Characteristics study and robustness comparison of various star pattern recognition algorithm for star sensor using digital simulation

Nugraha Setya Ardi¹, Ridanto Eko Poetro¹, M Arif Saifudin², Luqman Fathurrohim¹ and Ilham Akbar Courie¹

¹ Faculty of Mechanical and Aerospace Engineering, Bandung Institute of Technology, Indonesia.
² Indonesian National Institute of Aeronautics And Space (LAPAN)

E-mail: nugrahaardi53@gmail.com

Abstract. The last few years the use of star sensor, which is the most accurate sensor on satellites, began to penetrate into micro satellites and nano satellites, previously used only on high-end satellites. Hence we need to develop star sensor according to the specific needs of microsatellite and nano satellites. One of star sensor’s aspect is star pattern recognition algorithm. A good star pattern recognition algorithm must have a high level of matching success when there are a lot of disturbances (robust) and fast processing time. There are different types of star pattern recognition algorithms, which have their respective advantages and disadvantages, so it is necessary to characterize and compare the various algorithms. In this paper, the characterization and comparison of robustness in different types of star pattern recognition algorithm are done using digital simulation. The simulation results show that the success rate does not always rise with the number of stars used for matching because it depends on how the algorithm works. In addition, the more the number of stars are used, the longer processing time, and vice versa. The best algorithm in terms of success rate is an algorithm that uses a combination pattern of all possible triangles, in other words most robust, but has the longest processing time. This long processing time can be overcome by optimizing GSC (Guide Star Catalogue). From the results the multitriangles 2 algorithm has the best success rate compared to the other algorithms, with success rate for 5, 6, and 7 stars configuration are 97.8%, 98.7%, and 99.8% respectively.

1. Introduction

Star sensor is the most accurate attitude sensor in satellite [1], hence it is the most researched attitude sensor among others such as sun sensor, magnetometer, horizon sensor, etc [2]. It has two aspects, namely hardware and software (algorithm) [3]. Hardware aspect consists of buffer to minimize disturbance light, optical lens to focus the image, and imager or image sensor to capture the star image [2]. Algorithm aspect consists of star spot centroiding algorithm to detect position of star’s center [4], star pattern recognition algorithm to match pattern from star image with GSC (Guide Star Catalogue), and attitude estimation algorithm to estimate the attitude of star sensor. In algorithm aspect, star pattern recognition algorithm is widely researched because it plays important role: Firstly, a good star sensor should be able to accurately match star image’s pattern with star pattern from GSC. If matching process failed, then star sensor failed. Secondly, a good star sensor should has fast processing time in matching star image’s pattern with star pattern from GSC to provide first attitude [5]. There are many star pattern recognition algorithm that have been developed by researchers which have their own advantages and drawbacks [6]. This paper deals with the development and comparison of some star pattern recognition algorithms. Theree algorithms will be presented and compared each other. This paper focused on star
pattern recognition only and attitude calculation is not presented. The algorithms is performed by digital simulation. The performance of the algorithms are measured by success rate of pattern matching and its processing time for various star image resolutions. The first algorithm which will be discussed in this paper is algorithm which uses pivoted triangles pattern which first introduced by Craig L. Cole and John L. Cassidis [7]. The second algorithm is algorithm which uses combination of all possible triangles created by \(n\) stars [8]. The last algorithm is algorithm which uses only angular distance between stars [1].

2. Methodology

The methodology in creating star pattern recognition algorithm is as follows (figure 1.1).

Firstly, random input boresight consists of right ascension (\(\alpha\)), declination (\(\delta\)), and roll (\(\varphi\)) is used, as shown in figure 1.2., to create star image from star catalogue. We use SAO J2000 Star Catalogue with star whose magnitude is less than 6 to simulate star image. Then the star image is processed to get star’s center position using a star spot centroiding algorithm. In this paper we use centroid method [9]. Then star pattern are generated from star image and matching process is done. If the matched pattern is within FOV, then pattern recognition process is success, and if not, then it is failed. Those processes are done 1000 times for every different boresight to represent all sky direction. Then success rate is calculated by dividing the number of success by 1000. Digital simulation is done for various star image resolutions.

**Figure 1.1.** Flowchart of algorithm analysis (a) and digital simulation (b).
3. Star pattern recognition algorithm

In this section, we explain three algorithms we developed that will be analyzed and compared each other.

3.1. Multitriangles 1 algorithm

The main idea of this algorithm is using pivoted triangles to solve the redundancy problem. The steps are as follow:

a. Choose a closest star to the center of FOV, namely central star.

b. Choose \( n - 1 \) closest stars to the central star, namely star 2, star 3,..., star \( n \).

c. Choose a star as a pivot star among star 2 to star \( n \). In this work we choose star 2 as pivot star.

d. Create all possible triangle using central star, pivot star, and star 3 to star \( n \).

e. Calculate their planar angles, and store them in an array matrix \( \theta_{sl} \)

\[
\theta_{sl} = \begin{bmatrix}
\theta_{s11} & \theta_{s12} & \theta_{s13} \\
\theta_{s21} & \theta_{s22} & \theta_{s23} \\
\vdots & \vdots & \vdots \\
\theta_{s(n-2)1} & \theta_{s(n-2)2} & \theta_{s(n-2)3}
\end{bmatrix}
\]  

(1)

where the elements in matrix’s row are sorted ascendingly.

f. For every star \( i \) in GSC, choose its \( n - 1 \) neighbour stars. Create all possible triangle using step c and d, and compute all planar angles and store them in an array matrix namely \( \theta_{ci} \)
\[ \theta_c = \begin{bmatrix} \theta_{c11} & \theta_{c12} & \theta_{c13} \\ \theta_{c21} & \theta_{c22} & \theta_{c23} \\ \vdots & \vdots & \vdots \\ \theta_{c(n-2)1} & \theta_{c(n-2)2} & \theta_{c(n-2)3} \end{bmatrix} \]

where the elements in row are sorted ascendingly.

g. For every star \( i \) in GSC, compute
\[ e_i = \sum_{N=1}^{n-2} \text{sum} |\theta_{siN} - \theta_{cN}| \]

where \( \theta_{siN} \) is \( \theta_{si} \)'s Nth row, \( \theta_{cN} \) is \( \theta_c \)'s Nth row, \text{sum} is an operator to sum all of the matrix's elements, and \( |\theta_{siN} - \theta_{cN}| \) is an operator to make all elements of matrix become positive.

h. Choose the smallest value of \( e_i \) and its corresponded star \( i \) in GSC. That star \( i \) is the matched pattern.

Pivoting is needed to solve the redundancy problem. Suppose there are three candidate patterns from GSC as shown in figure 3. Then in figure 3 (1), the use of 3 stars can not solve the redundancy problem. When we use 4 stars, the redundancy remains unsolved, while we use 5 stars, the redundancy problem was solved and give one unique pattern solution.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{redundancy_problem.png}
\caption{Redundancy problem.}
\end{figure}

But the problem with this algorithm is when the pivot star is swapped, then the results will be wrong, as shown in figure 4.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{swapped_pivot_star.png}
\caption{The swapped pivot star.}
\end{figure}

In figure 4 leftside, the pivot star is not swapped hence the matched pattern from GSC is correct, while in rightside, the pivot star is swapped with star 3, hence the matched pattern from GSC is wrong. The exchange between star 2 to star 3 as pivot star can occur caused by the error of star spot centroiding algorithm, because the labelling between star 2 and star 3 is done based on its distance to central star. When the distance of star 2 and star 3 to central star is almost the same, then the error from star spot centroiding result could swap both label.
3.2. Multitriangles 2 algorithm

The main idea of this algorithm is using $C_3^n$ combination of triangles pattern, that is when there are $n$ stars to be used to do pattern recognition, then there will be $C_3^n = \frac{n!}{(n-3)!3!}$ triplet combination of stars to create triangle patterns. For example if there are 4 stars, namely star 1, star 2, star 3, and star 4, then the triangle patterns is formed by star 1-2-3, 1-2-4, 2-3-4, and 1-3-4. The steps are as follows

a. Choose a closest star to the center of FOV, namely central star.
b. Choose $n-1$ closest stars to the central star, namely star 2, star 3,..., star $n$.
c. Create all possible triplet combination of stars. A simple algorithm to generate all possible triplet combination is as follows

```plaintext
for i = 1 : n-2
    for j = i+1 : n-1
        for k = j+1 : n
            combination = [i j k]
            end
        end
    end
end
```

For example, if $n = 5$ then the pattern is depicted by figure 5

![Figure 5. Generated pattern for 5 stars in star image.](image)

```plaintext
d. Compute all triplet planar angles in every triangle. Then stored in an array matrix $\theta_s$ which has size $C_3^n \times 3$

$$
\theta_s = \begin{bmatrix}
\theta_{s11} & \theta_{s12} & \theta_{s13} \\
\theta_{s21} & \theta_{s22} & \theta_{s23} \\
\vdots & \vdots & \vdots \\
\theta_{sc1} & \theta_{sc2} & \theta_{sc3}
\end{bmatrix}
$$

where the elements in matrix’s row are sorted ascendingly.

e. For every star $i$ in GSC, choose its $n-1$ neighbour stars. Create all possible triplet combination of stars using step c, and compute all planar angles and stored in an array matrix namely $\theta_{ci}$

$$
\theta_{ci} = \begin{bmatrix}
\theta_{ci1} & \theta_{ci2} & \theta_{ci3} \\
\theta_{c11} & \theta_{c12} & \theta_{c13} \\
\vdots & \vdots & \vdots \\
\theta_{csc1} & \theta_{csc2} & \theta_{csc3}
\end{bmatrix}
$$
```
The generated pattern from GSC is depicted by figure 6

![Figure 6. Generated pattern for every star \( i \) in GSC.](image)

- Reduce every \( \theta_s \)'s row with every \( \theta_{ci} \)'s row and take its total absolute error. If the error is less than a particular value \( \varepsilon \), give label 1 and if not label 0. In this paper we use \( \varepsilon = 1^\circ \). The labels are stored in an matrix \( \Delta_i \) which has size \( 1 \times C_3^n \). Do the same step for every star \( i \) from star catalogue.
- Select \( \Delta_i \) which has the most number of label 1. The selected \( \Delta_i \) is the matched pattern.

In brief, this algorithm works with a voting principle. A pattern which has the most number of matched triangle is the matched pattern.

![Figure 7. Pattern candidate from GSC.](image)

Suppose there are two candidates as shown by figure 7. Then the voting scheme is as follows

![Figure 8. Voting scheme between two candidates.](image)

In figure 8, the first column is pattern candidate form GSC, while the second column is pattern from star image. The left side is candidate 1 and the right side is candidate 2. The green arrow is for the
matched triangle, while red arrow for the unmatched triangle. Candidate 2 has more matched triangle than candidate 2, thus the matched pattern is candidate 2.

3.3. Radial pattern algorithm
The main idea of this algorithm is using planar angles between two vectors that direct from central star to the neighborhood stars. The steps are as follows
a. Choose a closest star to the center of FOV, namely central star.
b. Choose \( n - 1 \) closest stars to the central star, namely star 2, star 3, ..., star \( n \).
c. Choose a star among star 2 to star \( n \) as a pivot star. In this work, we use star 2 as pivot star. Then the pivot vector is a vector that direct from central star to star 2, namely \( \vec{r}_{p} \).
d. Create the remaining possible vector that direct from central star to star 3 to star \( n \), namely \( \vec{r}_{13}, \vec{r}_{14}, ..., \vec{r}_{1n} \). For 5 stars, the generated pattern is depicted by figure 9.

![Figure 9. An example of generated pattern using 5 stars.](image)

e. Compute the angles between \( \vec{r}_{p} \) and \( \vec{r}_{1N} \) for \( N \) from 3 to \( n \). Then store them in an array matrix \( \theta_{sl} \)
\[
\theta_{sl} = [\theta_{s1}, \theta_{s2}, ..., \theta_{s(n-2)}].
\]
where the elements in row are sorted ascendingly.
f. For every star \( i \) in GSC, do step b to e, and store the angles in an array matrix \( \theta_{c} \)
\[
\theta_{c} = [\theta_{c1}, \theta_{c2}, ..., \theta_{c(n-2)}].
\]
where the elements in row are sorted ascendingly.
g. Compute
\[
e_{i} = \text{sum}|\theta_{c} - \theta_{sl}| \tag{8}
\]
where \( \text{sum} \) is an operator to sum all of the matrix’s elements, and \( |\theta_{c} - \theta_{sl}| \) is an operator to make all elements of matrix become positive.
h. Choose the smallest value of \( e_{i} \) and its corresponding star \( i \) in GSC. That star \( i \) is the matched pattern.
The drawback of this algorithm is like multitriangles 2 algorithm, that is the exchange of pivot star. Beside that, this algorithm only use the angle between \( \vec{r}_{p} \) and \( \vec{r}_{1N} \), hence it can not differ the angular distance between central star and its neighbor, as shown in figure 10.
Figure 10. The drawback of radial pattern algorithm.

In figure 10, the value of $\theta_1$, $\theta_2$, $\theta_3$ is the same between star image (d) and its candidate (a,b,c), whereas the candidates have difference angular distance between central star and its neighbors compared to star image’s, hence the matching process failed.

This algorithm uses the following two GSC: primary and secondary GSC. Structures are as follows [10]

**Table 1.** The structure of primary GSC.

| ID Star | Right ascension | Declination |
|---------|-----------------|-------------|
| 1       | $\alpha_1$      | $\delta_1$  |
| 2       | $\alpha_2$      | $\delta_2$  |
| ...     | ...             | ...         |
| n       | $\alpha_n$      | $\delta_n$  |

**Table 2.** The structure of secondary GSC.

| ID Star | ID neighbour 1 | ID neighbour 2 | ... | ID neighbour m |
|---------|----------------|----------------|-----|----------------|
| 1       | IDn 1          | IDn 2          | ... | IDn m          |
| 2       | IDn 1          | IDn 2          | ... | IDn m          |
| ...     | ...            | ...            | ... | ...            |
| n       | IDn 1          | IDn 2          | ... | IDn m          |

where ID neighbour 1 to ID neighbour m is ordered ascendingly according to its distance from main ID star. We use SAO J2000 Star Catalogue with star whose magnitude is less than 6 as GSC.

4. Results and Discussion

Digital simulation are done using the following hardware specifications:
- Processor: AMD FX 9830P 3.00 – 3.50 GHz
- RAM : 16 GB
- Graphic card: Radeon RX 460 4 GB DDR 4

While the digital simulation setups are as follows
- FOV = 17°
- Star’s magnitude limit in star image = 6
- Number of random input boresights = 1000
- GSC = SAO J2000
- Resolution = 1024×1024
**Table 3.** Percentage of successful recognition of each algorithm.

| n stars | Multitriangles 1 (%) | Multitriangles 2 (%) | Radial Pattern (%) |
|---------|----------------------|----------------------|--------------------|
| 3       | 82.3                 | 82.3                 | 24.5               |
| 4       | 91.6                 | 93.1                 | 89.8               |
| 5       | 89.7                 | 97.8                 | 90.5               |
| 6       | 86.2                 | 98.7                 | 86.7               |
| 7       | 89.4                 | 99.8                 | 89.3               |

**Table 4.** Processing time for each algorithm.

| n stars | Multitriangles 1 (s) | Multitriangles 2 (s) | Radial Pattern (s) |
|---------|----------------------|----------------------|--------------------|
| 3       | 0.3470846            | 0.347085             | 0.333843           |
| 4       | 0.58188              | 1.428472             | 0.62587            |
| 5       | 0.8076906            | 4.394685             | 0.757477           |
| 6       | 1.0262146            | 12.62772             | 1.023245           |
| 7       | 1.2650549            | 30.15096             | 1.260217           |

**Figure 11.** Success rate comparison.

**Figure 12.** Processing time comparison.
From table 3 and figure 11, we can see that multitriangles 2 has the best success rate compared to the other algorithms, with success rate for 5, 6, and 7 stars configuration are 97.8%, 98.7%, and 99.8% respectively. While for multitriangles 1 and radial pattern algorithm, beside the success rate is quite similar, we can see that when the number of stars increased, then the success rate is declined. This is because those algorithms label the stars based on the distance to central star.

![Figure 13. Multitriangles 1 using 5 stars.](image1)

![Figure 14. Multitriangles 1 using 7 stars without false labelling.](image2)

![Figure 15. Multitriangles 1 using 7 stars with false labelling.](image3)

From figure 13, when multitriangles 1 uses 5 stars and there is no false labelling, the pattern matched. When it uses 7 stars and there is no false labelling as shown in figure 14, the matched pattern is still correct (in figure 14 rightside, 1-2-6 and 1-2-7 triangle match the right triangle, hence $\varepsilon$ is small), while when there is false labelling as shown in figure 15, 1-2-6 and 1-2-7 triangle match the wrong triangle from GSC, hence $\varepsilon$ become large and that $\varepsilon$ is not chosen because there is another smaller $\varepsilon$ and the matched pattern is false.
Beside the wrong labelling, multitriangles 1 algorithm’s drawback is when using bigger number of stars, the star selection is wrong caused by star spot centroiding error as shown in figure 16, hence the pattern become wrong.

While for radial pattern algorithm, same as multitriangles 1, when the pivot star is wrong then the pattern is wrong. Beside that, when the bigger number of stars are used, star selection could be wrong as shown in figure 17, hence the pattern become wrong.

While for multitriangles 2 algorithm, the success increases as the number of stars increases, because multitriangles 2 works by choosing a pattern with the largest number of matched triangles. Hence, despite there is wrong star selection as shown in figure 18, the number of matched triangle still large, hence it is still chosen as matched pattern.

From figure 12 we can see that multitriangle 2’s processing time increases exponentially as the number of stars increase, while multitriangle 1 and radial pattern’s processing time increase linearly. This is because multitriangle 2 uses $C^3_n$ combination, as shown in figure 10 rightside.

5. Concluding remarks
From the results and discussion, we can conclude that algorithm’s performance vary from one algorithm to another depending on how algorithm works. Algorithm with the best success rate is multitriangle 2 algorithm, which uses combination of all triangles pattern, with success rate for 5, 6, and
7 stars configuration are 97.8%, 98.7%, and 99.8% respectively, but it has the longest processing time which increases exponentially. For the future works, we will implement these algorithms in hardware in the loop simulation and field test of star observation.

References
[1] Liebe C C 1995 Star trackers for attitude determination IEEE Aerospace and Electronic Systems Magazine 10 pp 10-16
[2] Huffman K M 2006 Designing Star Trackers to Meet Micro-satellite Requirements (Cambridge: Massachusetts Institute of Technology)
[3] Dikmen S 2016 Development of Star Tracker Attitude and Position Determination System for Spacecraft Maneuvering and Docking Facility (Luleå: Luleå University of Technology)
[4] Delabie T 2016 Star Tracker Algorithms and a Low-Cost Attitude Determination and Control System for Space Missions (Leuven: Katholieke Universiteit Leuven)
[5] Chao Li C Z W L Z 2011 A fast algorithm of simulating star map for star sensor International Conference on Computer and Network Technology, kanyakumari
[6] Tappe J A 2009 Development of Star Tracker System for Accurate Estimation of Spacecraft Attitude (Monterey: Naval Postgraduate School)
[7] Craig J L C, Cole L 2006 Fast Star-Pattern Recognition Using Planar Triangles Journal of Guidance, Control, and Dynamics 29 pp 64-71
[8] Ardi N S 2018 Comparison of Various Star Pattern Recognition Algorithm for Star Sensor using Digital Simulation [Undergraduate Thesis] (Bandung: Institut Teknologi Bandung)
[9] Zhang G 2017 Star Identification Methods, Techniques, and Algorithms (Berlin: Springer)
[10] Saifuldin M A 2016 The Making of Star Catalogue for Star Sensor Applications using the Magnitude Filtering Method Development of Operating Systems and Satellite Subsystems at the Satellite Technology Center Indonesia pp 38-52