Wrocław University of Science and Technology
Faculty of Electronics

Attitude determination system based on stars observation

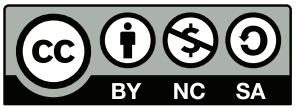
Krystian Borodacz

Class: Intermediate project
Supervisor: Ph. D. Witold Paluszyński

February 7, 2019

Abstract

The goal of the project was to develop an algorithm such that allow to determine attitude of the camera view axis in the International Celestial Reference System based on stars visible in the night sky image taken with a camera with wide field of view.



1 Introduction

One of the main tasks of spacecraft navigational system is to determine its orientation in space. The most accurate method is using devices called star sensors. The device consists of camera and electronic part for image processing. Orientation in space is determined based on stars visible in telescope field of view [4].

Project is an attempt to develop an algorithm of such a system using the means available to the author. The stationary mounted all-sky camera located in Białków (Poland) has been used and a pin-hole camera model has been assumed. It has been assumed that, due to light pollution, only the brightest stars will be visible in the image. This is not a problem because camera has wide field of view and there will be always several stars visible. Based on that assumptions, and the camera location, the database has been created using hundred brightest stars with declination greater than -30 degrees. Real stars position in International Celestial Reference System (ICRS) has been taken from Hipparcos star catalogue [1].

2 Main principle of work

First of all, the database containing characteristics of the real stars have to be created. The Database have to contain also positions of the stars, from which each characteristic has been computed, to determine the camera attitude after characteristics match.

When a night sky image is given, the position of the stars in the image have to be determined. Next, the positions of the stars in the image are converted to the positions in the camera fixed frame, using camera model. For designated stars positions, the characteristics are computed.

Characteristics computed for the stars in the image are compared with characteristics in the database. The best matching characteristics determines the attitude of the camera view axis.

3 Characteristic

Characteristic is computed for set of three stars, where first of them is called *central* star and others are called *neighbour* stars. Characteristic consists of three values [5]:

1. an angular distance from central star to first neighbour star
2. an angular distance from central star to second neighbour star
3. an angle at central star between arcs from central star to each from two neighbours stars.

4 Algorithm

An algorithm has been written in Python as a collection of functions. The following Python modules has been used: numpy, matplotlib, cv2, itertools, spherical_geometry, pandas, skyfield.

First, the image is converted to the gray colorspace. Then stars are searched in the image using `adaptiveThreshold()` function from cv2 module. The result is a mask pointing to the pixels containing star. Because image is slightly defocused, stars are visible in several pixels. It is assumed that the connected pixels represent one star. Precise position of the star in the image is computed as a center of mass of the star image with pixels brightness values as a weights.

An angles of right ascension (α) and declination (β) in camera frame is computed using inverted model of the camera (1), where (y, x) is position of the star in the image, and f is camera focal length computed using equation (2), where N is the number of pixels, and γ is angle of view. The position (X, Y, Z) of the star on the unit sphere is computed using equation (3). Brightness of the star is computed as a sum of values of the pixels creating image of the star.

$$\begin{cases} \alpha = \arctan(\frac{x}{f}) \\ \beta = \arctan\left(\frac{y}{f}\cos(\alpha)\right) \end{cases} \tag{1}$$

$$f = \frac{N/2}{\tan(\gamma/2)} \tag{2}$$

$$\begin{cases} X = \cos(\beta)\cos(\alpha) \\ Y = \cos(\beta)\sin(\alpha) \\ Z = \sin(\beta) \end{cases} \tag{3}$$

For each permutation of 3 from N brightness stars the characteristic is computed. N is the number of stars which brightness is greater than 650. This value has been selected based on histogram of brightness of stars from several images. For computation of the characteristics, the spherical_geometry python module has been used.

Computed characteristics are compared with database and an error of the comparison between characteristics is computed as a root mean square of the differences of each value in characteristic. Corresponding stars between image and database are assumed as that with the smallest characteristic comparison error. Based on that, the rotation matrix which rotates ICRS to the camera coordinates is computed.

Matrix of rotation is computed in two steps. First, the ICRS is rotated such that the direction to the central stars of the matched characteristics coincide. Then it is rotated around vector directed to the central star to coincide the neighbour stars. From the rotation matrix, the Euler angles

are derived. Transformations between axis–angle, rotation matrix and euler angles representation has been made with transforms3D Python module.

At the end, in the image are marked matched stars found in the image (blue circles), matched stars found in database (green circles) and projection of that stars from the database which should be visible based on computed transformation (red circles).

5 Effects

Example of the obtained result is presented in the figure 1. It can be noted, that near the center of the image the Cassiopeia constellation is visible, both as a stars visible in the image and as a projection of the stars from the database which should be visible based on the computed transformation. Unfortunately the stars doesn't coincide. Projection of database should be rotated approximately 45 degrees clockwise. Note also that matched stars (blue and green circles) coincide quite well. We can point which star was used as a central star (stars with exact coincidence) and which was a neighbour stars.

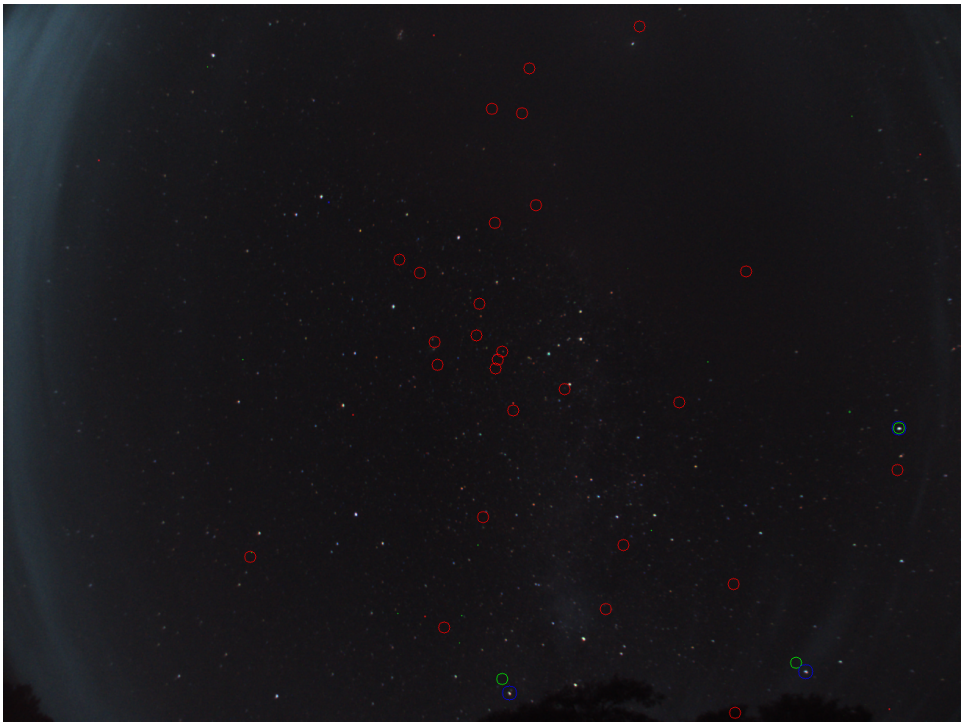


Figure 1: Example of the obtained results

In the figure 2 presented is the same image, but with stars selected and matched by hand. Only transformations are computed by algorithm. Note

that pin-hole camera model is used for cameras with narrow field of view while in project has been used camera with wide field of view. This can explain visible differences. Taking this into consideration, it can be stated that stars in the image and projection of the stars from the database coincide quite well.

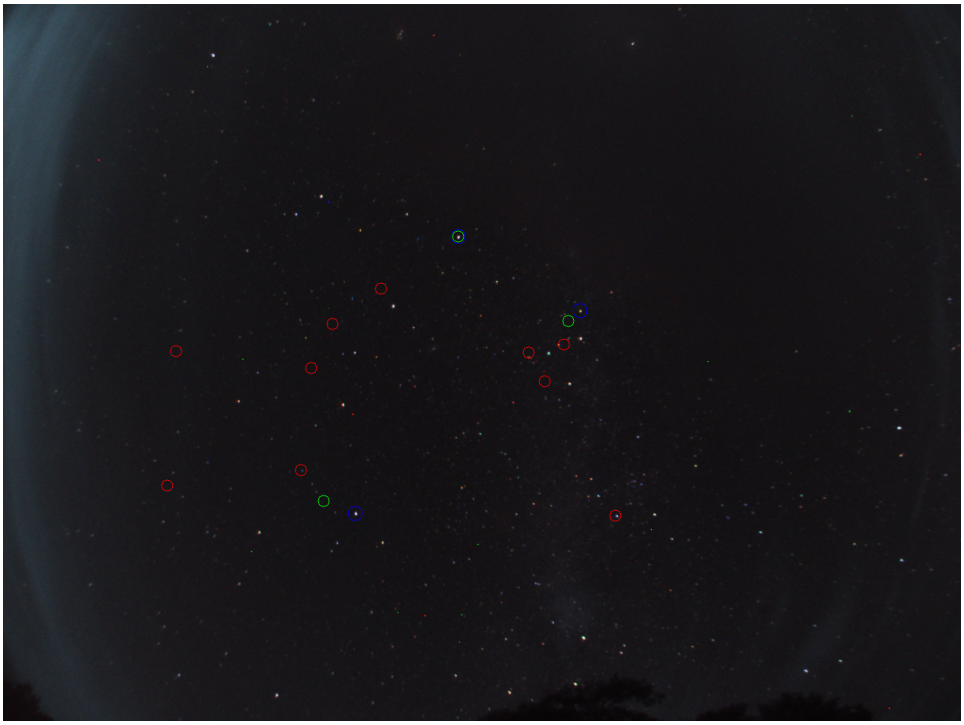


Figure 2: Example of the obtained results, with stars selected and matched by hand

6 Conclusions

As we can see in the figure 2, the rotations are computed correctly. The problem visible in the 1 figure is caused by incorrect match of the characteristics. Method used for selecting the best match between characteristics could be improved. Pin-hole camera model gives quite well results also for cameras with wide field of view, but better model could be used. Note also, that improper model can be source of mistakes of the matching algorithm. Algorithm of star detection and rotation computation works well.

References

[1] Hipparcos star catalogue. `http://cdsarc.u-strasbg.fr/ftp/I/239/hip_main.dat.gz`

[2] Astrometry.net — Tool for image astrometric calibration. `http://nova.astrometry.net/`

[3] M. W. Knutson, D. Miller: *Fast Star Tracker Centroid Algorithm for High Performance CubeSat with Air Bearing Validation.*

[4] W. Ley, K. Wittmann: *Handbook of Space Technology.* John Wiley & Sons, 2009

[5] C. C. Liebe: *Pattern Recognition of Star Constellations for Spacecraft Applications.* IEEE AES Systems Magazine, 1993