

$$x_{CM} = \sum_{i=x_{start}+1}^{x_{end}-1} \sum_{j=y_{start}+1}^{y_{end}-1} \frac{j \times \tilde{I}(i, j)}{B} \tag{17}$$

6. When the centroid location has been calculated for each pixel above the intensity threshold, it is still necessary to make clustering of them. This is done via averaging together all values that are clustered together. Before clustering it is sure that some stars will have more centroids than one, and clustering helps merge them into one. It is configurable by the number of proximity pixels of the centroids. Of course it is

not possible to do it perfectly. The main problem is that sometimes there are few stars in close proximity to each other on the photo and it is possible to count those few stars as one, while doing this correctly for all other stars in the photo. On the other hand, if threshold is lowered too much, it is possible to count few stars correctly, but also possible to count one star as a few. The Figure 13 shows example of such trade-off. The clustering can be accomplished by checking each new centroid against a list of already processed centroids. If the new location is within the given range (for example 5 pixels) it is assumed they represent the same star and therefore they values are averaged. The output of this step is the list of star centroids, where each centroid should represent separate star.

7. Now convert each found centroid location into unit vector $\boldsymbol{u}$ using the camera pixel size μ and camera focal length f

$$\boldsymbol{u} = \frac{\begin{bmatrix} \mu x_{CM} & \mu y_{CM} & f \end{bmatrix}^T}{|| \begin{bmatrix} \mu x_{CM} & \mu y_{CM} & f \end{bmatrix} ||} \tag{18}$$

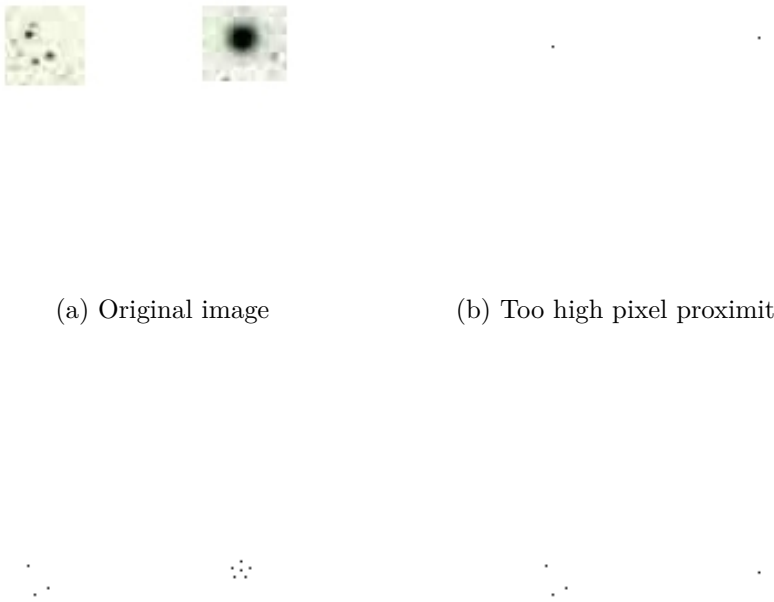
3.2 Star identification

Star identification is the part of star-tracker program, which actually finds out which recognized stars are which. Prior this part it is only known that some stars were read from the image, and now it is the time to identify them by comparing to on-board star catalogue. There is a number of techniques to do that[63]. In this chapter most of them will be described.

3.2.1 Angle Matching

The Angle Matching is probably the oldest way for identifying stars. It needs at least two stars as an input[64]. The Figure 14 visualizes the way the method works.

The angle between two vectors pointing to the stars is given by:



(a) Original image

(b) Too high pixel proximity threshold

(c) Too low pixel proximity threshold

(d) Ideal result of clustering

Figure 13: Example of centroids' clustering trade-off (in negative colours).

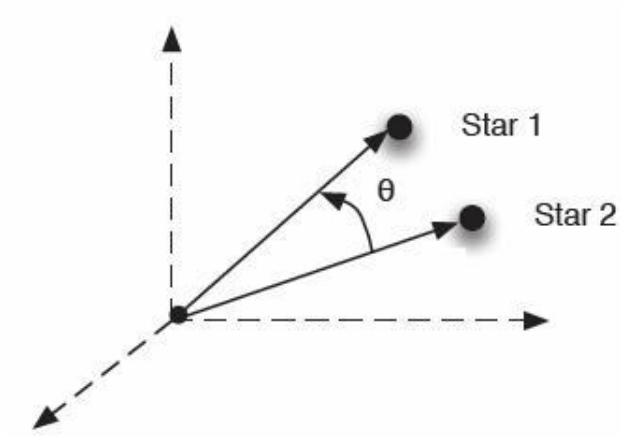


Figure 14: Vector angle method, Image Gottlieb [64]

$$\theta = \cos^{-1}(\boldsymbol{r}_1 \cdot \boldsymbol{r}_2) \tag{19}$$

where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are unit vectors pointing to each star. The vectors are given in inertial space. The angle θ is the same when inertial vectors or body vectors are used. The problem is that the angle measured between stars in the image contain significant measurement error, that cannot be ignored. What is needed by this technique, is the standard deviation of the angle between two stars when each star measurement possesses the error described. It can be used to provide a bound on the expected errors. Firstly standard coordinate transformation is necessary:

$$\boldsymbol{b}_i = A\boldsymbol{r}_i \tag{20}$$

where $\boldsymbol{r}_i$ is direction the direction of star in the ECI coordinate system, A is the direction cosine matrix, $\boldsymbol{b}_i$ is direction of star in the star-tracker body coordinate system.

Nearly all the probability of errors is focused on a very small area about the direction of $A\boldsymbol{r}_i$, that the sphere containing that point can be approximated by a tangent plane, characterized by:

$$\tilde{\boldsymbol{b}}_i = A\boldsymbol{r}_i + \boldsymbol{v}_i, \quad \boldsymbol{v}_i^T A\boldsymbol{r}_i = 0 \tag{21}$$

where $\tilde{\boldsymbol{b}}_i$ is the itith measurement and sensor $\boldsymbol{v}_i$ is approximately Gaussian, which satisfies

$$E\{\boldsymbol{v}_i\} = 0 \tag{22a}$$

$$E\left\{\boldsymbol{v}_i\boldsymbol{v}_i^T\right\} = \sigma_i^2[\boldsymbol{I} - (A\boldsymbol{r}_i)(A\boldsymbol{r}_i)^T] \tag{22b}$$

where σ_i^2 is the variance and E is expectation. The dot product of two body observations gives

$$\boldsymbol{b}_1^T \boldsymbol{b}_2 = \boldsymbol{r}_1^T A^T A \boldsymbol{r}_2 = \boldsymbol{r}_1^T \boldsymbol{r}_2 \tag{23}$$

what show that the dot product is attitude-invariant measurement. In case of two measurements $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ with noise, where $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are uncorrelated:

$$\tilde{\boldsymbol{b}}_1 = A\boldsymbol{r}_1 + \boldsymbol{v}_1$$
$$\tilde{\boldsymbol{b}}_2 = A\boldsymbol{r}_2 + \boldsymbol{v}_2$$

define the effective measurement:

$$z \equiv \tilde{\boldsymbol{b}}_1^T \tilde{\boldsymbol{b}}_2 = \boldsymbol{r}_1^T \boldsymbol{r}_2 + \boldsymbol{r}_1^T A^T \boldsymbol{v}_J + \boldsymbol{r}_2^T A^T \boldsymbol{v}_1 + \boldsymbol{v}_1^T \boldsymbol{v}_2 \tag{25}$$

As $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are uncorrelated, then:

$$E\{z\} = \boldsymbol{r}_1^T \boldsymbol{r}_2 \tag{26}$$

Define the following variable:

$$p \equiv z - E\{z\} = \boldsymbol{r}_1^T A^T \boldsymbol{v}_J + \boldsymbol{r}_2^T A^T \boldsymbol{v}_1 + \boldsymbol{v}_1^T \boldsymbol{v}_1 \tag{27}$$

Taking $E\{p^2\}$ gives the variance for the dot product measurement error:

$$\begin{aligned} \sigma_p^2 \equiv E\left\{p^2\right\} = \\ \boldsymbol{r}_1^T A^T R_2 A \boldsymbol{r}_1 + \boldsymbol{r}_2^T A^T R_a A \boldsymbol{r}_2 + Trace(R_1 R_2) = \\ Trace(A\boldsymbol{r}_1 \boldsymbol{r}_1^T R_2) + Trace(A\boldsymbol{r}_2 \boldsymbol{r}_2^T R_1) + Trace(R_1 R_2) \end{aligned} \tag{28}$$

where $E\left\{\boldsymbol{v}_1 \boldsymbol{v}_1^T\right\} \equiv R_1$ and $E\left\{\boldsymbol{v}_2 \boldsymbol{v}_2^T\right\} \equiv R_2$.

The effective measurement noise in the dot product consist of both Gaussian and chi-squared distribution, but since the noise is small in comparison to the unit vector observation, the effective noise can be treated as purely Gaussian. As the attitude is not known, the $A\boldsymbol{r}_1$ and $A\boldsymbol{r}_2$ are replaced by $\tilde{\boldsymbol{b}}_1$ and $\tilde{\boldsymbol{b}}_2$. Errors introduced in this substitution are second-order in nature, hence the variance is completely independent of the attribute matrix A.

In case there exist more than one possible solution, so called pivoting is made. This means that if for first angle more than one solution was found, the second angle is selected, that shares one common star with the first angle. When all solutions for second angle are obtained, the solutions not common

for those two angles are discarded. If there is still more than one solution, then another pivot is made. Now it compares the second angle with third angle, with a common star. The rest goes analogically. This happens till finally there is only one solution for some two angles, or there are no more angles left in the input, and system enters into LIS mode.

3.2.2 Spherical Triangle Matching

Instead of measuring the inter-star angle, this method creates spherical triangles out of three stars. Due to that more information can be obtained from such triangle than an angle, what enables to identify stars faster and using fewer stars overall than in previous method. In this technique the area and polar moment are used instead of angles. The main disadvantage comparing to angle matching method is that this one requires not two but three stars in order to work properly, however under typical conditions in angle matching it is necessary to use more stars due to measurement error. Also the spherical triangle catalogue is about 10 times larger than angle catalogue[10].

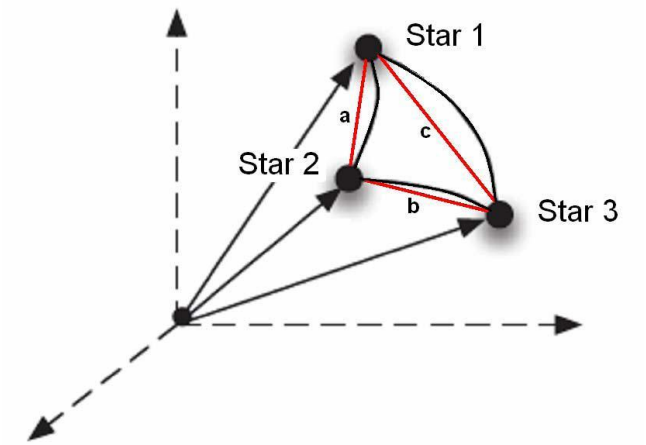


Figure 15: Spherical Triangle Method, Image Cole and Crassidus [10]

Given three unit vectors pointing toward three stars, the area of a spherical triangle is as follows:

$$A = 4\tan^{-1}\sqrt{\tan\frac{s}{2}\tan\frac{s-a}{2}\tan\frac{s-b}{2}\tan\frac{s-c}{2}} \tag{29}$$

where

$$s = \frac{1}{2}(a+b+c)$$
$$a = \cos^{-1}\left(\frac{\boldsymbol{b}_1 \cdot \boldsymbol{b}_2}{|\boldsymbol{b}_1||\boldsymbol{b}_2|}\right)$$
$$b = \cos^{-1}\left(\frac{\boldsymbol{b}_2 \cdot \boldsymbol{b}_3}{|\boldsymbol{b}_2||\boldsymbol{b}_3|}\right)$$
$$c = \cos^{-1}\left(\frac{\boldsymbol{b}_3 \cdot \boldsymbol{b}_1}{|\boldsymbol{b}_3||\boldsymbol{b}_1|}\right)$$

The equation is given for the body frame, but can be used in the ECI frame too. To get a bound for the measurement error, the standard deviation must be calculated.

The polar moment is good supplement to area, because it is possible for two spherical triangles to have the same area but different polar moment, and all way around. This helps to quickly reduce the number of possible solutions.

The polar moment can be obtained as follows

$$I_p = \sum \theta^2 dA \tag{31}$$

The polar moment of each spherical triangle is calculated by recursive algorithm, which divides the triangle into smaller triangles till the recursion's depth is met.

The standard deviations for area and polar moment are given in Cole and Crassidus [10].

In this technique pivoting works analogically to the angle matching method. Note that while calculation of the standard deviation of area is quite straightforward, the standard deviation of the second moment calculation is difficult to compute analytically.

3.2.3 Planar Triangle

The third technique, quite similar to previous one is Planar Triangle. Instead of spheres it uses planes, what makes it much less computationally demand-

ing. It also uses the same two properties: triangle area and polar moment. In this case also three stars are necessary[9].

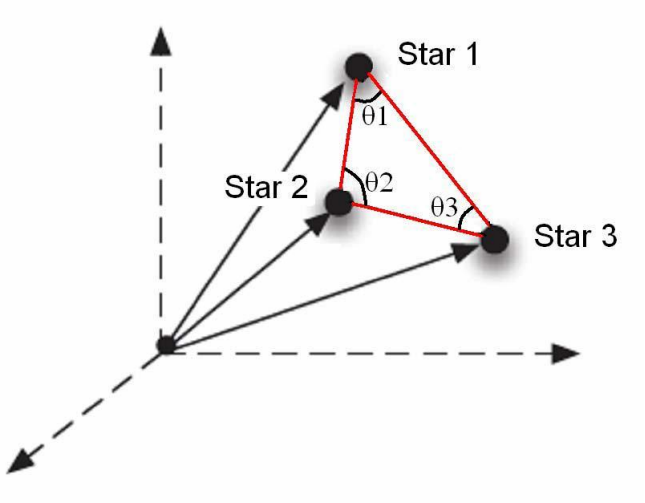


Figure 16: Planar Triangle Method, Image Cole and Crassidis [9]

Given thee unit vectors pointing toward three stars $\boldsymbol{u_p}$, $\boldsymbol{u_q}$, $\boldsymbol{u_r}$, the area for planar triangle is as follows:

$$A = \sqrt{s(s-a)(s-b)(s-c)} \tag{32}$$

where

$$s = \frac{1}{2}(a+b+c) \tag{33a}$$

$$a = ||\boldsymbol{u_p} - \boldsymbol{u_q}|| \tag{33b}$$

$$b = ||\boldsymbol{u_q} - \boldsymbol{u_r}|| \tag{33c}$$

$$c = ||\boldsymbol{u_p} - \boldsymbol{u_r}|| \tag{33d}$$

The equation is given for the body frame, similarly as for the spherical triangle. It can be also used for ECI frame. Also here the standard deviation has to be calculated in order to obtain a bound of the measurement error.

Similarly as in the previous method, here also polar moment is calculated:

$$J = A\frac{(a^2+b^2+c^2)}{36} \tag{34}$$

Since the planar area is a nonlinear function, a linearization must be used to determine its variance. To compute this, the following matrix must be evaluated:

Derivatives

$$H = \begin{bmatrix} \boldsymbol{h}_1^T & \boldsymbol{h}_2^T & \boldsymbol{h}_3^T \end{bmatrix} \tag{35}$$

where

$$\boldsymbol{h}_1^T \equiv \frac{\delta A}{\delta a}\frac{\delta a}{\delta \boldsymbol{b}_1} + \frac{\delta A}{\delta c}\frac{\delta c}{\delta \boldsymbol{b}_1} \tag{36a}$$

$$\boldsymbol{h}_2^T \equiv \frac{\delta A}{\delta a}\frac{\delta a}{\delta \boldsymbol{b}_2} + \frac{\delta A}{\delta b}\frac{\delta b}{\delta \boldsymbol{b}_2} \tag{36b}$$

$$\boldsymbol{h}_3^T \equiv \frac{\delta A}{\delta b}\frac{\delta b}{\delta \boldsymbol{b}_3} + \frac{\delta A}{\delta c}\frac{\delta c}{\delta \boldsymbol{b}_3} \tag{36c}$$

To calculate the partials with respect to a, b and c serve the following equations:

$$\frac{\delta A}{\delta a} = \frac{u_1 - u_2 + u_3 + u_4}{4A} \tag{37a}$$

$$\frac{\delta A}{\delta b} = \frac{u_1 + u_2 - u_3 + u_4}{4A} \tag{37b}$$

$$\frac{\delta A}{\delta c} = \frac{u_1 + u_2 + u_3 - u_4}{4A} \tag{37c}$$

where

$$u_1 = (s-a)(s-b)(s-c) \tag{38a}$$

$$u_2 = s(s-b)(s-c) \tag{38b}$$

$$u_3 = s(s-a)(s-c) \tag{38c}$$

$$u_4 = s(s-a)(s-b) \tag{38d}$$

The partials with respect to $\boldsymbol{b}_1$, $\boldsymbol{b}_2$ and $\boldsymbol{b}_3$ are calculated formulas

$$\frac{\delta a}{\delta \boldsymbol{b}_1} = (\boldsymbol{b}_1 - \boldsymbol{b}_2)^T/a, \quad \frac{\delta a}{\delta \boldsymbol{b}_2} = -\frac{\delta a}{\delta \boldsymbol{b}_1} \tag{39a}$$

$$\frac{\delta b}{\delta \boldsymbol{b}_2} = (\boldsymbol{b}_2 - \boldsymbol{b}_3)^T/b, \quad \frac{\delta b}{\delta \boldsymbol{b}_3} = -\frac{\delta b}{\delta \boldsymbol{b}_2} \tag{39b}$$

$$\frac{\delta c}{\delta \boldsymbol{b}_1} = (\boldsymbol{b}_1 - \boldsymbol{b}_3)^T/c, \quad \frac{\delta c}{\delta \boldsymbol{b}_3} = -\frac{\delta c}{\delta \boldsymbol{b}_1} \tag{39c}$$

The variance of the area is as follows

$$\sigma_A^2 = HRH^T \tag{40}$$

where

$$R \equiv \begin{bmatrix} R_1 & 0_{3x3} & 0_{3x3} \\ 0_{3x3} & R_2 & 0_{3x3} \\ 0_{3x3} & 0_{3x3} & R_3 \end{bmatrix} \tag{41}$$

As with the area, it is necessary to obtain the variance of the polar moment. The following partial matrix has to be evaluated first:

$$\bar{H} = \begin{bmatrix} \bar{\boldsymbol{h}}_1^T & \bar{\boldsymbol{h}}_2^T & \bar{\boldsymbol{h}}_3^T \end{bmatrix} \tag{42}$$

where

$$\bar{\boldsymbol{h}}_1^T \equiv \frac{\delta J}{\delta a}\frac{\delta a}{\delta \boldsymbol{b}_1} + \frac{\delta J}{\delta c}\frac{\delta c}{\delta \boldsymbol{b}_1} + \frac{\delta J}{\delta A}\boldsymbol{h}_1^T \tag{43a}$$

$$\bar{\boldsymbol{h}}_2^T \equiv \frac{\delta J}{\delta a}\frac{\delta a}{\delta \boldsymbol{b}_2} + \frac{\delta J}{\delta b}\frac{\delta b}{\delta \boldsymbol{b}_2} + \frac{\delta J}{\delta A}\boldsymbol{h}_2^T \tag{43b}$$

$$\bar{\boldsymbol{h}}_3^T \equiv \frac{\delta J}{\delta b}\frac{\delta b}{\delta \boldsymbol{b}_3} + \frac{\delta J}{\delta c}\frac{\delta c}{\delta \boldsymbol{b}_3} + \frac{\delta J}{\delta A}\boldsymbol{h}_3^T \tag{43c}$$

with

$$\frac{\delta J}{\delta a} = Aa/18, \quad \frac{\delta J}{\delta a} = Ab/18, \quad \frac{\delta J}{\delta a} = Ac/18 \tag{44a}$$

$$\frac{\delta J}{\delta A} = (a^2 + b^2 + c^2)/36 \tag{44b}$$

Other quantities are given from the area variance calculations. The variance of the polar moment is calculated with the following formula:

$$\sigma_J^2 = \bar{H} R \bar{H}^T \tag{45}$$

Pivoting works analogically as in the previous methods.

This method gives similar results as spherical triangle method, but planar triangle method does not have to perform recursive calculations of polar moment like the previous method, hence it has lower computational demand for star-identification[65].

3.2.4 Pyramid

Pyramid method is described by Mortari et al. [11]. This algorithm is designed to be used efficiently with 4-star input, however can also function for 3-star input if no more are available. It uses set of star pairs in elementary measured star polygons (3 for a triangle, six for a 4-star pyramid). The vertices between adjacent measured star pairs share a common catalogued star.

The Figure 17 shows how the algorithm works. If there are less than 3 stars on input, the algorithm returns error, what actually brings back to start of the algorithm with new input data. In case there are three stars only, algorithm gets rid of redundant solutions, and if the solution is unique, the stars are identified and solution is outputted. In case of not finding unique solution, algorithm returns an error, and starts again with new data. If there are more than three stars, algorithm gets three stars for a triangle, and then one reference star. Later it checks pairs of reference star and triangle's vertices. If unique solution is found, it identifies the remaining stars and outputs all the identified stars. If no unique solution has been found, algorithm tries to get new triangle and repeat the steps. There are two possible exits of this loop: the already mentioned finding unique solution, or running out of new triangles. In the last case algorithm removes redundant solutions, then check again if it is unique solution. Finally either unique solution is found and algorithm identifies all other stars from the input and outputs them, or returns an error and stars again with the new input data.

According to authors the algorithm can solve LIS problem and is highly robust. The key to algorithm's efficiency lies in the usage of k-vector method and the usage of expected random frequencies connected to matching inter-star angles from measured star polyhedron. The algorithm has been tested on Draper's "Inertial Stellar Compass" star tracker[66] and on MIT's satellites HETE and HETE-2[67] successfully. This algorithm is currently under exclusive contract to StarVision Technologies[68][69].

3.2.5 Voting

Algorithm was presented by Kolomenkin et al. [12]. Pair of stars in the catalogue votes for a pair of stars in the image if angular distance between

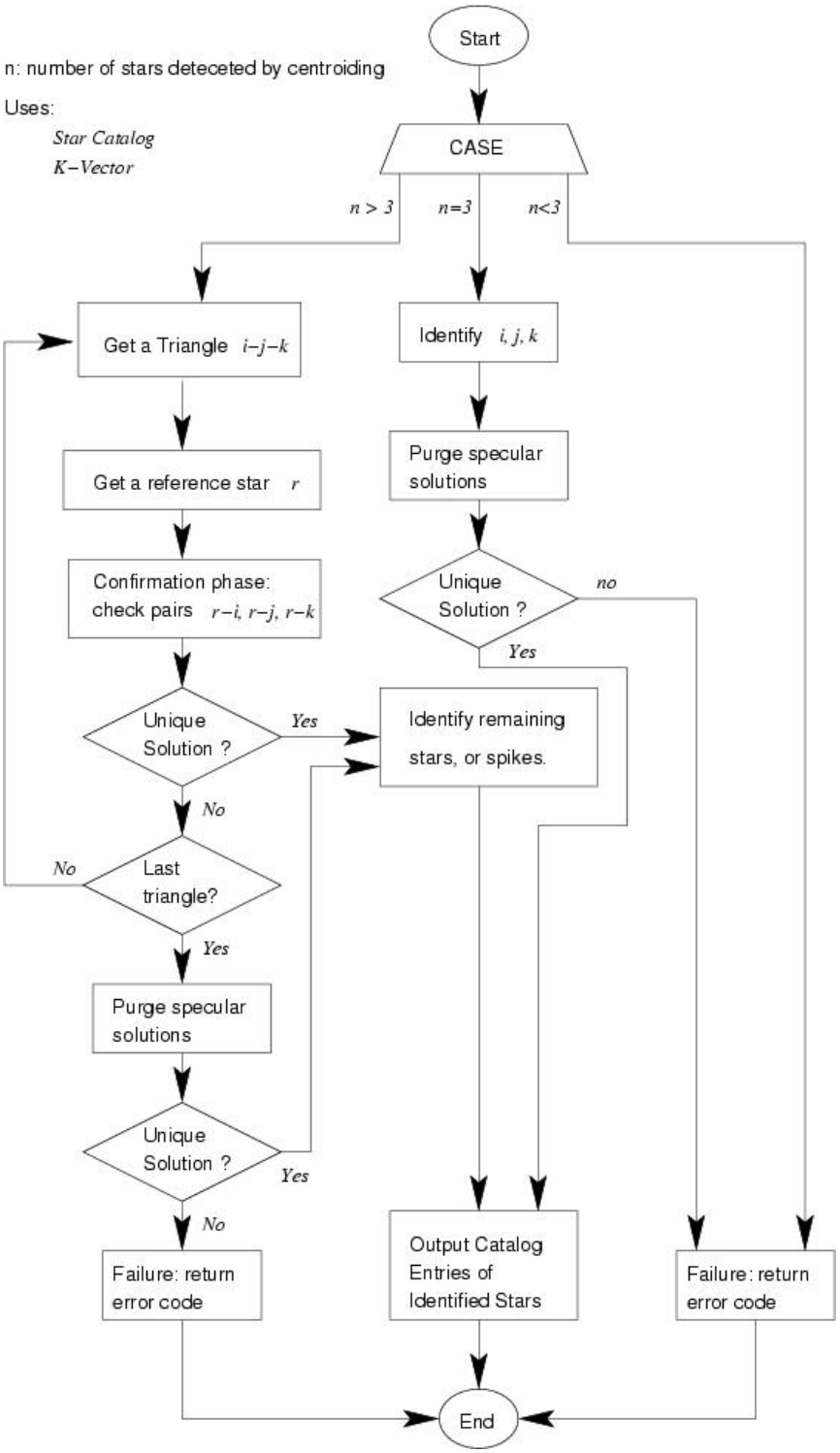


Figure 17: Pyramid Method Flowchart, Image Mortari et al. [11]

the stars of both pairs is similar. Since angular distance is a symmetric relationship, each catalogue star in the pair vote for each in the image pair. Then the image star is identified as the catalogue star which gave the most votes.

Authors' testing has shown that the algorithm is as accurate, as fast and more robust than other methods. The authors tested the algorithm in simulation only, and it does not appear to be tested in space yet. Furthermore, all simulations and testing were done on assumption of slow rotation, hence it is not known how it would function in the real environment. The more details about this algorithm can be found in Kolomenkin et al. [12].

3.2.6 Grid

The Grid Algorithm was designed by Padgett and Kreutz-Delgado [13]. The main ideas of the algorithm are to generate a specified pattern database and to construct a grid pattern from the measured star field in the CCD plane. The specified pattern is designed by using grids. Then the measured grid pattern is identified by best matching pattern in the database.According to authors the algorithm is robust and comparable with other methods in terms of accuracy and performance.

This algorithm will not be described in more detail, since it is designed for CCD camera, and this thesis focuses on star-tracker for OBC mentioned before, which uses CMOS camera. More information can be found in Padgett and Kreutz-Delgado [13].

3.2.7 Other techniques

Other techniques involve:

- Spherical-Polygon (SP) Search by Mortari [15]:

  The idea of this method is that any star direction can be expressed as a linear combination of two non-parallel star directions (star pair basis) together with their vector cross-product. The star identification problem is transformed into the simpler problem of finding which stars,

within a large star catalogue, are admissible with a given direction. This is solved by approximating a cone with a spherical polygon and search for all of the three components of the given direction.

- Star Neighbourhood Approach by Samaan et al. [70]:

  This method does not solve LIS problem, but can be used only in star-tracker's tracking mode. The algorithm makes use of the knowledge of the previous frame data to initiate the current frame star.

- Brightness Independent 4-Star Matching Algorithm by Dong et al. [14]:

  This technique needs at least 4 stars as an input, than calculates angular separations between each pair, and tries to find corresponding star pairs with similar angular separations in the database. Identified stars are not used again, and the remaining stars are used with additional stars to make a new 4-star group. Then this step is repeated. In case no star was identified, at least one star will be removed to build a new 4-star group. If there would be no more than 4 stars left, then already identified stars have to be used. No information about real-life usage has been found.

- Use of neural networks and machine learning algorithms:

  - 'Star Identification using Neural Networks' by Lindblad et al. [17] describes approach to star identification in two different ways: Radial Basis Funcion (RBF) Neural Network and k Nearest Neighbours,
  - Parallel Backpropagation (BP) multi-subnets are designed by Li et al. [18],
  - Method based on Delunay Triangulation algorithm and neural networks is designed by Miri and Shiri [16].

3.2.8 Conclusions

From the described techniques the Planar Triangle method was chosen. Comparing to Angle Method it gives better results and comparing to Spherical Triangle it less computationally demanding[30][65]. It requires more memory for the database, but it is still in range of the designed OBC. Pyramid technique cannot be used as it is patented. Algorithms based on neural networks need significantly more computational power and memory. What is more, no

information about the usage of such algorithms on real spacecraft was found. As for other algorithms, Star Neighbourhood does not support LIS mode, for Brightness Independent, SP-Search, Voting and Grid methods no information about real-life use was found. Moreover Planar Triangle was tested in at least two other works (but only against Angle and Spherical Triangle Methods), and was chosen as giving the best overall results, and in one work it was chosen for implementation along one other algorithm[34] due to its speed.

3.3 Star-catalogue and database search method

3.3.1 Star Catalogue Generation

For each of previously mentioned algorithms it is necessary to match patters, angles, areas, polar moments, etc. with already known data. Such data saved in the database prior to launching the spacecraft. The database is usually made out of star database collected by Earth's observatories.

For nearly every start pattern matching it is necessary to convert firstly the star positions to unit vectors. Most star catalogues use right ascension-declination coordinates while for star identification unit vectors must be usedMcBryde and Lightsey [62]. The following formula transforms the coordinates to unit vectors:

$$\boldsymbol{u} = \begin{bmatrix} \cos\alpha\cos\delta \\ \sin\alpha\cos\delta \\ \sin\delta \end{bmatrix} \tag{46}$$

where $\boldsymbol{u}$ is unit vector, α is right ascension and δ is declination.

Since Planar Triangle method was chosen, the rest of catalogue generation will be described for this method. For each three catalogue stars that are above certain light intensity threshold

$$i_a \geq i_{max} \tag{47a}$$

$$i_b \geq i_{max} \tag{47b}$$

$$i_c \geq i_{max} \tag{47c}$$

and in camera's FOV

$$\boldsymbol{u_a^T u_b} \geq \cos\theta_{FOV} \tag{48a}$$

$$\boldsymbol{u_b^T u_c} \geq \cos\theta_{FOV} \tag{48b}$$

$$\boldsymbol{u_c^T u_a} \geq \cos\theta_{FOV} \tag{48c}$$

area A and polar moment J should be calculated, exactly the same as in Planar Triangle method, however without variances. If all above conditions are met, star identification numbers, area and polar moment are saved in the database for later.

$\boldsymbol{u_x^T u_y}$ is equal to cosine of angle between the two unit vectors, therefore checking if condition in Equation 48 is satisfied is the same as stating the angular distance is less than FOV angle, what ensures that all three stars can be seen by camera at the same time.

3.3.2 Search Less Algorithm and k-vector

Search Less Algorithm (SLA) by Mortari [19] is special algorithm used to speed up looking up the corresponding stars in the on-board computer. Its heart is k-vector technique. Comparing to binary search technique, k-vector demonstrates high speed gain rate, from 10 to more than 50 times [20].

The k-vector database is built before spacecraft's start. The k-vector table is a structural database of all catalogued star pairs that could possibly fit into the camera FOV over the whole sky. The star pairs are ordered with increasing inter-star angle.

The k-vector technique works in the following way: $\boldsymbol{y}$ is the data vector of n elements ($n \gg 1$) and $\boldsymbol{s}$ is the same vector but sorted in ascending mode: $\boldsymbol{s}(i) \leq \boldsymbol{s}(i+1)$, and $\boldsymbol{s}(i) = \boldsymbol{y}(\boldsymbol{I}(i))$, where $\boldsymbol{I}$ is the integer vector of the sorting with length n [21].

In the Figure 18 it is shown how such k-vector construction looks like. In this example the database has 10 elements. X-axis describes k-vector elements $\boldsymbol{k}(i)$ and y-axis describes the sorted values $\boldsymbol{s}(i)$ (inter-star

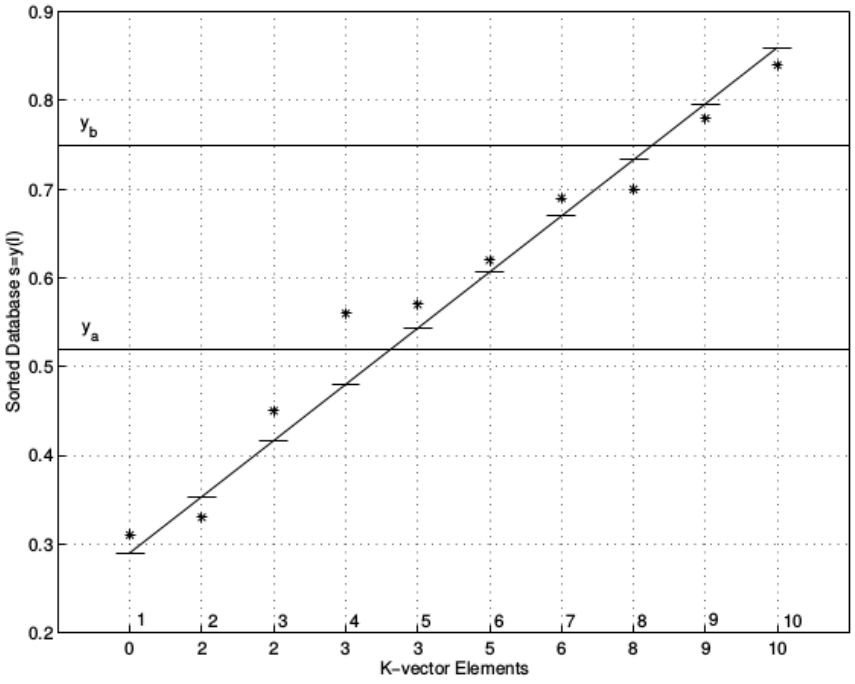


Figure 18: K-vector technique example, Image from [20]

angle in this case). The value of $\boldsymbol{k}(i)$ represents the number of elements of the database vector $\boldsymbol{y}$ below the value $z(i)$. The resulting k-vector is $\boldsymbol{k} = \{0, 2, 2, 3, 3, 5, 6, 8, 9, 10\}^T$.

The following equations describe how the k-vector database is created. The line connecting two extreme points, $[1, y_{min}]$ and $[n, y_{max}]$, has to be a little stepper and connect points $[1, y_{min} - \delta\epsilon]$ and $[n, y_{max} + \delta\epsilon]$. This assures $\boldsymbol{k}(1) = 0$ and $\boldsymbol{k}(n) = n$ and simplifies the code by avoiding many index checks.

$$z(x) = mx + q \begin{cases} m = \frac{y_{max} - y_{min} + \delta\epsilon}{n-1} \\ q = y_{min} - m - \delta\epsilon \end{cases} \tag{49}$$

$$\delta\epsilon = (n-1)\epsilon \tag{50}$$

ϵ is relative machine precision for double precision arithmetic

$$\epsilon \approx 22.2 \times 10^{-16} \tag{51}$$

Each other element can be found using the following equation:

$$\boldsymbol{k}(i) = j \quad where \quad s(j) \leq z(i) < s(j+1) \tag{52}$$

or

$$\boldsymbol{k}(i) = j \quad \text{where j is the greatest index such} \quad s(j) \leq \boldsymbol{y}(\boldsymbol{I}(i)) \quad \text{is satisfied.} \tag{53}$$

At this point k-vector has been built, and further use is quite simple. The equation to find indices identifying in $\boldsymbol{s}$ vector all possible data within the range $[y_a, y_b]$ is following:

$$j_b = \left\lfloor \frac{y_a - q}{m} \right\rfloor \quad and \quad j_t = \left\lceil \frac{y_b - q}{m} \right\rceil \tag{54}$$

In the example of Figure 18 it is $j_b = 4$ and $j_t = 9$. After indices are evaluated it is possible to calculate

$$k_{start} = \boldsymbol{k}(j_b) + 1 \quad and \quad k_{end} = \boldsymbol{k}(j_t) \tag{55}$$

Finally, having found k_{start} and k_{end} it is immediately known, that searched elements are $\boldsymbol{y}(i) \in [y_a, y_b]$, which are all the $\boldsymbol{y}(\boldsymbol{I}(i))$ where k ranges from k_{start} and k_{end}.

Note that in example the searched elements should be $k_{start} = 4$ and $k_{end} = 8$ while the presented technique returns $k_{start} = 4$ and $k_{end} = 9$. This happens due to fact that k_{start} and k_{end} have 50% possibility to not belong to the $[y_a, z(j_a + 1)]$ and $[z(j_b), y_b)]$ ranges respectively.

3.4 Attitude Determination

The last main part of the star-tracker program is attitude determination. After star pattern is found, there are come two lists of unit vectors. One list is in the body frame while the other is in the inertial frame. In order to find rotation between them and later CubeSat's attitude, an attitude determination method must be applied. A number of techniques were designed to solve this problem. Typically the output can come in the form of a quaternion, Euler angles or a rotation matrix (DCM).

The attitude determination methods should be fast and robust. This section describes few most common algorithms designed to solve this problem.

3.4.1 TRIAD

Three Axis Attitude Determination (TRIAD) is based on constructing two triads of orthonormal unit vectors using the vector information from input. Both triads are components of the same reference frame F_t expressed in the body and inertial frames[29].

$\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ are two vectors in body frame and $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are in reference frame. First step of the algorithm is following:

$$\boldsymbol{r}_1 = \boldsymbol{w}_1/|\boldsymbol{w}_1| \tag{56a}$$

$$\boldsymbol{r}_2 = (\boldsymbol{r}_1 x \boldsymbol{w}_2)/|\boldsymbol{r}_1 x \boldsymbol{w}_2| \tag{56b}$$

$$\boldsymbol{r}_3 = \boldsymbol{r}_1 x \boldsymbol{r}_2 \tag{56c}$$

and the following matrices in reference frame:

$$\boldsymbol{s}_1 = \boldsymbol{v}_1/|\boldsymbol{v}_1| \tag{57a}$$

$$\boldsymbol{s}_2 = (\boldsymbol{s}_1 x \boldsymbol{v}_2)/|\boldsymbol{s}_1 x \boldsymbol{v}_2| \tag{57b}$$

$$\boldsymbol{s}_3 = \boldsymbol{s}_1 x \boldsymbol{s}_2 \tag{57c}$$

where $\boldsymbol{r}_i, i \in \{1,2,3\}$ are matrices in body frame and $\boldsymbol{s}_i, i \in \{1,2,3\}$ are matrices in reference frame.

Finally the attitude matrix $\boldsymbol{A}$ is computed:

$$\boldsymbol{A} = \boldsymbol{r}_1 \cdot \boldsymbol{s}_1^T + \boldsymbol{r}_2 \cdot \boldsymbol{s}_2^T + \boldsymbol{r}_3 \cdot \boldsymbol{s}_3^T \tag{58}$$

The disadvantage of this algorithm is discarding some of the information, therefore losing the accuracy. The main advantage is that TRIAD is quite fast.

3.4.2 q-method

In 1968 Paul Davenport applied Wahba's problem to spacecraft attitude determination, via method called q-method[71].

Let loss function be represented in quaternion form:

$$L(A) = \frac{1}{2}\sum |\boldsymbol{b}_i - D(\boldsymbol{q})\boldsymbol{r}_i|^2 \tag{59}$$

where quaternion q is searched, that minimizes cost function L. Here, q is the eigenvector corresponding to the largest eigenvalue of K matrix, which is generated as follows:

$$\sigma = \sum \boldsymbol{b}_i^T \boldsymbol{r}_i \tag{60a}$$

$$B = \sum \boldsymbol{b}_i \boldsymbol{r}_i^T \tag{60b}$$

$$S = B + B^T \tag{60c}$$

$$\boldsymbol{Z} = \sum \boldsymbol{b}_i x \boldsymbol{r}_i \tag{60d}$$

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{S} - \sigma \boldsymbol{I}_3 & \boldsymbol{Z} \\ \boldsymbol{Z}^T & \sigma \end{bmatrix} \tag{61}$$

$\boldsymbol{I}_3$ is 3x3 identity matrix.

According to [72] the computational time and accuracy in q-method and TRIAD are comparable.

3.4.3 QUEST

The quaternion estimation (QUEST) method was proposed by Shuster [73]. The method builds on the q-method. The main goal for this method was to develop the quaternion estimator was to obtain finding the optimal eigenvalue for the loss function. The eigenvalue problem is reformulated to problem

of solving a characteristic equation. The minimization of loss function is following:

$$J(\boldsymbol{q}) = \frac{1}{2}\sum_{j=1}^{n}\frac{1}{\sigma_j^2}(\boldsymbol{b}_j - \boldsymbol{R}_b^i(\boldsymbol{q})\boldsymbol{r}_j)^T(\boldsymbol{b}_j - \boldsymbol{R}_b^i(\boldsymbol{q})\boldsymbol{r}_j) = \frac{1}{2}\sum_{j=1}^{n}\frac{1}{\sigma_j^2}(\boldsymbol{b}_j^T\boldsymbol{b}_j - 2\boldsymbol{b}_j^T\boldsymbol{R}_b^i(\boldsymbol{q})\boldsymbol{r}_j + \boldsymbol{r}_j^T\boldsymbol{r}_j) \tag{62}$$

where $\boldsymbol{r}_j$ is unit vector given in the NED frame, while $\boldsymbol{b}_j$ is unit vector given in the fixed BODY frame. The parameter σ_j is the standard deviation of the measurement error. Due to fact that both unit vectors have to be of equal length, the cost function can be reduced to:

$$J(\boldsymbol{q}) = \sum_{j=1}^{n}\frac{1}{\sigma_j^2}(1 - \boldsymbol{b}_j^T\boldsymbol{R}_b^i(\boldsymbol{q})\boldsymbol{r}_j) \tag{63}$$

The main advantage is that QUEST method can find exact solution after one step. The method is designed to preserve quaternion normalization. The problem is that it is mostly estimation, hence some accuracy is not perfect.

3.4.4 Singular Value Decomposition (SVD)

This method was proposed by Markley [74]. It is not usually used while it output type is Direction Cosine Matrix (DCM), not quaternions. It is used however for simulations, like for example by McBryde and Lightsey [62]. Since the needed output from attitude determination must be quaternion, not DCM, this method will not be used.

The algorithm works as follows:

Firstly the matrix $\boldsymbol{B}$ is calculated with n observed stars, where b_i is image star and r_i catalogue star for real star i:

$$\boldsymbol{B} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{b_i r_i^T} \tag{64}$$

Secondly singular value decomposition (SVD) of matrix $\boldsymbol{B}$ has to be found

$$\boldsymbol{B} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\boldsymbol{T}} \tag{65}$$

where matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal and diagonal matrix of singular values $\boldsymbol{S}$.

Now the proper orthogonal matrices $\boldsymbol{U}_+$ and $\boldsymbol{V}_+$ using:

$$\boldsymbol{U}_+ = \boldsymbol{U} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det\boldsymbol{U} \end{bmatrix} \tag{66}$$

$$\boldsymbol{V}_+ = \boldsymbol{V} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det\boldsymbol{V} \end{bmatrix} \tag{67}$$

The DCM $\boldsymbol{A}$ can be found by:

$$\boldsymbol{A} = \boldsymbol{U}_+\boldsymbol{V}_+^{\boldsymbol{T}} \tag{68}$$

3.4.5 Other techniques

There exist also other methods:

- The Fast Optimal Attitude Matrix [75],
- Predictive Attitude Determination Crassidis and Markley [76].

However they are not very popular and not often used.

3.4.6 Conclusions

From all the presented methods, QUEST was selected due to low computational requirements and good accuracy in attitude estimation.

QUEST outperformes q-method in terms of speed [28], and while it is less accurate it still qualifies for the CubeSat. Comparing to TRIAD, QUEST is more robust to noise. TRIAD is designed only for two vectors, hence in case of inaccurate measurements, it will yield erroneous result, even if there are more vector that could be used for estimation.

QUEST was also chosen by Huffman et al. [34] and Tappe [30] to be used on CubeSats' star-trackers.

4 Prototype

For now the following parts are finished in Python:

1. Centroiding
2. Planar Triangle Recognition with variations
3. k-vector - results were inconsistent. Moreover for just 1500 stars it didn't give any speed advantage. It is possible it would be worth using in case with more stars.
4. QUEST
5. generating star catalogue
6. connecting all the elements into one program

Testing

Kruijff et al. [77], Kruijff and vd Heiden [78], Dzamba and Enright [79],

The catalog was created with the following parameters: - stars with max magnitude 5; - FOV 10°what gave 1625 stars which gave 44922 triangles.

In other (source) works around 6000 stars were used to create catalog, hence another catalog was created with stars' magnitude 6.2, what gave 6279 stars and 1160853 triangles. This however proved to be both not accurate and a lot slower than in case of smaller catalog, hence the smaller catalog was choosen to be used in this startracker program.

The small catalog is good enough, because it covers most of the sky with at least 3 stars. Tests prove that in random part of the sky program is able to identify stars in around 90% of cases. It is best if no star with magntide greater than the given in the catalog would be passed from star centroid algorithm into star identifier, although star identifier can deal with some false stars with quite high accuracy.

5 Complete program

6 Future steps

Necessary things to do: - test with dedicated camera; - rewrite program to C/C++ as the Python Anaconda distribution for ARM architecture is experimental and it could break on the satellite.

Optional things to explore: - rewrite star identifier on GPU; - rewrite tracking mode to work more accurately; - get better values for sigm, sigx; - catalog with more stars; - use k-vector algorithm for more stars;

7 Testing of star-tracker

Tappe et al. [80]

References

[1] M. A. Swartwout, "A brief history of rideshares (and attack of the CubeSats)," in *Aerospace Conference, 2011 IEEE*. IEEE, 2011, pp. 1–15.

[2] H. Heidt, J. Puig-Suari, A. Moore, S. Nakasuka, and R. Twiggs, "CubeSat: A new generation of picosatellite for education and industry low-cost space experimentation," 2000.

[3] C. C. Liebe, "Accuracy performance of star trackers-a tutorial," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 2, pp. 587–599, 2002.

[4] M. A. Samaan, D. Mortari, T. Pollock, and J. L. Junkins, "Predictive centroiding for single and multiple FOVs star trackers," *Advances in the Astronautical Sciences*, vol. 112, pp. 59–71, 2002.

[5] M. W. Knutson, "Fast star tracker centroid algorithm for high performance CubeSat with air bearing validation," Master's thesis, Massachusetts Institute of Technology, 2012.

[6] M. Azizabadi, A. Behrad, and M. Ghaznavi-Ghoushchi, "VLSI implementation of star detection and centroid calculation algorithms for star tracking applications," *Journal of real-time image processing*, vol. 9, no. 1, pp. 127–140, 2014.

[7] M. Lindh, "Development and implementation of star tracker electronics," Master's thesis, 2014.

[8] P. Zhang, Q. Zhao, J. Liu, and N. Liu, "A brightness-referenced star identification algorithm for aps star trackers," *Sensors*, vol. 14, no. 10, pp. 18 498–18 514, 2014.

[9] C. L. Cole and J. L. Crassidis, "Fast star-pattern recognition using planar triangles," *Journal of guidance, control, and dynamics*, vol. 29, no. 1, pp. 64–71, 2006.

[10] C. L. Cole and J. Crassidus, "Fast star pattern recognition using spherical triangles," in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit. Providence, Rhode Island: AIAA*, 2004.

[11] D. Mortari, M. A. Samaan, C. Bruccoleri, and J. L. Junkins, "The pyramid star identification technique," *Navigation*, vol. 51, no. 3, pp. 171–183, 2004.

[12] M. Kolomenkin, S. Pollak, I. Shimshoni, and M. Lindenbaum, "Geometric voting algorithm for star trackers," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 2, pp. 441–456, 2008.

[13] C. Padgett and K. Kreutz-Delgado, "A grid algorithm for autonomous star identification," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 1, pp. 202–213, 1997.

[14] Y. Dong, F. Xing, and Z. You, "Brightness independent 4-star matching algorithm for lost-in-space 3-axis attitude acquisition," *Tsinghua Science & Technology*, vol. 11, no. 5, pp. 543–548, 2006.

[15] D. Mortari, "SP-search: A new algorithm for star pattern recognition," *Advances in the Astronautical Sciences*, vol. 102, no. Pt II, pp. 1165–1174, 1999.

[16] S. S. Miri and M. E. Shiri, "Star identification using Delaunay triangulation and distributed neural networks," *International Journal of Modeling and Optimization*, vol. 2, no. 3, p. 234, 2012.

[17] T. Lindblad, C. S. Lindsey, Å. Eide, Ö. Solberg, and A. Bolseth, "Star Identification using Neural Networks," 1997.

[18] C. Li, K. Li, L. Zhang, S. Jin, and J. Zu, "Star pattern recognition method based on neural network," *Chinese Science Bulletin*, vol. 48, no. 18, pp. 1927–1930, 2003.

[19] D. Mortari, "A fast on-board autonomous attitude determination system based on a new star-ID technique for a wide FOV star tracker," *Advances in the Astronautical Sciences*, vol. 93, pp. 893–904, 1996.

[20] D. Mortari and B. Neta, "K-vector range searching techniques," *Adv. Astronaut. Sci*, vol. 105, pp. 449–464, 2000.

[21] D. Mortari and J. Rogers, "A k-vector Approach to Sampling, Interpolation, and Approximation," *The Journal of the Astronautical Sciences*, vol. 60, no. 3-4, pp. 686–706, 2013.

[22] T. Delabie, "A highly efficient attitude estimation algorithm for star trackers based on optimal image matching," in *AIAA Guidance, Navigation and Control Conference, Minneapolis, Minnesota*, 2012.

[23] M. Shuster, "Kalman filtering of spacecraft attitude and the QUEST model," *Journal of the Astronautical Sciences*, vol. 38, pp. 377–393, 1990.

[24] Y. Cheng and M. D. Shuster, "Improvement to the Implementation of the QUEST Algorithm," *Journal of Guidance, Control, and Dynamics*, 2014.

[25] M. Psiaki, "Extended quest attitude determination filtering," in *NASA CONFERENCE PUBLICATION*. NASA, 1999, pp. 1–16.

[26] T. B. Rinnan, "Development and comparison of estimation methods for attitude determination," Master's thesis, Institutt for teknisk kybernetikk, 2012.

[27] J.-N. Juang, H.-Y. Kim, and J. L. Junkins, "An efficient and robust singular value method for star pattern recognition and attitude determination," *NASA, Tech. Rep. TM-2003-212142*.

[28] F. L. Markley and D. Mortari, "How to estimate attitude from vector observations," 1999.

[29] C. D. Hall, "Spacecraft attitude dynamics and control," *Lecture Notes posted on Handouts page [online]*, vol. 12, no. 2003, 2003. [Online]. Available: http://www.dept.aoe.vt.edu/ cdhall/courses/aoe4140/attde.pdf

[30] J. A. Tappe, "Development of star tracker system for accurate estimation of spacecraft attitude," Master's thesis, Monterey, California. Naval Postgraduate School, 2009.

[31] M. Gąska, "Moduł komputera pokładowego do satelity CubeSat z funkcją Star Tracker," Master's thesis, Warsaw University of Technology, 2016.

[32] D. Felikson, J. Hahmall, M. F. Vess, and M. Ekinci, "On-Orbit Solar Dynamics Observatory (SDO) Star Tracker Warm Pixel Analysis," in *AIAA Guidance, Navigation and Control Conference, Portland, Oregon*, vol. 6728, 2011.

[33] R. Kandiyil, "Attitude determination software for a star sensor," Master's thesis, Luleå University of Technology, 2010.

[34] K. M. Huffman *et al.*, "Designing star trackers to meet micro-satellite requirements," Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, 2006.

[35] K. D. Diaz, "Performance Analysis Of A Fixed Point Star Tracker Algorithm," Master's thesis, California Polytechnic State University, 2006.

[36] E. Jalabert, E. Fabacher, N. Guy, S. Lizy-Destrez, W. Rappin, and G. Rivier, "Optimization of star research algorithm for ESMO star tracker," 2011.

[37] S. Lizy-Destrez and D. Mimoun, "Str: a student developed star tracker for the esa-led esmo moon mission," 2010.

[38] A. Rose, "STAR integrated tracker," *arXiv preprint nucl-ex/0307015*, 2003.

[39] D. Mortari and A. Romoli, "StarNav III: a three fields of view star tracker," in *Aerospace Conference Proceedings, 2002. IEEE*, vol. 1. IEEE, 2002, pp. 1–57.

[40] M. Cannata, M. Greene, S. Mulligan, and V. Popovici, "Autonomous star-imaging attitude sensor (ASIAS)," *Faculty of Science and Engineering. York University*, pp. 2006–07, 2007.

[41] M. Swartwout, "The first one hundred cubesats: A statistical look," *Journal of Small Satellites*, vol. 2, no. 2, pp. 213–233, 2013.

[42] "Nanosatsatellite and CubeSat Database," http://www.nanosats.eu/, accessed: 2017-08-15.

[43] "PW-Sat2: oficjalna strona projektu PW-SAT 2 (oraz PW-SAT)," https://pw-sat.pl/, accessed: 2017-08-15.

[44] "BRITE-PL Pierwszy polski satelita naukowy," http://www.brite-pl.pl/pliki/nauka.html, accessed: 2017-08-15.

[45] "Polskie Radio: Pierwszy polski satelita poleciał w kosmos," http://www.polskieradio.pl/7/129/Artykul/536883,Pierwszy-polski-satelita-polecial-w-kosmos, accessed: 2017-08-15.

[46] "BRITE-PL, PL2 (CanX 3C, 3D / Lem, Heweliusz) - Gunter's Space Page," http://space.skyrocket.de/doc_sdat/brite-pl.htm, accessed: 2017-08-15.

[47] "Future Planetary Expeditions: CubeSats to the Planets," http://futureplanets.blogspot.com/2013/10/, accessed: 2017-08-15.

[48] "Warsaw University of Technology - News - PW-SAT 2 gets funding," https://www.pw.edu.pl/Studenci/Aktualnosci/Satelita-studencki-PW-Sat2-otrzyma-dofinansowanie, accessed: 2017-08-15.

[49] "GPS: The Global Positioning System," http://www.gps.gov/systems/gps/space/, accessed: 2017-08-18.

[50] "NASA's Utilization of Global Positioning System (GPS)," https://www.nasa.gov/directorates/heo/scan/communications/ policy/GPS_Utilization.html, accessed: 2017-08-18.

[51] "Space Station Using GPS in Attitude Control," https://www.nasa.gov/centers/johnson/news/releases/2002/j02-61.html, accessed: 2017-08-18.

[52] V. Capuano, C. Botteron, Y. Wang, J. Tian, J. Leclère, and P.-A. Farine, "GNSS/INS/Star tracker integrated navigation system for Earth-Moon transfer orbit," in *ION GNSS+ 2014*, no. EPFL-CONF-202129, 2014.

[53] W. J. Larson and J. R. Wertz, "Space mission analysis and design," Microcosm, Inc., Torrance, CA (US), Tech. Rep., 1992.

[54] "Earth's navigation orbits image," https://upload.wikimedia.org/wikipedia/commons/thumb/b/b4/Comparison_satellite_navigation_orbits.svg/ 512px-Comparison_satellite_navigation_orbits.svg.png, accessed: 2017-08-18.

[55] "NED frame image," http://www.basicairdata.eu/knowledge-center/background-topics/coordinate-system/, accessed: 2017-09-02.

[56] L. Euler, "Formulae generales pro translatione quacunque corporum rigidorum," *Novi Acad. Sci. Petrop*, vol. 20, pp. 189–207, 1775.

[57] W. R. Hamilton, "LXXVIII. On quaternions; or on a new system of imaginaries in Algebra: To the editors of the Philosophical Magazine and Journal," *Philosophical Magazine Series 3*, vol. 25, no. 169, pp. 489–495, 1844.

[58] A. Cayley, "XIII. On certain results relating to quaternions: To the editors of the Philosophical Magazine and Journal," *Philosophical Magazine Series 3*, vol. 26, no. 171, pp. 141–145, 1845.

[59] R. Courant and D. Hilbert, "Methods of mathematical physics, Volume I," 1953.

[60] K. Shoemake, "Animating rotation with quaternion curves," in *ACM SIGGRAPH computer graphics*, vol. 19, no. 3. ACM, 1985, pp. 245–254.

[61] G. Wahba, "A least squares estimate of satellite attitude," *SIAM review*, vol. 7, no. 3, pp. 409–409, 1965.

[62] C. R. McBryde and E. G. Lightsey, "A star tracker design for CubeSats," in *Aerospace Conference, 2012 IEEE*. IEEE, 2012, pp. 1–14.

[63] B. B. Spratling and D. Mortari, "A survey on star identification algorithms," *Algorithms*, vol. 2, no. 1, pp. 93–107, 2009.

[64] D. Gottlieb, "Star pattern recognition techniques," *Spacecraft Attitude Determination and Control, The Netherlands*, pp. 257–266, 1978.

[65] F. Alidoost, F. Dadras Javan, and AWT_TAG, "A Review of Pattern Matching Methods in Star Trackers," *Geospatial Engineering Journal*, vol. 4, 2013. [Online]. Available: http://gej.issge.ir/article-1-126-en.html

[66] T. Brady, C. Tillier, R. Brown, A. Jimenez, and A. Kourepenis, "The inertial stellar compass: A new direction in spacecraft attitude determination," 2002.

[67] G. Crew, R. Vanderspek, and J. Doty, "Hete experience with the pyramid algorithm," *MIT Center for Space Research, Cambridge, MA*, vol. 2139, 2002.

[68] M. Jacox, J. Ochoa, J. Zbranek, B. Wood, A. Katake, and C. Bruccoleri, "Method and Apparatus for Automatic Identification of Celestial Bodies," Apr. 13 2006, uS Patent App. 11/279,668.

[69] "StarVision Technologies Secures Exclusive License to Star Identification Software," http://www.prweb.com/releases/2005/05/prweb245145.htm, accessed: 2017-09-06.

[70] M. A. Samaan, D. Mortari, and J. L. Junkins, "Recursive mode star identification algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1246–1254, 2005.

[71] J. Keat, "Analysis of least-squares attitude determination routine DOAOP," Technical Report CSC/TM-77/6034, Comp. Sc. Corp, Tech. Rep., 1977.

[72] O. Çelik and C. Hajiyev, "A comparison of attitude determination methods for small satellites," in *Recent Advances in Space Technologies (RAST), 2013 6th International Conference on*. IEEE, 2013, pp. 261–264.

[73] M. D. Shuster, "Approximate algorithms for fast optimal attitude computation," in *Guidance and Control Conference*, 1978, pp. 88–95.

[74] F. L. Markley, "Attitude determination using vector observations and the singular value decomposition," *Journal of the Astronautical Sciences*, vol. 36, no. 3, pp. 245–258, 1988.

[75] ——, "Attitude determination using vector observations: A fast optimal matrix algorithm," 1993.

[76] J. Crassidis and L. Markley, "A Predictive Attitude Determination Algorithm," in *NASA Conference Publication*. NASA, 1997, pp. 249–264.

[77] M. Kruijff, E. Heide, C. De Boom, and N. Heiden, "Star sensor algorithm application and spin-off," in *54th International Astronautical Congress of the International Astronautical Federation(IAF)*, 2003.

[78] M. Kruijff and N. vd Heiden, "Automated star sensor performance assessment using real-sky data of MEFIST II," *Moon*, vol. 100, no. 1, pp. 500–2, 2003.

[79] T. Dzamba and J. Enright, "Ground testing strategies for verifying the slew rate tolerance of star trackers," *Sensors*, vol. 14, no. 3, pp. 3939–3964, 2014.

[80] J. Tappe, J. J. Kim, A. Jordan, and B. Agrawal, "Star tracker attitude estimation for an indoor ground-based spacecraft simulator," in *AIAA Guidance, Navigation, and Control Conference*, vol. 1, 2011, pp. 1116–1122.

List of Tables

List of Figures