TRANSPORT PROBLEMS 2026 Volume 21 Issue 1
PROBLEMY TRANSPORTU DOI: 10.20858/tp.2026.21.1.06

Keywords: celestial navigation; starry sky simulation; attitude estimation error

Sergiy IVANOV[1]*, Pavlo OLIYNYK[2]

DEVELOPMENT OF A SKY SIMULATION ALGORITHM FOR THE CALIBRATION OF STAR DETECTORS AND ANALYSIS OF ATTITUDE ESTIMATION ERRORS

Summary. This study presents an algorithm for starry sky image synthesis intended for debugging and calibrating star detectors. The proposed algorithm generates images of the starry sky as they would be observed by a star detector mounted on an aircraft or a maritime vessel, taking into account the platform's geographic coordinates, attitude, and velocity. The simulation incorporates the detector's spatial resolution and spectral sensitivity, with particular emphasis on infrared (IR) radiation in J, H, and K photometric bands. An algorithm implemented in MATLAB was used to produce synthetic starry sky images, which were subsequently processed using star identification and attitude determination methods in order to ensure that simulated images can be used in practice. Next, a comparative analysis of attitude estimation errors was conducted for several commonly employed attitude determination algorithms, providing quantitative insights into their performance under simulated observational conditions.

1. INTRODUCTION

An important element of modern navigation systems is the star detector, which is stabilized by a system based on fiber-optic gyroscopes. Starry sky simulators are used for the debugging, calibration, and testing of star detectors, telescopes, etc. These devices create images of the sky by considering the physical laws of radiation, absorption, and the scattering of light. From the point of view of all-weather navigation, it is important to create simulators for the infrared (IR) radiation range, since the atmosphere is transparent to IR radiation in certain frequency bands (namely, J-, H-, and K-bands).

The classic approach to creating a starry sky simulator is to create star models on a sphere using LEDs and install the star detector inside the sphere on a mechanical turntable [1]. Such a simulator does not allow one to simulate image noise or the influence of the atmosphere, so modern starry sky simulators use computer image synthesis based on the use of star catalogs, such as 2MASS (Two Micron All-Sky Survey) [2] and Gaia (Global Astrometric Interferometer for Astrophysics) [3].

In [4], a simulation of a starry sky image was performed, and the signal-to-noise ratio and the impact of object motion and vibration on the image were evaluated. The disadvantage of the study [4] is that only the J-band range was used.

Authors of [5] used the star color index and the theory of blackbody radiation when modeling the image of the starry sky. According to [5], the accuracy of radiation intensity modeling increased by 5-15%. However, the model proposed in [5] does not take into account the influence of the movement of the star detector platform and vibration.

[1] State University of Information and Communication Technologies; Solomyanska 7, 03110 Kyiv, Ukraine; e-mail: s.ivanov@duikt.edu.ua; orcid.org/0000-0003-3001-2451
[2] State University of Information and Communication Technologies; Solomyanska 7, 03110 Kyiv, Ukraine; e-mail: poleinik@ukr.net; orcid.org/0000-0002-9481-0551
* Corresponding author. E-mail: s.ivanov@duikt.edu.ua

In [6], a starry sky simulator and a platform for verifying star detectors that allow one to model the starry sky are presented. A disadvantage is that the influence of vibration and temperature effects was not analyzed.

In [7], an algorithm for simulating the starry sky was proposed that takes into account degradation models of the points representing the stars, an image noise model, the model of uneven illumination, and the perturbations of the star detector position. Simulations made in [7] showed similarities with real images, but the model does not take the overlapping of star images into account. It is also necessary to conduct a detailed analysis of the image noise.

In [8], an analysis of the influence of high-temperature and high-speed fields on the image of the starry sky during hypersonic flight (speeds from Mach 6 to 16) was conducted. The error of the simulation compared to the experiment was 30%, which is insufficient for practical use.

Thus, the development of the starry sky simulator is a relevant task. The key element of such a simulator is the image synthesis algorithm.

Many algorithms were developed to identify stars and determine vehicle attitude [9, 10, 11]. These algorithms are based either on pattern recognition or estimations of angular distances between stars and a comparison of the obtained subgraph with an on-board star catalog database. Most known algorithms require at least five stars to be present in the star detector's field of view. However, the identification of stars itself is not sufficient for the solution of the navigation problem or attitude estimation. In order to estimate the attitude of the vessel, one has to determine its attitude matrix relative to some base coordinate system. In most cases, this is done by estimating the matrix using vectors measured in base (reference) and body (vehicle) frames (i.e., solving Wahba's problem). Wahba's problem may be solved using various algorithms [12, 13]: singular value decomposition (SVD), Davenport's q method, quaternion estimator (QUEST), the estimators of the optimal quaternion (ESOQ and ESOQ2), quaternion-based semidefinite relaxation for robust alignment (QUASAR), fast global registration (FGR), and the RANSAC algorithm. One of the fastest attitude estimation algorithms is the TRIAD algorithm.

In order to estimate the efficiency of the starry sky simulation algorithm, e.g. proposed in the paper, attitude estimation algorithm should be used. If the attitude set during the simulation of the sky image and attitude, calculated using the synthesized images, shows good correspondence, then the simulation algorithm is suitable for practical use.

The goals of the present study are to create an image synthesis algorithm for a starry sky simulator in the IR range and to test the possibility of using the synthesized image for setting up a star detector.

2. ALGORITHM DESCRIPTION AND TESTING

2.1. Algorithm description

As stated above, it is important to create starry sky simulators for the IR radiation range. This is related to the fact that the Earth's atmosphere contains water vapor and carbon dioxide. Both these substances absorb visible light, and when the sky is cloudy, stars are invisible. Near infrared radiation, however, is not attenuated by the atmosphere as strongly as visible light is, in at least three ranges:

- J-band – wavelength of 1.1 to 1.4 μm
- H-band – wavelength of 1.5 to 1.8 μm
- K-band – wavelength of 2.0 to 2.4 μm

Fig. 1 shows the relative intensity of radiation transmitted through the Earth's atmosphere vs. the wavelength for J-, H-, and K-bands. Spectral transmission profiles were computed using a MATLAB simulation that incorporates H_2O and CO_2 absorption. As can be seen in Fig. 1, in the J-band (1.1-1.4 μm), radiation undergoes minimal attenuation near the central wavelength, making this range highly favorable for ground-based observations and navigation. In contrast, the H-band (1.5-1.8 μm) experiences significant atmospheric absorption, primarily due to water vapor, resulting in a reduction

of transmitted intensity by approximately 36%. Radiation in the K-band (2.0-2.4 μm) is strongly attenuated because of combined absorption by water vapor (H_2O) and carbon dioxide (CO_2), rendering this band impractical for navigation and attitude determination under typical atmospheric conditions. The simulation results are in good agreement with recent studies on atmospheric transparency in the near-infrared [14-16] and are consistent with the HITRAN molecular spectroscopic database [17]. Taken together, these findings emphasize the suitability of the J-band for applications requiring the reliable transmission of radiation through the Earth's atmosphere while highlighting the limitations of the H-band and K-band for practical navigation and remote sensing.

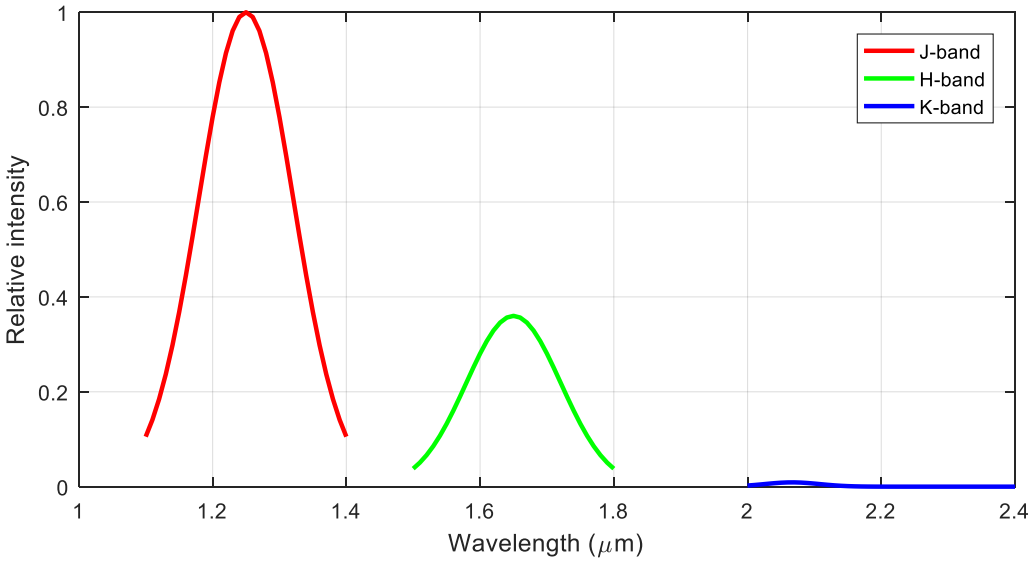

Fig. 1. Relative intensity of radiation vs. wavelength for J-, H-, and K-bands of near-IR radiation

In our starry sky simulator, J- and H-bands were used, as the Earth's atmosphere is maximally transparent to radiation in those frequency bands.

To synthesize an image of the starry sky suitable for adjustment of the star detector, an algorithm consisting of the following steps is proposed:

- Determination of the area of the sky covered by the star detector's field of view based on the coordinates of the observation site (latitude and longitude), altitude above sea level, observation time (UTC), azimuth and elevation angles, which determine the orientation of the star detector, as well as the star detector's angle of view.
- Selection of stars from the star catalog that are in the field of view (at least five stars are required to ensure correct identification and calibration).
- Generation of star images, taking their temperature and radiation spectrum, as well as stellar magnitude (which determine the radiation range and brightness of the star) into account.
- Generation of starry sky illumination and background noise.
- Taking atmospheric effects–primarily, light attenuation and image blurring–into account in order to increase the realism of images.
- Taking the resolution of the star detector (which determines the detail of images) and its spectral sensitivity to each radiation frequency range into account.

The proposed algorithm will allow one to check the operation of the device in conditions that are as close as possible to real ones. Generated images of a section of the starry sky will be implemented in a projector, which can be an optical, infrared, or combined device.

2.2. Starry sky synthesis modeling

As an example, let's consider the generation of a starry sky image that will be perceived by the star detector's sensor with a 512 × 512 pixel matrix and a pixel size of 10 × 10 μm. The image generation algorithm is modeled in MATLAB and provides a selection of stars with a stellar magnitude $m \leq 2$ from the catalog, taking into account background noise and the influence of the atmosphere. The influence of the atmosphere is simulated by applying a Gaussian blur with $\sigma = 2$. The simulated image of the starry sky is shown in Fig. 2. As can be seen in Fig. 1, due to the blurring, the stars in the image are shown not as bright points but as blurred circles. Accordingly, to identify stars and determine their coordinates, one will have to determine the centers of the characteristic (brightest) stars.

The sky image synthesis algorithm, in its basic form, is as follows:

```
% Create image matrix star_sky
star_sky = background_intensity * ones(image_size, image_size);  % Background

% Add stars to the image with the given intensity
for i = 1:num_stars
   star_sky(x(i), y(i)) = star_sky(x(i), y(i)) + intensity(i);
   if widestars
      if (x(i)-1<=0) || (y(i)-1<=0)
         continue;
      end
      if (x(i)+1>image_size) || (y(i)+1>image_size)
         continue;
      end
      star_sky(x(i)-1, y(i)-1) = star_sky(x(i)-1, y(i)-1) + intensity(i);
      star_sky(x(i)  , y(i)-1) = star_sky(x(i)  , y(i)-1) + intensity(i);
      star_sky(x(i)+1, y(i)-1) = star_sky(x(i)+1, y(i)-1) + intensity(i);

      star_sky(x(i)-1, y(i)) = star_sky(x(i)-1, y(i)) + intensity(i);
      star_sky(x(i)+1, y(i)) = star_sky(x(i)+1, y(i)) + intensity(i);

      star_sky(x(i)-1, y(i)+1) = star_sky(x(i)-1, y(i)+1) + intensity(i);
      star_sky(x(i)  , y(i)+1) = star_sky(x(i)  , y(i)+1) + intensity(i);
      star_sky(x(i)+1, y(i)+1) = star_sky(x(i)+1, y(i)+1) + intensity(i);
   end
end

% Add blur
star_sky = imgaussfilt(star_sky, atmospheric_blur);

% Visualization
figure;
imshow(star_sky, []);
```

In Fig. 3, the result of MATLAB processing of the starry sky image is shown, including characteristic stars identified using the nearest neighbor method.

The image in Fig. 3, which was obtained using the algorithm described above, allows one to identify characteristic stars and obtain their coordinates with an accuracy of 3–5 pixels (i.e., the synthesized image is suitable for setting up a star detector).

It follows from the Nyquist theorem that it is necessary to ensure that sky images are generated with a frequency at least twice as high as the refresh rate of images from the CCD/CMOS sensor of the star detector in order to take the motion of the star detector into account. If the synthesis of a starry sky image with a given quality in real time is impossible, then one may synthesize images and play video during the adjustment process.

If a star detector with field stabilization is adjusted, the proposed algorithm can be used to simulate the residual drift of the instrument. In the case of vibrations, micro-shifts of stars can be added to each frame. In addition, the intermediate positions of stars can be interpolated in intermediate frames in order to reduce the frame creation time.

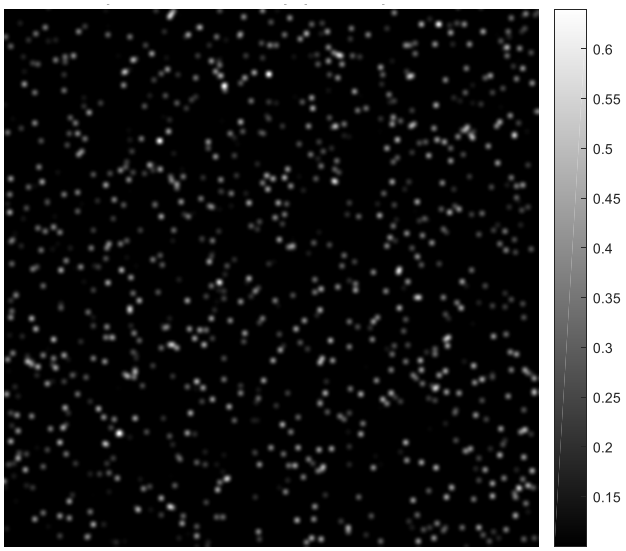

Fig. 2. Modeled starry sky image with atmosphere effects (blurring and attenuation; intensity scale is shown at right)

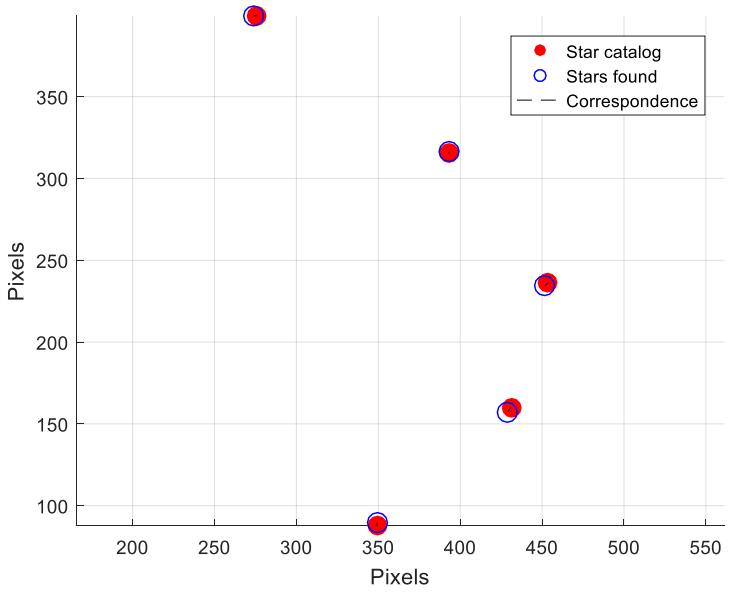

Fig. 3. Starry sky image processed using the nearest neighbor method with characteristic stars selected

3. STAR IDENTIFICATION AND ESTIMATION OF ATTITUDE

Star identification and attitude estimation are related processes, as the latter requires the former. The following algorithm was used as the basis for star identification:

- An image obtained from the star detector (in case of synthesis algorithm testing, this is a synthesized image of the sky) is converted to grayscale and then to black-and-white using adaptive Otsu binarization [18] with lowered threshold.

- The black-and-white image is cleared of small objects (stars with high magnitude numbers and noise).
- Stars are selected as circle-shaped regions using the Hough gradient method, or the centers of the stars are estimated as the gravity centers of their respective regions on the black-and-white image.
- The star is identified using the nearest neighbor method.
  The nearest neighbor method is implemented as follows:
- Equatorial coordinates (right ascension and declination) of each detected star are converted into standard J2000 format, taking precession and the movement of the star into account, and the IAU 2000/2006 model is used for correction.
- For each star detected, the nearest star from the catalog is found using the spherical distance (calculation using the Haversine formula or cosine rule).

A star is assumed to be identified if the distance between the detected star on the image and its place according to the star catalog is less than an allowable error. If there are multiple star candidates, then photometric characteristics (star magnitude) are taken into account. The brightest star according to the catalog is assumed to be the star on the image, obtained by a star detector. In ambiguous cases, the color indices of the stars are used to select a corresponding star; weight coefficients and threshold levels may also be used.

The star identification algorithm was evaluated using the 2MASS All-Sky Point Source Catalog (PSC) [2] with the metadata current as of 11 April 2025. Stars brighter than a magnitude of 2 in the J- or H-band were excluded to match the sensor's dynamic range. For illustrative purposes, a representative test image (Fig. 4) was acquired in Stuttgart, Germany, using an IMX990 SWIR camera (1280 × 1024 px, 0.9–1.7 μm spectral range, 12 mm focal length, 30° FOV, thermoelectrically cooled) with a 50-ms exposure. The image was preprocessed to ensure its suitability for star detection; this included dark-frame subtraction, flat-field correction, noise suppression, background subtraction, and intensity normalization.

Stars were detected using Otsu's global thresholding, with the initial MATLAB gray threshold reduced by 5% to improve sensitivity to fainter stars. Threshold adjustment was based on preliminary experiments conducted to balance detection efficiency and false-positive rate. Performance metrics, including true positive rate, false positive rate, and precision, were calculated relative to the reference catalog. The MATLAB implementation is provided for reproducibility.

```
% 1. Load image
RGB = imread(filename);
figure, imshow(RGB), title('1. Source image');

% 2. Convert to grayscale
gray = rgb2gray(RGB);
figure, imshow(gray), title('2. Grayscale image');

% 3. Otsu binarization
T = graythresh(gray);
deltaT = 0.05;
T_adj = max(0, T - deltaT); % Limit T in order to ensure >= 0
bw = imbinarize(gray, T_adj);
figure, imshow(bw), title('3. Otsu binarization result');

% 4. Delete small objects (e.g., less than 5 pixels)
bw_clean = bwareaopen(bw, 5);
figure, imshow(bw_clean), title('4. Cleared image with no small objects');

% 5. Detection of star centers
stats = regionprops(bw_clean, 'Centroid');
centroids = cat(1, stats.Centroid);
% Show result
figure, imshow(gray), title('5. Detected star centers');
```

```
hold on;
plot(centroids(:,1), centroids(:,2), 'ro');
hold off.
```

For the nearest-neighbor method, the allowable error threshold for the spherical distance was set at 0.5° to account for positional uncertainties. When two or more candidate stars were available, both catalog magnitude and color index (dominating wavelength) were considered, each with a weight coefficient of 0.5. Fig. 4 shows that at least 10 stars from the catalog were successfully identified using this algorithm. In practice, the error threshold and weight coefficients may need to be adjusted depending on sensor characteristics and observational conditions.

Fig. 4. Stars identified using the described algorithm

In addition to the nearest neighbor method, the triangle method can be used for star identification, and it is less sensitive to noise and partial visibility of stars. The algorithm of the method is as follows:

- For each possible combination of three stars, calculate the distances between those stars and the angles between sides of the triangle with vertices in those stars.
- For each triangle, calculate the coordinates of the triangle to obtain a system of related distances and angles.
- Using triangle geometry and star coordinates, compare stars with a star catalog either by using the nearest neighbor method or by looking for similar triangles, and identify them.

The following numeric experiment was performed to check the suitability of the synthesized images to debug and test star detectors. For a given latitude, longitude, and height above sea level, the vectors of the positions of base stars in the reference frame (i.e., the Earth-based coordinate system) were calculated. Next, the pitch θ, roll φ, and yaw ψ of the star detector changed with time t by following certain laws (all angles are expressed in degrees).

$$\begin{cases} \psi = 20 + 5\sin(0.5t) \\ \theta = 10 + 3\sin(0.7t) \\ \varphi = 15 + 2\sin(0.9t) \end{cases} \quad (1)$$

The images of the starry sky were synthesized with noise added. Stars were identified, and the attitude of the star detector was estimated using the TRIAD algorithm [19, 20], as it is the fastest available algorithm.

The TRIAD algorithm is conceptually straightforward and relies on the use of two known vector observations in the inertial and body reference frames. Specifically, given the celestial coordinates of two stars in the inertial frame—right ascension (RA) and declination (DE)—and their corresponding angular positions in the body frame—elevation (E) and azimuth (A)—two sets of vectors vI_i and vB_i are constructed:

- For the reference frame (i.e., ECI [Earth-Centered Inertial]) coordinate system,

$$vI_i = \begin{bmatrix} \cos(DE_i)\cos(RA_i) \\ \cos(DE_i)\sin(RA_i) \\ \sin(DE_i) \end{bmatrix}. \tag{2}$$

- For the body frame,

$$vB_i = \begin{bmatrix} \cos(E_i)\cos(A_i) \\ \cos(E_i)\sin(A_i) \\ \sin(E_i) \end{bmatrix}, \tag{3}$$

where i$\in\{1,2\}$ denotes the star index. Vectors vI_i and vB_i are computed according to Equations (2) and (3), respectively, and are normalized to ensure unit magnitude. These vectors form the basis for constructing orthonormal frames, which are then used to estimate the attitude rotation matrix.

Next, two orthonormal coordinate frames (triads) are constructed—one in the inertial frame and one in the body frame—with corresponding basis vectors denoted as eI_i and eB_i:

$$eI_1 = vI_1,\ eI_2 = \frac{eI_1 \times vI_2}{\|eI_1 \times vI_2\|},\ eI_3 = \frac{eI_1 \times eI_2}{\|eI_1 \times eI_2\|}, \tag{4}$$

$$eB_1 = vB_1,\ eB_2 = \frac{eB_1 \times vB_2}{\|eB_1 \times vB_2\|},\ eB_3 = \frac{eB_1 \times eB_2}{\|eB_1 \times eB_2\|}. \tag{5}$$

The transformation matrix between these triads is computed using the following expression:

$$C = \begin{bmatrix} eB_1 & eB_2 & eB_3 \end{bmatrix} \cdot \begin{bmatrix} eI_1 & eI_2 & eI_3 \end{bmatrix}^T. \tag{6}$$

Thus, the attitude of the body (vehicle) frame relative to the inertial frame is defined by the rotation matrix C, which is a proper orthogonal 3×3 matrix with unit determinants, representing a rigid-body rotation in the three-dimensional space.

The results of the yaw, pitch, and roll estimation (i.e., the expected and calculated angles) are presented in Fig. 5. The expected yaw, pitch, and roll angles were computed using Equation (1), while the calculated angles were derived from the rotation matrix obtained according to Equation (6).

As can be seen in Fig. 5, the expected attitude angles, set up according to (1), and the calculated ones, determined using the TRIAD algorithm, are close. The maximal absolute error of angle estimation is 0.1314° for yaw, 0.1516° for pitch, and 0,1599° for roll angle.

The TRIAD algorithm, although computationally efficient, can exhibit instability in the presence of measurement noise because only two stars are used for attitude estimation. Furthermore, it does not utilize any additional information, such as the coordinates of other visible stars. Therefore, in practical applications, several alternative algorithms are used to improve the robustness and accuracy of attitude determination; among the most widely used are the SVD, QUEST, and FGR algorithms.

The SVD algorithm solves Wahba's problem by utilizing SVD. It determines the rotation matrix C that minimizes the sum of squared errors between the observed star vectors in the body frame, rotated by C, and the corresponding reference vectors in the inertial frame:

$$\sum \|vI_i - C \cdot vB_i\|^2 \to \min, \tag{7}$$

where $i \in [1; N]$ is the index of a star.

When calculating the rotation matrix, C, it is expressed in the form of SVD as

$$C = U \cdot \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{bmatrix} \cdot V^T, \tag{8}$$

where $s_1 = 1$, $s_2 = 1$, $s_3 = \left| U \cdot V^T \right|$, V, and U are the orthogonal matrices that are obtained by calculating the SVD of the matrix B defined by the formula $B = U \cdot X \cdot V^T$. Matrix B is expressed as follows:

$$B = \sum_i vB_i^T \cdot vI_i, \tag{9}$$

where $i \in [1; N]$ is the index of a star, and N is the number of stars used. During calculations, matrix X is dropped.

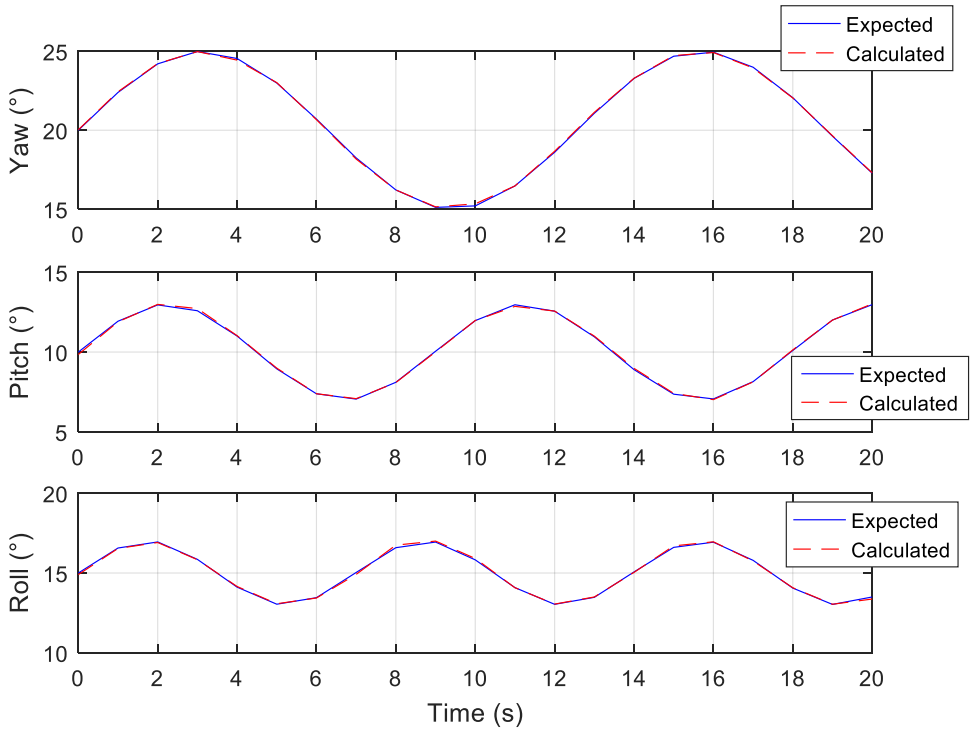

Fig. 5. Expected and calculated yaw, pitch, and roll

As can be seen from (9), the SVD algorithm, unlike the TRIAD algorithm, uses all available information about stars and, therefore, is more stable when noise is present and is tolerant to small measurement errors. Algorithms of matrix SVD calculation are robust. The SVD calculation algorithm is described, for example, in [21]. Attitude calculations using the SVD algorithm require more time and computational power than those using the TRIAD algorithm.

The QUEST algorithm is the most widely used algorithm for solving Wahba's problem. Its basic idea is to find a unity quaternion that describes the rotation between inertial and body frames optimally in the least-square sense. To that end, matrix B is first calculated using (9), and then matrices

$$S = B + B^T, \tag{10}$$

$$Z = \begin{bmatrix} B_{2,3} - B_{3,2} \\ B_{3,1} - B_{1,3} \\ B_{1,2} - B_{2,1} \end{bmatrix}, \tag{11}$$

and

$$K = \begin{bmatrix} S - tr(B)I & Z \\ Z^T & tr(B) \end{bmatrix}, \tag{12}$$

where I is the 3×3 identity matrix, are calculated.

Next, the maximum eigenvalue of matrix K $\lambda_{\max}$ and the corresponding eigenvector q is calculated, often using numerical methods.

After that, using the designations

$$qv = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}, \; qw = q_4, \tag{13}$$

where q_k are eigenvector's components and $k \in [1;4]$, matrix C is evaluated as follows:

$$C = \left(qw^2 - qv^T \cdot qv\right)I + 2qv \cdot qv^T - 2qw \begin{bmatrix} 0 & -qv_3 & qv_2 \\ qv_3 & 0 & -qv_1 \\ -qv_2 & qv_1 & 0 \end{bmatrix}. \tag{14}$$

When the value of C is obtained, the QUEST algorithm operation is finished.

The QUEST algorithm is fast and effective for problems with three or more stars. This is the algorithm of choice for high-precision attitude estimation and navigation systems; its drawback is the need to calculate eigenvalues, which may be computationally costly.

The FGR algorithm is the method of gradient attitude estimation. The basic idea of the algorithm is to minimize the difference between rotated body vectors and inertial system vectors of stars using a simple rotation correction at each iteration step.

The FGR algorithm is implemented as follows. First, the rotation matrix C is set to be equal to the identity matrix I (no rotation). Next, in each iteration of the algorithm, vector Ω is calculated as

$$\Omega = \frac{1}{N} \sum_i \left(\frac{C \cdot vI_i^T}{\left\| C \cdot vI_i^T \right\|} \right) \times \left(\frac{vB_i^T}{\left\| vB_i^T \right\|} - \frac{C \cdot vI_i^T}{\left\| C \cdot vI_i^T \right\|} \right), \tag{15}$$

and angle α is calculated as the Euclidean norm of that vector:

$$\alpha = \|\Omega\|. \tag{16}$$

If the value of α is less than a predefined tolerance, the calculations are finished, and the current value of matrix C is assumed to be the FGR algorithm's result.

Next, if the calculations are not finished, vector Ω is normalized, and vector A is obtained as

$$A = \frac{\Omega}{\alpha}. \tag{17}$$

Based on that vector, matrix L is calculated as follows:

$$L = \begin{bmatrix} 0 & -A_3 & A_2 \\ A_3 & 0 & -A_1 \\ -A_2 & A_1 & 0 \end{bmatrix}, \tag{18}$$

and matrix C is calculated using the formula

$$C = \left(I + \sin(\alpha)L + \left(1 - \cos(a)\right)L^2\right) \cdot C. \tag{19}$$

After the last step presented above, the FGR algorithm continues calculations starting from Formula (15).

The key advantage of the FGR algorithm is its simplicity: one does not have to perform SVD or calculate eigenvalues and eigenvectors of a matrix. Thus, the FGR algorithm can be used on devices with low computing power (e.g., MCUs). The drawback of the FGR algorithm is common to all

gradient-based optimization methods—namely, it has a rather slow convergence speed near the minimal value of the objective function. Therefore, as an additional calculation termination condition, a maximum number of iterations (for example, 100 iterations) is often used.

In the FGR algorithm, an initial estimate of the rotation matrix C, obtained from any other attitude determination method (e.g., SVD, QUEST, TRIAD), may be used instead of the identity matrix. This often improves convergence and reduces the number of iterations required.

To ensure that matrix C remains a proper rotation matrix ($C \in SO(3)$) during FGR iterations, orthogonality correction can be applied at each step using SVD. The current estimate of C, calculated using (19), is decomposed as

$$C = U \cdot D \cdot V^T, \tag{20}$$

where U and V are orthogonal matrices, and D is a diagonal matrix. The corrected matrix is computed as follows to enforce orthogonality and ensure that $|C| = 1$:

$$C_\perp = U \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \left|U \cdot V^T\right| \end{bmatrix} \cdot V^T. \tag{21}$$

Next, matrix C is assumed to be equal to the corrected matrix—that is,

$$C = C_\perp. \tag{22}$$

After orthogonality correction is applied, the algorithm continues its operation, starting with Formula (15).

The use of SVD for orthogonality correction ensures that the matrix C remains a valid rotation matrix—that is, that matrix C has the following characteristics:

- It is orthogonal: $C \cdot C^T = 1$. There is no distortion or scaling.
- It is proper: $|C| = 1$. There is no reflection or mirroring.

This keeps the orientation estimate physically correct and numerically stable throughout FGR iterations.

The following numerical experiment was carried out to estimate the influence of sensor noise on the precision of attitude estimation. The vehicle was assumed to be stationary and situated in Stuttgart (λ=48.78° N, φ=9.18° E), and the local sidereal time is 50.21°. Fourteen stars that are very bright in near IR were used as base stars: Betelgeuse, Aldebaran, Arcturus, Vega, Capella, Rigel, Procyon, Deneb, Altair, Spica, Pollux, Fomalhaut, Regulus, and Adhara.

For the body frame (vehicle), its attitude relative to the inertial frame was set as axis-angle rotation (i.e., as a rotation axis vector and the angle of rotation around that axis). That rotation was used as a reference for attitude error estimation. Gaussian noise (angular measurement error), with σ=0.3°, is added to each star vector measurement in the body frame using Rodrigues' rotation formula:

$$vB_i = \left(R \cdot {vB_i}^T\right)^T, \tag{23}$$

where R is defined by

$$R = I + \sin(\sigma)K + \left(1 - \cos(\sigma)\right)K^2, \tag{24}$$

where K is a skew-symmetric matrix for axis-angle rotation,

$$K = \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}, \tag{25}$$

and r_1, r_2, r_3 are Gaussian-distributed random numbers that denote normalized rotation axis vector elements.

Based on these data, for a given set of stars, attitude was estimated using four algorithms: TRIAD, SVD, QUEST, and FGR. Matrix C was calculated using Formulae (6), (8), (14), and (15)–(22). Invisible stars (i.e., those below the horizon) were excluded from calculations. One of each pair of stars that were closer than 10° to one another was excluded as well. In the SVD, QUEST, and FRG algorithms, all visible stars were used to estimate matrix C; in the TRIAD algorithm, only the first two visible stars were used.

The following calculations were done to estimate the sensitivity of attitude estimation to noise. For a given noise level, 40 trial calculations were done for each algorithm, and for each algorithm, the mean attitude estimation error err was calculated as the angle between the given and estimated transformation matrices. That angle represents the smallest rotation required to align the two coordinate frames, and is calculated using the following formula:

$$err = \arccos\left(\frac{tr\left(C^T C_t\right)-1}{2}\right), \tag{26}$$

where C_t is the transition matrix between the body and reference (inertial) frames when no noise is present. All calculations were performed in MATLAB.

Fig. 6 shows the vectors of stars in the inertial frame and in the body frame obtained during modeling.

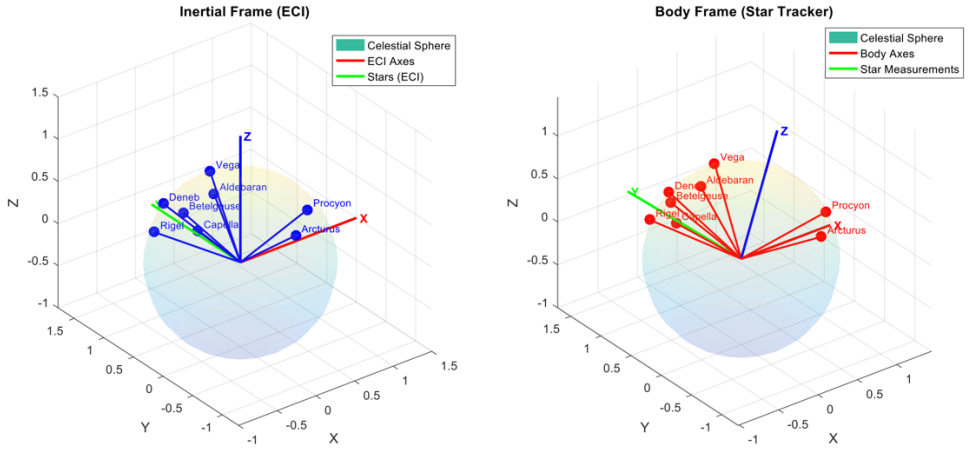

Fig. 6. Vectors of stars in the inertial and in the body frame

Fig. 7 is a plot of the attitude estimation error vs. the measurement number for each tested algorithm.

As can be seen in Fig. 7, the SVD, QUEST, and FGR algorithms yield comparable attitude estimation errors (from 0.05–0.3°). The TRIAD algorithm gives attitude estimation errors from 0.1–1.6°, which is significantly higher than those obtained for the other algorithms. However, all error values shown in Fig. 7 are acceptable for most attitude estimation systems used in air and sea vehicles. Thus, images may be used for the debugging, calibration, and testing of star detectors when synthesized with the proposed algorithm. The SVD, QUEST, or FGR algorithms may be used as a base algorithm for attitude determination.

4. CONCLUSIONS

This paper presented a novel algorithm for synthesizing a realistic starry sky within the field of view of a star detector, specifically designed for the infrared spectrum. The primary aim of this work was to develop a comprehensive model that integrates critical physical and instrumental factors. The algorithm operates in the J and H atmospheric bands and takes advantage of their maximal transparency. It also consciously accounts for atmospheric effects—namely, light attenuation and image blurring—coupled with the detector's own parameters, including its resolution and spectral

sensitivity. Numerical simulation in a MATLAB environment confirmed the algorithm's practical viability. The results demonstrate that the synthesized images are suitable for use in an infrared starry sky simulator and that they are effective for the calibration and tuning of star detectors.

Regarding practical implementation, this study addresses the critical problem of detector motion simulation. If motion is to be accurately simulated, sky images must be generated at a frequency at least twice the refresh rate of the detector's CCD/CMOS sensor. Furthermore, the proposed framework offers a flexible solution for handling computational constraints: if real-time synthesis is impossible, pre-rendered video sequences can be employed. The model's fidelity is further enhanced by its capacity to simulate platform-specific disturbances; for example, micro-shifts of stars can be injected into each frame to mimic vibrations and residual drift.

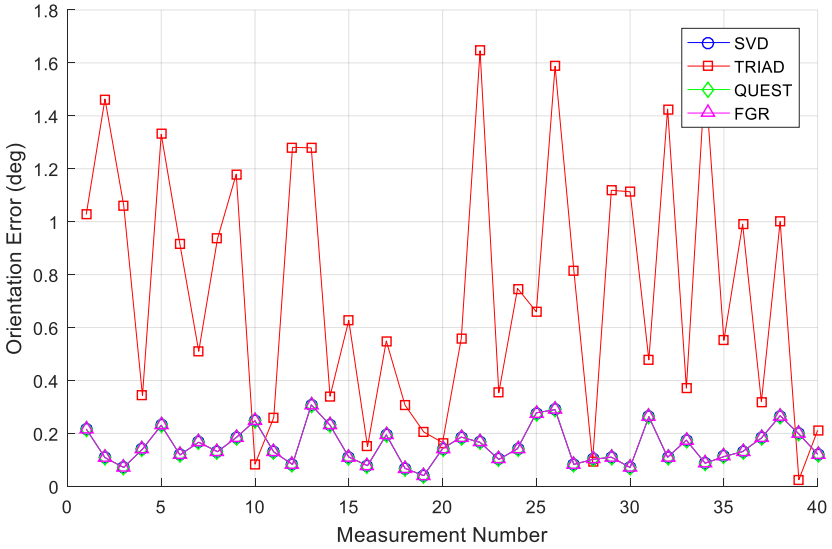

Fig. 7. Attitude estimation error vs. measurement number

A comparative analysis of four attitude estimation algorithms (TRIAD, SVD, QUEST, and FGR) was conducted to guide their implementation. A numerical experiment, which introduced Gaussian noise (σ=0.3°) to star vector measurements, revealed that the SVD, QUEST, and FGR algorithms yield comparable low estimation errors (0.05°–0.3°), while the TRIAD algorithm exhibited a significantly higher error (0.1°–1.6°), primarily because it relies on only two measurement vectors. Therefore, the SVD, QUEST, and FGR algorithms are recommended as the superior choice for precise attitude determination within this system. The present study was conducted within a computational simulation framework, and a robust proof-of-concept was established. Its practical implementation may be affected by hardware-specific issues, which can influence image synthesis and processing. Future work will focus on enhancing the real-time performance of image synthesis through implementation on GPU and FPGA architectures, quantitative evaluations of image realism through comparison with empirical sensor data, and improvements to model representativeness by incorporating more complex noise and vibration profiles.

References

1. Boone, B.G. & Bruzzi, J.R. & Dellinger, W.F. et al. Optical simulator and testbed for spacecraft star tracker development. In: *Proceedings Volume 5867, Optical Modeling and Performance Predictions II.* 2005. Vol. 5867. P. 586711-1 – 586711-14. DOI: 10.1117/12.619133.
2. *IRSA Viewer.* Available at: https://irsa.ipac.caltech.edu/irsaviewer/?__action=layout.showDropDown&view=IrsaCatalog.
3. *Gaia archive*. Available at: https://gea.esac.esa.int/archive/.

4. Zheng, X. & Shen, J. & Wei, Z. & Wang, H. Star map simulation and platform influence of airborne star sensor based on J-band data of 2MASS catalog. *Infrared Physics & Technology.* 2020. Vol. 111(12). P. 103541.1-103541.10. DOI: 10.1016/j.infrared.2020.103541.
5. Wang, H. & Yan, Z. & Mao, X. et al. A new high-precision star map simulation model and experimental verification. *Journal of Modern Optics.* 2021. Vol. 68(16). P. 856-867. DOI: 10.1080/09500340.2021.1955165.
6. Schulz, V.H. & Marcelino, G.M. & Seman, L.O. et al. Universal verification platform and star simulator for fast star tracker design. *Sensors.* 2021. Vol. 21(3). P. 907-930. DOI: 10.3390/s21030907.
7. Yang, H. & Jin, Y. & Hu, Y. & Zhang, D. et al. Image degradation model for dynamic star maps in multiple scenarios. *Photonics.* 2022. Vol. 9(10). P. 673-686. DOI: 10.3390/photonics9100673.
8. Liu, H.-N. & Sun, T. & Wu, S.-Y. et al. Development and validation of full-field simulation imaging technology for star. *Optics Express.* 2025. Vol. 33(3). P. 6526-6542. DOI: 10.1364/OE.550772.
9. Spratling, B.B. & Mortari, D. A Survey on Star Identification Algorithms. *Algorithms. 2009.* Vol. 2. P. 93-107. DOI: 10.3390/a2010093.
10. Rijlaarsdam, D. & Yous, H. & Byrne, J. et al. A survey of lost-in-space star identification algorithms since 2009. *Sensors.* 2020.Vol. 20(9). P. 2579.1-2579.18. DOI: 10.3390/s20092579.
11. Toloei, A. &·Zahednamazi, M. & Ghasemi, R. &·Mohammadi, F. Comparative analysis of star identification algorithms. *Astrophysics and Space Science.* 2020. Vol. 365(4). P. 63.1-63.9. DOI: 10.1007/s10509-020-03775-9.
12. Markley, F.L. & Mortari, D. How to estimate attitude from vector observations. In: *Astrodynamics Specialist Conference*. Alaska, Girdwood. 1999. Available at: https://ntrs.nasa.gov/citations/19990104598.
13. Wang, Y. A comparative study of algorithms for the Wahba problem and the proposal of a new algorithm. *Operations Research and Fuzziology*. 2025. Vol. 15(2). P. 218-226. DOI: 10.12677/orf.2025.152078.
14. Elsey, J. & Coleman, M.D. & Gardiner, T.D. et al. Atmospheric observations of the water vapour continuum in the near infrared windows between 2500 and 6600 cm^{-1}. *Atmospheric Measurement Techniques.* 2020. Vol. 13. P. 2335-2350. DOI: 10.5194/amt-13-2335-2020j.
15. Wu, P. & Shan, C. & Liu, C. et al. Ground-based remote sensing of atmospheric water vapor using high-resolution FTIR spectrometry. *Remote Sensing*. 2023. Vol. 15. No. 3484. DOI: 10.3390/rs15143484.
16. Mukherjee, L. & Wu, D.L. & Abuhassan, N. et al. Twilight near-infrared radiometry for stratospheric aerosol layer height. *Remote Sensing*. 2025 Vol. 17. No. 2071. DOI: 10.3390/rs17122071
17. Rothman, L. S. & Gordon, I. E. & Babikov, Y. et al. The HITRAN2012 molecular spectroscopic database. *Journal of Quantitative Spectroscopy & Radiative Transfer*. 2013. Vol. 130. P. 4-50. DOI: 10.1016/j.jqsrt.2013.07.002.
18. Vala H.J. & Baxi, A. A review on Otsu image segmentation algorithm. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET).* 2013. Vol. 2(2). P. 387. DOI: 10.4028/www.scientific.net/AMR.989-994.1959.
19. Black, H. A passive system for determining the attitude of a satellite. *AIAA Journal.* 1964. Vol. 2(7). P. 1350-1351.
20. Martinez, Genaro, J. & Morales, R. Implementing the TRIAD algorithm for nanosatellite attitude using NUCLEO-F303RE microcontroller: Contributing to Mexico's space initiatives. *Journal of Physics: Conference Series.* 2024. Vol. 2804. No. 012010. DOI: 10.1088/1742-6596/2804/1/012010.
21. Higham, N.J. & Mary, T. Mixed precision algorithms in numerical linear algebra. *Acta Numerica*. 2022. Vol. 31. P. 347-414. DOI: 10.1017/S0962492922000022.

Received 22.08.2024; accepted in revised form 19.02.2026