Neural Network Calibration of Star Trackers

Shaghayegh Khodabakhshian, *Graduate Student Member, IEEE*, and John Enright, *Member, IEEE*

***Abstract*— To maintain ideal performance, star trackers must be able to predict the direction of incident starlight to within a few arcseconds across the entire instrument field of view (FOV). Parametric camera models are commonly used to calculate star vectors from camera images and correct for aberrations in the instrument optics. This conventional approach can be quite effective, but systematic errors can be difficult to eliminate, and proper selection of calibration basis functions is often difficult to determine. This study explores using supervised machine learning (ML) approaches such as radial basis function networks (RBFNs) and support vector machines (SVMs) for star tracker calibration as an alternative to conventional aberration formulations. These networks can be formulated as either a correction to a low-order camera model or a complete replacement for the whole model. When applied to the instrument calibration of a dozen Sinclair Interplanetary ST-16RT2 sensors, the RBFN formulation offers 27% reduction in the calibration residuals and almost 12% reduction in the validation residuals over conventional formulations.**

***Index Terms*— Camera calibration, neural network, star tracker.**

I. Introduction

STAR trackers have become more compact over the years and current models are commonly found on nanosatellites. As part of their operation, star trackers must take a 2-D image of the stars in the field of view (FOV), calculate the centroid of the star images on the detector, and use this centroid information to calculate the corresponding instrument-frame vector direction to the star. Geometric patterns formed by the brightest stars in the FOV are then compared with the onboard catalog to find a match and compute the attitude (orientation) of spacecraft. Good matching and attitude determination performance depends on accurately calculating these star vectors to within arcseconds or better, across the entire instrument FOV.

The camera calibration models are used to relate 3-D world coordinates to 2-D detector positions. The traditional low-order parametric models capture the effects of optical aberrations using parameterized modifications to ideal pinhole ray-tracing. Some practitioners favor physical models of nominal and aberrant optical phenomena [1]. These models often derive the imaging effects of concrete defects such as lens decentering. Other models forego any physical explanation of the optical aberrations and rely solely on mathematical models of their effects. These include the relatively low-order models of Lopes and Shuster [2] and the medium-order, rational function models (RFMs) popular in photogrammetry [3]. In either case, a suite of calibration images is used to numerically optimize the intrinsic parameters. Although generally effective, sizable systematic errors often remain in the calibration residuals indicating the presence of modeling error.

In contrast to the above techniques, implicit calibration deemphasizes the role of parameterized basis functions and adopts a different approach to relating world and image coordinates. Artificial neural networks (ANNs) have been implemented for implicit camera calibration and shown promising results [4], [5]. The benefit of a nonparametric implicit approach such as radial basis function network (RBFN) over the common low-order parametric calibration model is the lack of need to have a priori knowledge of the true camera model and physical parameters.

This study explores implementation of few supervised learning models including ANNs and support vector machines (SVMs) for calibration of ST-16RT series star trackers. Two different approaches are tested for each supervised machine learning (ML) technique: the purely IMPLICIT approach relies on a shallow neural network or SVM to find a matching from 2-D star image position to the sensor-frame star vector. The second approach, HYBRID, still relies on the conventional pinhole camera model and uses an shallow neural network or SVM to make corrections for remaining errors associated with the basis functions in the camera distortion models. We show that these supervised ML methods are capable of correcting systematic residual errors leftover from the ST-16RT's explicit calibration.

We acknowledge that this application is a somewhat unconventional target for ML—our typical training datasets consist of only a few hundred points, and the embedded nature of the these sensors prioritizes efficient use of storage and processing resources. However, we have found that the ML approaches can be quite effective in practice. By avoiding the need to identify a specific mathematical representation of the sensor behavior, our calibrations significantly outperform the low- and medium-order explicit parametric models. Furthermore, these ML techniques offer similar or better performance to direct interpolation while requiring less data (model parameters) storage. This can have a significant positive effect on the effort needed to calibrate these devices during manufacture.

The remainder of this section provides an overview of prior work and an introduction to the ST-16RT star tracker. We then detail the current parametric camera model and the

Manuscript received 22 July 2022; revised 26 September 2022; accepted 18 October 2022. Date of publication 1 November 2022; date of current version 11 November 2022. This was supported by Rocket Lab through direct and in-kind funding from Sinclair Interplanetary. The Associate Editor coordinating the review process was Dr. Xiangchen Qian. *(Corresponding author: Shaghayegh Khodabakhshian.)*

The authors are with the Department of Aerospace Engineering, Toronto Metropolitan University, Toronto, ON M5B2K3, Canada (e-mail: shaghayegh.khodabakh@ryerson.ca; jenright@ryerson.ca).

Digital Object Identifier 10.1109/TIM.2022.3218556

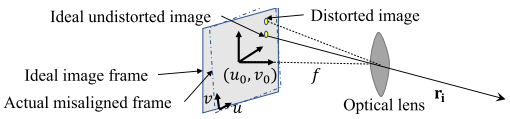

Fig. 1. Simple representation of the camera model.

accompanying calibrations process. Section III introduces the artificial neural network (ANN) and SVM frameworks with notes on our approaches to effective training.

Evaluation of the novel calibration processes is organized into three main experiments. The first experiment (see Section IV) explores the training process and overall performance using an engineering model ST16RT star tracker (ST-1). This unit was available, in-house, so we were able to vary the data collection process for both calibration and validation. The second experiment investigates the required storage for model parameters and the computational cost for software implementation of the best performing models from the first experiment. The third experiment applies the most promising techniques to historical calibration and validation data collected during the manufacture of 11 additional sensors (ST-2–ST-12). This larger cohort of sensors is valuable for establishing wider trends in the improved performance that can be expected from these new approaches to calibration. Together, these experiments provide a thorough investigation of the promise of ANN-based calibration.

A. Prior Work

The extant literature on camera calibration is extensive. These studies examine applications such as stereovision, 3-D reconstruction, and robotic navigation. Commonly applied camera calibration approaches look for basis functions of optimized camera parameters that best represent the relationship between 2-D image position and actual object position in the world coordinate frame. Formulation of these basis functions often starts with the simple pinhole perspective projection model and adds some number of aberration corrections. The pinhole model assumes that the object is projected by a straight line through the projection center. However, radial, decentering, and other distortion effects can shift the position of the undistorted image as depicted in Fig. 1.

Wang et al. [1], Tsai [6], Zhang [7], and Heikkila and Silven [8] are few examples of authors who have presented low-order parametric camera models and calibration procedures. These studies use a variety of basis functions to model camera aberrations and frequently discuss optimization methods to avoid local minima when solving for the camera parameters. Common parameters used by these models include focal length, optical center, radial distortion coefficients, etc. Specialized calibration imagery is used to fit optimal numerical values to these model parameters. Although closed-form solutions are sometimes possible, most methods rely, at least in part, on a nonlinear, iterative optimization. For instance Tsai's [6] and Wei et al.'s [9] approach involves a two-step optimization process. The first step assumes there is no distortion and uses a closed-form solution to find best initial guesses for the core intrinsic parameters. These initial values are then used for a full numerical optimization of the camera model.

The alternative camera models such as the RFM presented by Tao and Hu [3] and the generic camera models [10] are good alternatives for applications where camera behavior cannot be easily represented as modifications to simple pinhole camera optics. RFMs are the *de facto* standard approach in photogrammetry and relate image and world coordinates through ratios of polynomials. These are medium-order model—often containing a few dozen parameters.

Researchers have also explored using ANNs for camera calibration. Some works [4], [5], [11] have used multilayer perceptron (MLP) networks to implicitly find the centroid position of image from the 3-D real-world coordinates of the corresponding object or the directional angles without the need to find the camera parameters. These works often compare the performance of the ANN-based implicit calibration with Tsai's [12] or other explicit camera calibration methods and show improvements. In some literature [13], the intrinsic and extrinsic parameters of camera are also found using the ANN approach. Although they report some performance improvement, this approach seems unnecessarily restrictive as the system is constrained by the same explicit parameterization. Other than simple MLPs, RBFNs [14] and generalized regression neural networks (GRNNs) [15] are other neural network architectures commonly used for regression and function approximation problems.

Apart from ANNs, the SVM regression (SVR) approaches have also been used for camera calibration. These works often use SVR to find the projection matrix (camera parameters) explicitly [16] or estimate camera distortion [17]. SVMs are often faster to train than the ANNs and are less sensitive to the initialization of parameters. However, ANN is a better choice for high-dimensional large dataset and can also have multiple outputs.

Unlike many terrestrial calibration applications, our goal is to predict star direction vectors with minimal angular error; reconstructing full 3-D world-vectors is of limited utility. In addition, compatibility with the on-orbit autonomous recalibration methods [18], [19], [20] is also important. Thus, while the absolute orientation of the sensor is known during ground calibration, the relative geometry between stars is the only truth measure available for on-orbit studies.

B. ST-16RT Star Tracker

The Sinclair Interplanetary ST-16RT series star trackers are the baseline sensors in this study. The pixel detector used in this star tracker is the *onsemi* MT9P031.[1] Table I shows the properties of these sensors. The exact focal distance is optimized during the calibration process.

The lens system used in the star tracker is a custom four-element, F1.6 lens. It is designed to be achromatic and match the microlens acceptance angles on the detector. During production, detector placement and tilt is set using shims and focus adjustments are made using bulk motions of the lens

[1]Formerly produced under Micron and Aptina brand names.

TABLE I
PROPERTIES: SINCLAIR INTERPLANETARY ST-16RT SERIES STAR TRACKERS

Parameter	Values
Field of View (FOV)	7.5 × 11.5deg. half axis
Detector Type	Aptina MT9P031, CMOS
Pixel Pitch	2.2 μm
Lens diameter	12 mm
Nominal Focal Length	16 mm
Active Pixel Array	1944×2592

assembly. More details of the star tracker optics are found in Sinclair et al. [21].

II. STAR TRACKER CALIBRATION

The calibration process collects a series of sample images and attempts to identify the relationship between a specific area on the detector and the corresponding direction vector in 3-D-space. In contrast to terrestrial cameras that must deal with complex scenes and variable working distances, the star tracker imaging environment is comparatively simple. Stars are all very far away, allowing us to treat our imaging targets as point sources at infinity. We must take care during calibration to separate the behaviors of the star tracker (i.e., intrinsic properties) from those of the laboratory environment (i.e., extrinsic properties).

The remainder of this section outlines the procedures, models, and equipment used during the calibration process of the ST-16RT star trackers. We present details of the calibration apparatus, introduce the mathematical minimization framework used for assessing goodness of fit, and review the parametric calibration model used by these sensors.

A. Data Acquisition

Laboratory calibration of the ST-16RT star tracker uses a precision three-axis gimbal (see Fig. 2). Star trackers are mounted to the gimbal and can then be rotated about three orthogonal axes. This system has a measured repeatability of 0.002° and an encoder resolution twice as fine. A pinhole light source and an off-axis parabolic mirror create a collimated beam that illuminates the sensor. Viewed from the sensor, a perfectly collimated beam replicates a point source at infinity. Thus, any translation of the sensor due to gimbal motion will not affect the incidence angle of light striking the sensor. The actual wavefront radius of curvature (as measured by a shearing interferometer) is ∼1.5 km which contributes less than 1 μrad of the systematic error).

During calibration, the beam axis is first aligned with Axis-3 of the gimbal. Then, through a series of gimbal motions, the star image is moved throughout the FOV. The image and/or position of the star target on the detector is recorded at each orientation, and collectively these datasets are known as *surveys*. Most surveys contain centroid data from $N = 300$ unique orientations. As part of the production process, an initial survey is used for calibration and a later survey is collected for validation purposes after all the environmental tests (e.g., thermal, vibration) have been performed on the sensor.

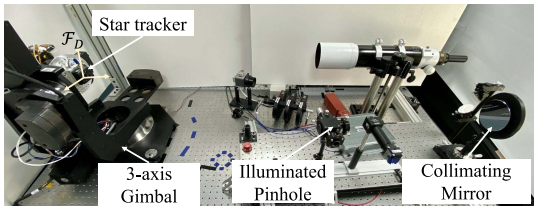

Fig. 2. Laboratory setup for calibration data acquisition.

Four coordinate frames are used during calibration, and we use the notation $\mathcal{F}_A$ as a shorthand for Frame-A. Effective calibration relies on understanding the relationships between these frames. The base frame for all the calibrations is the inertial laboratory frame, $\mathcal{F}_L$. In this frame, the beam direction (i.e., star vector) is constant. This quantity is denoted $\mathbf{r}_L$ and is measured during the beam alignment process. Motion of the gimbal determines the orientation of the end-effector frame $\mathcal{F}_E$, a theoretical frame that depends only on the platform joint angles, $\mathbf{\Theta}$, and the gimbal kinematics. The sensor frame, $\mathcal{F}_M$, is defined by two precision alignment surfaces on the sensor chassis, and internally, $\mathcal{F}_D$ is fixed to the sensor detector.

Given the constant star vector in inertial laboratory frame, $\mathbf{r}_L$, the corresponding detector frame vector, $\mathbf{r}_D$, is found by computing

$$\mathbf{r}_D = \mathbf{C}_{\mathrm{DM}}\mathbf{C}_{\mathrm{ME}}\mathbf{C}_{\mathrm{EL}}\mathbf{r}_L \tag{1}$$

where $\mathbf{C}_{\mathrm{AB}}$ is the direction cosine matrix (DCM) that rotates $\mathcal{F}_B$ to $\mathcal{F}_A$. The $\mathbf{C}_{\mathrm{EL}}$ DCM is computed from the platform kinematics and includes three principal axis rotations in the joint angles, $\mathbf{\Theta} = [\,\theta_1\ \theta_2\ \theta_3\,]^T$ as well as additional small rotations to account for axis misalignment. At the ith position in the survey, we have

$$\mathbf{C}_{\mathrm{LE}} = \mathbf{C}_{\mathrm{LE}}(\mathbf{\Theta}_i) = \mathbf{C}_{\mathrm{LE},i}\,. \tag{2}$$

The $\mathbf{C}_{\mathrm{EM}}$ rotation can be estimated with the aid of alignment features on the chassis. The $\mathbf{C}_{\mathrm{MD}}$ rotation (an intrinsic parameter) depends on the misalignment of image detector with respect to the camera mount surface and can be found during the numerical optimization.

As mentioned earlier, separate surveys are conducted for calibration and validation. To avoid overfitting, the platform orientations used for validations are offset from those used for calibration, and thus, $\mathbf{r}_{i,\mathrm{calib}} \neq \mathbf{r}_{i,\mathrm{valid}}$. The intrinsic rotation $\mathbf{C}_{\mathrm{MD}}$ is constant, barring bulk changes in the star tracker itself, but the extrinsic $\mathbf{C}_{\mathrm{EM}}$ can change slightly as the sensor is detached and reattached to the gimbal platform. An external autocollimator is used to measure C_{EM} before each survey, but because star trackers are such precise instruments, the accuracy of this alignment measurement is often of comparable order to the residual calibration and validation errors. The implications of this are discussed in Section IV.

B. Systematic Errors

There are a number of stochastic and systematic errors that can affect calibration. Liebe [22] discusses star tracker error budgets in some detail, but we highlight some of the more

important factors here noting testing circumstances will dictate the salient error terms.

1) *Temporal Noise:* This is the expected observation-to-observation variability. The bulk of this effect should be driven by detector noise and the brightness of observed star. The structural and thermal disturbances in the laboratory could also affect these measurements but we have not seen any evidence of these effects in practice. Temporal noise contributes $\sim$1 arcseconds of the residual error.
2) *Gimbal Accuracy:* The stated manufacturers quoted accuracy of laboratory gimbal encoders is 0.001°. This contributes a standard deviation of 1.04 arcseconds.
3) *Gimbal Nonorthogonality:* The motorized gimbal axes are close to orthogonal, but not exactly so. We correct for this nonideality but some residual uncertainty in the correction may remain.
4) *Stellar Aberration:* Stellar aberration is present in any observation of real stars. Correcting for the Earth's heliocentric velocity can remove the bulk of this effect, but correcting for orbital motion as well can reduce the errors still further.

Table II lists the expected error introduced by each of the stated factors. When contemplating the net effect, we note that our cost functions are based on the arc lengths between pairs of star observations. All the errors add in a residual sum of square (RSS) sense.

TABLE II
CALIBRATION ERROR BUDGET

Factor	Error (arcseconds)
Temporal Noise	1.00
Gimbal Accuracy	1.04
Gimbal Non-Orthogonality	0.08
Stellar Aberration[a] (orbital observations, heliocentric correction)	1.36
Lab calibration (net)	2.86
Orbital calibration (net)	2.31

[a]Worst case differential effect across ST-16RT field of view.

C. ST-16RT Parametric Calibration Model

Star trackers must calculate the *outgoing*, estimated star vector, $\widehat{\mathbf{r}}$ that corresponds to a measure detector-plane centroid location, (u, v). This inverts the sense of many camera models, designed to calculate centroids from the world coordinates. Transcendental terms in the model frequently prevent algebraic inversion, so instead we formulate the inverse model directly.

Documented by Enright et al. [20], the ST16-RT camera model starts with a pinhole perspective projection model and adds basis functions that can model the distortion effects. We summarize the model equations here for convenience

$$\begin{aligned}\begin{bmatrix}U\\V\end{bmatrix} &= \begin{bmatrix}\gamma\,(u-u_0)\\ \gamma\, c_s(v-v_0)\end{bmatrix}\\ \begin{bmatrix}x_d\\y_d\end{bmatrix} &= \frac{f}{\alpha_2 U+\alpha_1 V+f}\begin{bmatrix}U\\V\end{bmatrix}\\ \mathbf{s} &= \begin{bmatrix}\left(1+\beta_1\rho^2+\beta_2\rho^4\right)x_d\\ \left(1+\beta_1\rho^2+\beta_2\rho^4\right)y_d\\ f\end{bmatrix}\end{aligned} \tag{3}$$

$$\hat{\mathbf{r}}_D = \frac{\mathbf{s}}{\|\mathbf{s}\|}. \tag{4}$$

In these equations, we find most of the explicit calibration parameters, namely,

1) the principal point, (u_0, v_0);
2) the focal distance, f;
3) radial distortion coefficients β_1, β_2;
4) tip/Tilt rotations α_1, α_2;
5) the y-axis skew ratio, c_s.

The remaining terms include the constant pixel pitch, γ, an unnormalized star direction, $\mathbf{s}$, and a number of intermediate coordinates used for clarity [i.e., U, V, x_d, y_d, $\rho = (x_d^2 + y_d^2)^{1/2}$]. Referring to (1), the rotation $\mathbf{C}_{\mathrm{MD}}$ must also be computed for precise calibration. After optimizing the camera parameters, the best-fit rotation can be found with Davenport's q-method or another Wahba problem solution. Thus, the calibration optimization must solve for the following state vector:

$$\mathbf{x} = \begin{bmatrix} f & u_0 & v_0 & c_s & \alpha_1 & \alpha_2 & \beta_1 & \beta_2 \end{bmatrix}^T. \tag{5}$$

D. Cost Function and Optimization

In the conventional calibration process, the optimal parameters are those that minimize the calibration cost function. This cost function is defined by comparing the star vectors in a pairwise manner. Assuming all the star vectors are normalized, $\|\mathbf{r}_i\| = 1$, the angular separation (arclength) between two calibration locations can be found from

$$\phi = \sin^{-1}\left(\|\mathbf{r}_i \times \mathbf{r}_j\|\right). \tag{6}$$

The model prediction angles are calculated using the $\mathcal{F}_D$ vectors from (3), and hence,

$$\widehat{\phi}_{ij} = \sin^{-1}\left(\|\widehat{\mathbf{r}}_{D,i} \times \widehat{\mathbf{r}}_{D,j}\|\right). \tag{7}$$

The true angles are calculated from the platform kinematics and the joint angles

$$\phi_{ij} = \sin^{-1}\left(\|\mathbf{r}_{E,i} \times \mathbf{r}_{E,j}\|\right) \tag{8}$$

where

$$\mathbf{r}_{E,i} = \mathbf{C}_{\mathrm{EL},i}\mathbf{r}_L. \tag{9}$$

The prediction arclength error is then

$$\delta\phi_{ij} = \phi_{ij} - \widehat{\phi}_{ij}. \tag{10}$$

Using all the possible pairwise combinations includes a large amount of redundant information. To avoid the numerical difficulties this can present during optimization, we select $N_k = 2N-3$ pairs of calibration points (see Enright et al. [20] for more details). Thus, we adopt the shorthand notation $\phi_k \equiv \phi_{i_k,j_k}$. The total cost over the calibration survey to be minimized is then

$$J = \frac{1}{2}\sum_{k=1}^{N_k}(\delta\phi_k)^2. \tag{11}$$

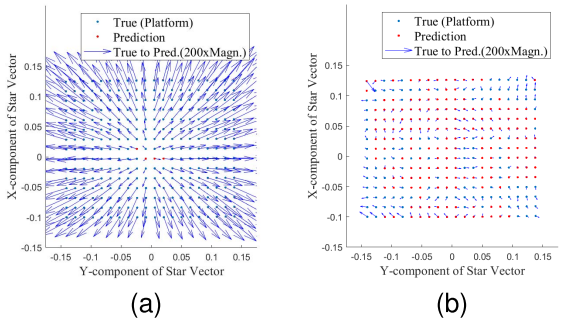

Fig. 3. Typical calibration residual plot. (a) Incorrect focal length. (b) Optimized parameters.

This formulation is convenient for conventional calibration as it decouples the parameters in (5) from $\mathbf{C}_{\mathrm{MD}}$ rotation. As such, it can be used for either terrestrial calibration (where platform kinematics are available) or on-orbit recalibration (where no external ground-truth orientation is available and ϕ_k must be calculated from cataloged star positions). The optimal $\widehat{\mathbf{x}}$ is found using the Levenberg–Marquardt or any other convenient numerical algorithm. The remaining intrinsic parameter $\mathbf{C}_{\mathrm{MD}}$ is found using the autocollimator derived $\mathbf{C}_{\mathrm{EM}}$ and a best-fit orientation calculation operating on $\widehat{\mathbf{r}}_{D,i}$ and $\mathbf{r}_{M,i}$.

Although the cost function is formulated in terms of the squared error magnitude, the rms arclength is a better metric for comparing calibrations from different sensors or surveys

$$E_{\mathrm{pair}} = \delta\phi_{\mathrm{rms}} = \sqrt{\frac{\sum_{k=1}^{N_k} \delta\phi_k^2}{N_k}} = \sqrt{\frac{2J}{N_k}}. \tag{12}$$

In addition, it is often convenient to directly examine the spatial error distribution across the FOV. This requires a slightly different cost function, evaluated for individual star

$$E_{\mathrm{vec}} = \sqrt{\frac{\sum_{i=1}^{N} \left(\sin^{-1}\left(\left\|\mathbf{r}_{D,i} \times \widehat{\mathbf{r}}_{D,i}\right\|\right)\right)^2}{N}}. \tag{13}$$

This quantity can be used to select best performing prediction model from terrestrial calibration survey. Both E_{vec} and E_{pair} cost functions can be used during calibration; however, there are small but important differences between the two. The E_{pair} formulation does not depend on measuring the rotation C_{EM} [evident from (1) and (8)], so it generally gives more consistent results when the sensor is remounted between calibration and validation. In contrast, although E_{vec} validations are tied to the precision of the rotation measurement, they have the advantage of enabling a more concrete and intuitive visualization of residual error.

Fig. 3 shows a typical residual plot for a calibration survey. Fig. 3(a) shows an example of the radial errors that result from an error in f, while Fig. 3(b) shows a typical postoptimization residual plot. Although the overall cost function value predicts acceptable performance from the sensor, some systematic error is clearly still present in the residual distribution—the model parameters are simply unable to correct these particular trends. It is this phenomenon that has motivated the supervised ML approach described in this study.

III. Alternative Calibration Models

Apart from the traditional low-dimensional parametric models, other generic camera models exist which do not rely on the physical parameters of the camera geometry. Methods such as RFMs and ANNs express the relationship between image pixel position and the corresponding 3-D vector in a purely mathematical way without explicit computation of physical parameters of the camera.

A. Rational Function Models

The RFM has been used extensively in photogrammetry research and practice. When formulated for inverse applications—i.e., reconstructing incident direction vectors from image coordinates—the model has the following form:

$$r_x = \frac{P_1(u,v)}{P_2(u,v)}, \quad r_y = \frac{P_3(u,v)}{P_4(u,v)}, \; r_z = \frac{P_5(u,v)}{P_6(u,v)} \tag{14}$$

where $P_i(u,v)$ is a polynomial function of the image centroid position. Each polynomial is an eight-term cubic function as shown in (15)

$$P_i(u,v) = a_0 + a_1 v + a_2 u + a_3 vu + a_4 v^2 + a_5 u^2 + a_6 v^3 + a_7 u^3. \tag{15}$$

The numerical optimization techniques are used to find the model parameter values. Following the method presented by Tao and Hu [3], we first use direct least squares to find the initial estimates for the parameter values and refine them with nonlinear optimization. Both the optimizations use the cost function from (11). Before optimization, the centroid positions shifted and scaled to lie within ± 1.

B. Interpolation

Data interpolation can be an alternative reliable approach if the calibration dataset is large enough to cover most regions across the FOV. The calibration dataset is then used as a lookup table to interpolate the vector values for other areas (star image centroid locations) not included in the calibration dataset. Using the centroid locations (u, v) in the calibration dataset, the scattered data interpolation methods such as Delaunay triangulation can be applied to find the three vector components for any given 2-D image position. The scattered calibration centroid locations [$N = 300$ distinct points, $p_i = (u, v)$] are first triangulated using the Delaunay method. Then an interpolation function, $f(u, v)$, is found for each triangle such that the output star vector is

$$\mathbf{r}_i = f(u_i, v_i) \tag{16}$$

for all the points within the triangle. The linear multivalued triangular interpolation, as described by Amidror [23], is one method to find the interpolation function. This work also implements the linear interpolation. The interpolation approach, however, has a few drawbacks including higher storage requirement for the entire calibration dataset.

C. Supervised Machine Learning Approach

This section provides an overview of the supervised ML approaches that can be implemented as an alternative to the low-order parametric camera model discussed in Section II-C.

1) Neural Network Framework: Feedforward ANN can be used to learn the nonlinear relationship between the input data and the target output. MLP, RBFN, and GRNN are few examples of feedforward neural networks. The architecture of these networks is defined by the number of input variables, number of hidden layers, nonlinear mapping functions applied at hidden layers, number of outputs, and number of neurons or nodes at each hidden layer. The neural networks designed for the star tracker calibration would require two input nodes for the normalized position of the star image centroid on the image plane, $\mathbf{p} = [u, v]$, and the three outputs. Shallow networks with one or two hidden layers can be implemented due to simplicity of the star tracker calibration data.

Two ANN correction architectures are considered: HYBRID and IMPLICIT. As shown in Fig. 4, the network outputs for the fully IMPLICIT star tracker calibration are $\hat{r}_x$, $\hat{r}_y$, and $\hat{r}_z$ components of the unit star vector resolved in the sensor mount frame. From (1), $\mathbf{C}_{\text{EL}}$ and $\mathbf{C}_{\text{ME}}$ can be found by controlling the orientation of the platform and measuring alignment surfaces with an autocollimator. However, it is challenging to find the intrinsic rotation $\mathbf{C}_{\text{MD}}$ for the implicit ANN-based model. Therefore, the ANN for IMPLICIT method is trained with vectors resolved in $\mathcal{F}_M$. The HYBRID ANN architecture, on the other hand, outputs residual error of the star vector components found by the simple distortion free pinhole camera model ($\Delta\widehat{\mathbf{r}}_D = \mathbf{r}_D - \widehat{\mathbf{r}}_D$). The pinhole model is simply (3) but sets $\alpha_1 = \alpha_2 = \beta_1 = \beta_2 = 0$.

For each correction architecture, four different ANNs are applied with different structures for the hidden layers: NET1 is an RBFN, NET2 is a one layer feedforward network with the rectified linear activation function (ReLU) transfer function instead of the Gaussian RBF, NET3 is similar to NET2 with three hidden layers, and NET4 is a GRNN.

NET1, NET2, and NET4 each have one hidden layer with N_n neurons. Each neuron in the hidden layer adds a nonlinearity by transfer function $\varphi_i(\mathbf{p})$. NET3 has three hidden layers. The differences between these networks are the transfer functions $[\varphi_i(\mathbf{p})]$, weights applied at the input layer ($\mathbf{W}_i^0$), and bias applied at the hidden layer.

The RBFN (NET1) transfer function for each network input $\mathbf{p}_k$, at neuron, i, is a radial basis function (Gaussian form) given by

$$\varphi_{i,\text{net1}}(\mathbf{p}_k) = e^{\left(\frac{-\|\mathbf{p}_k - \mu_i\|^2}{2\sigma_i^2}\right)} \tag{17}$$

where μ_i (1×2 vector) and σ_i are the center and spread of the RBF at the ith neuron, respectively. The output of the hidden layer in vector form is then

$$\Phi = [\varphi_i], \quad 1 < i < N_n. \tag{18}$$

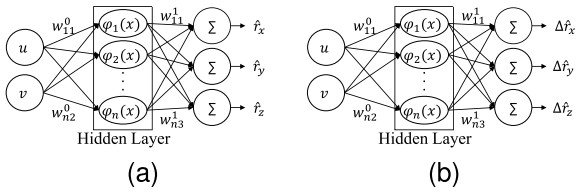

Fig. 4. ANN architectures for the two correction implementations. (a) IMPLICIT. (b) HYBRID.

The transfer function for NET2 and NET3 is the ReLU transfer function

$$\varphi_{i,\text{net2}}(\mathbf{p_k}) = \begin{cases} \mathbf{p_k}\mathbf{W}_i^0 + \mathbf{b}, & \mathbf{p_k}\mathbf{W}_i^0 + \mathbf{b} \geq 0 \\ 0, & \mathbf{p_k}\mathbf{W}_i^0 + \mathbf{b} < 0 \end{cases} \tag{19}$$

where $\mathbf{b}$ is bias, and $\mathbf{W}_i^0 = [w_{i1}^0 \; w_{i2}^0]^T$ are the optimized weights applied to the connection of inputs to the hidden layer neuron i as shown in Fig. 4. The trainable input layer weights are only applicable to NET2 (they are set to unity for RBFN).

At the output layer, the weighted sum of $\varphi_i(\mathbf{p})$ is calculated for NET1, NET2, and NET3. Given $\mathbf{W} = [w_{ij}^1]$ as a $3 \times N_n$ matrix containing the output layer weights and Φ as a $N_n \times 1$ vector containing the hidden layer outputs, the network would predict the star vector to be

$$\hat{\mathbf{r}}_M = \Phi^T \mathbf{W}^T + \mathbf{b}_2 \tag{20}$$

where $\mathbf{b}_2$ is a 1×3 vector containing the bias for each output. Then we normalize the star vector: $\hat{\mathbf{r}}_M = (\hat{\mathbf{r}}_M / \|\hat{\mathbf{r}}_M\|)$.

GRNNs have similar architecture to RBFN but with slight variation in the output layer. Unlike RBFN(NET1), the final network output is found by normalized dot product of weights and hidden layer outputs. The GRNN network can be trained using the MATLAB `newgrnn` function.

After selection of network architecture, we need to select an optimization method to train the network. The goal of training the neural network is to search for the learnable parameters such that the cost function is minimized. The learnable parameters for NET1-RBFN and NET4-GRNN are the number of neurons in the hidden layer (N_n), the center and spread of the RBFs (μ_i, σ_i), and the weights and biases of the output layer (w_{ij}, $\mathbf{b}_2$). The learnable parameters for NET2 and NET3 are the number of neurons, weights (w_{ij}^0, w_{ij}^1), and biases ($\mathbf{b}$).

Feedforward networks (NET2 and NET3) are often trained by the gradient descent method which iteratively updated the network learnables (weights and biases) using the gradient of cost function—often mse of the network output—with respect to the learnable parameters ($\mathbf{W}$, $\mathbf{b}$). Unlike NET2 and NET3, RBFN and GRNN can be trained without backpropagation. Some of the optimization methods commonly used to train RBFNs include orthogonal least squares, clustering, and gradient-based algorithm [14]. Two-step learning is a common approach for training RBFN where the first step involves optimization of the center and spread values (σ_i, μ_i) and the second step is finding the weights of the output layer. In some literature, a third step is added where backpropagation is used for optimization of all the learnable parameters. Schwenker et al. [24] discusses the training process of

RBFN in detail. The training process used for all the network architectures in this work uses the Levenberg–Marquardt method to minimize the cost functions (11), (13). The network weights and biases for NET2 and NET3 are initialized by the Nguyen–Widrow method. The spread values, σ_i, for NET1 are initialized to 1, and μ_i values are initialized by random selection of the network input values from the calibration (training) set.

Although the raw centroid data are calculated in pixel coordinates, some initial testing revealed that normalizing the centroid location by the array dimensions, $(u_{\max}, v_{\max})$, yielded superior error performance and required fewer neurons. This approach is used in all ML calibrations presented here.

2) SVM Framework: SVMs represent another supervised ML approach. These systems search for a linear predictor in a high-dimensional feature space. For nonlinear SVR, the input data in the training set are transformed to a higher dimensional feature space using a kernel such as Gaussian or polynomial $[G(x)]$.

The SVM learning process then becomes a convex optimization process where we minimize the loss term $[(1/2)\|\mathbf{w}\|^2+\Omega]$ such that $\mathbf{w}x_i + b - y_i \leq \epsilon$. $\mathbf{w}x_i + b$ is the equation of the optimal hyperplane that minimizes the model error, ϵ is a margin where deviations from the optimal hyperplane are tolerable (insensitive band), and Ω is a penalty term imposed on data points that are beyond the tolerable margin to avoid overfitting the training data points. More details on SVMs can be found in Vapnik [25].

Three SVMs are trained for each component of star vector in sensor mount frame. The inputs for each SVM are the 2-D location of centroid on the image plane. Similar to the neural networks described in Section III-C1, IMPLICIT-SVM model outputs the star vectors directly, while the HYBRID-SVM finds the corrections for the EXPLICIT model output. The SVMs are optimized using the SMO method presented by Platt [26]. Training parameters insensitive band (ϵ), Lagrange multipliers' constraint, and kernel scale are optimized by the Bayesian method.

IV. CALIBRATION RESULTS

To summarize the development thus far, we wish to compare the effectiveness of different *calibration methods* together with various *network structures*. The five calibration methods are as follows.

1) EXPLICIT: A conventional parameterized camera calibration [from (3)].
2) RFM: Rational function model as presented in Section III-A.
3) INTERPOLATION: Scattered triangular interpolation using the calibration dataset.
4) ML-HYBRID: An ANN correction to a simplified pinhole camera model. This will correct residual errors only.
5) ML-IMPLICIT: An ANN replacement for the entire camera model.

The five network structures for the ML-based methods (Methods 4 and 5) are as follows.

1) NET1: An RBFN-based ANN.
2) NET2: A single-layer feedforward ANN.
3) NET3: A three-layer feedforward ANN.
4) NET4: GRNN.
5) SVM: A support vector machine.

The calibration methods mentioned above are first tested for a single sensor unit (ST-1) available in-house. The best performing method is selected based on the value of E_{pair} from (12). The calibration and validation datasets for sensor ST-1 are collected without remounting the sensor.

The calibration methods that show improvement compared with the EXPLICIT method are then compared in terms of data storage and computational power needed for implementation. The best performing model and optimization procedure are then applied to a larger population of sensors to evaluate whether the results are generally valid for different sensor units. These datasets are gathered from production archives, not from the tests performed specifically for this study. Each sensor is associated with two datasets; one for calibration and another for validation. In general, the validation datasets are collected at different times than the initial calibrations. Therefore, remounting of sensor may have caused variations in $\mathbf{C}_{\text{EM}}$. Even though this change in $\mathbf{C}_{\text{EM}}$ for validation data can be measured by an auto-collimator, we can also account for measurement error by computing the bulk rotation from true survey vectors, $\mathbf{r}_M$, to the predicted $\widehat{\mathbf{r}}_M$ using the q-method. The bulk rotation would only affect the E_{vec} value, and this correction is not needed if E_{pair} is to be used as metric of performance. Without bulk rotation correction, the average E_{vec} for a population of 12 different sensors was found to be $\sim$100arcsec.

A. Experiment-1: Calibration and Network Structure

This section investigates the calibration results for the ST-1 sensor. From Section III-C1, the output of ANN from the IMPLICIT approach is expressed in $\mathcal{F}_M$ and should be rotated to $\mathcal{F}_D$ using $\mathbf{C}_{\text{MD}}$ found from the EXPLICIT method to compare the accuracy of the calibration methods.

The number of survey points in the calibration dataset can affect the accuracy of the trained prediction model. Historically, 300-point datasets have provided good accuracy for the EXPLICIT models, and same historical datasets are used to train the ML-based models discussed earlier. The ST-1 calibration and validation datasets in Table III also use 300-point surveys. Section IV-B provides more details on effective calibration (training) dataset size. As discussed in Section II-A, offsetting the gimbal angles ensures that the calibration and validation survey locations are disjoint on the camera FOV.

Table III shows a summary of the calibration results for ST-1, where E_{vec} and E_{pair} values are expressed in arcsec. The SVM method performed poorly for the IMPLICIT calibration method. NET1 listed in Table III showed best results with respect to the EXPLICIT method with $\sim$20% improvement in the angular error of star vectors. GRNN leads to best calibration data fitting results at the expanse of larger network size. Furthermore, high level of overfitting is observed as evident by a poor validation result. Even though GRNN

TABLE III
SUMMARY OF VALIDATION RMSE FOR ST-1

Calibration Method	N_h[a]	N_n	Calibration		Validation	
			E_{vec} [arcsec]	E_{pair} [arcsec]	E_{vec} [arcsec]	E_{pair} [arcsec]
EXPLICIT	n/a	n/a	6.82	3.04	8.87	3.51
RFM	n/a	n/a	4.95	2.88	7.60	3.54
INTERPOLATION	n/a	n/a	n/a	n/a	5.3	2.80
IMPLICIT-SVM	n/a	n/a	113.23	434.07	79.68	431.67
IMPLICIT-NET1	1	25	2.86	2.09	5.09	2.48
IMPLICIT-NET2	1	25	5.95	3.55	8.56	4.38
IMPLICIT-NET3	3	25	3.62	2.98	6.93	3.41
IMPLICIT-GRNN	1	300	1.80e-01	4.19e-02	45.09	1.84
HYBRID-SVM	n/a	n/a	4.84	1.73	6.79	2.50
HYBRID-NET1	1	25	2.99	2.27	6.42	2.72
HYBRID-NET2	1	25	4.02	2.77	7.13	3.62
HYBRID-NET3	3	25	2.86	2.48	6.67	3.32

[a]Number of hidden layers

validation E_{pair} is good, overfitting often results in output vectors to be a rotated or translated version of the true values, which is evident by poor E_{vec} results. Although NET3 shows more improvement compared with the one-layered NET2, it is less efficient for implementation due to larger number of parameters and higher computational cost resulting from higher number of hidden layers. Comparing NET1–NET3, the RBFN performs the best. Furthermore, the two-step learning process commonly used for RBFNs can result in better performance and does not rely on backpropagation which leads to faster training process. For instance, RBFN training implemented by MATLAB function `newrb` results in fast consistent training results. The INTERPOLATION method also gives us accuracy levels similar to NET1, but requires storing a lookup table with fairly higher number of values (i.e., 1200 values for 300 survey points' calibration dataset).

Section IV-C provides the optimization results for a larger population of sensors to better judge which prediction model is the best choice in terms of performance.

Fig. 5 shows comparison of the residual errors plot for the EXPLICIT, IMPLICIT-NET1, and HYBRID-NET1 methods. We can see that the residuals are improved in areas where the EXPLICIT fails to accurately model the distortions effects for this particular sensor. However, there some structured error still remains. The validation performance improvements are smaller than those seen in calibration, suggesting that some overfitting may be present.

Lower errors (E_{pair}) in star vector computation can lead to improvements in the attitude accuracy. Although the EXPLICIT model listed in Table III provides sufficient level of attitude accuracy for most applications, significant improvements in E_{pair} justify the implementation of the alternative. In this case, NET1 (RBFN) shows lower E_{pair} with acceptable added computational complexity which in turn leads to higher attitude accuracy.

B. Experiment-2: Effect of Network Size and Computational Costs

From Experiment-1, the NET-1 and INTERPOLATION techniques show promise. However, we expect the size of the network and the amount of training data—i.e., the size of the calibration survey—to influence the final performance as well. Three different calibration datasets are gathered for ST-1: one with 300 survey points which was used for Experiment-1, one with 600 data points, and finally a 1200-point dataset. These three different datasets are used for training the RBFN, and the same 300-point validation dataset is used to show the impact of the calibration survey size. Furthermore, the number of the hidden layer neurons, N_n, for NET-1 is also varied. Fig. 6 depicts the change in validation E_{vec} as the network size and calibration dataset size are changed.

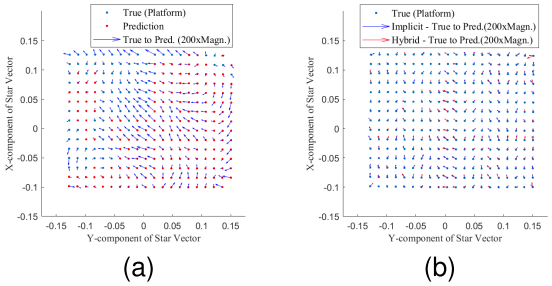

Fig. 5. Plot of residual errors for the validation dataset (ST-1). (a) EXPLICIT calibration ($E_{vec} = 8.87$ arcsec). (b) RBFN calibration (IMPLICIT: $E_{vec} = 5.09$ arcsec and HYBRID: $E_{vec} = 6.49$ arcsec).

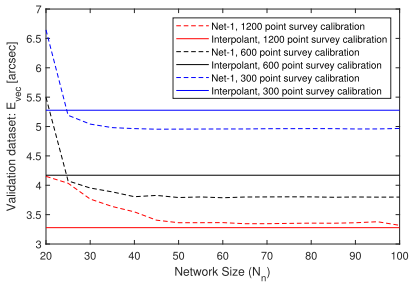

Fig. 6. Effect of calibration survey size on validation results (E_{vec}[arcsec] for validation set).

The ideal number of hidden layer neurons, N_n, was found to be 25. When varying the number of neurons for the IMPLICIT-NET1 system trained by MATLAB function `newrb`, the validation E_{vec} values decrease from 20 arcsec with 15 neurons to 5.15 arcsec with 25 neurons. Very small networks performed poorly, but each of the curves in Fig. 6 hits a point of diminishing returns after which accuracy improvements are minimal. That said, larger networks benefit from additional training data (i.e., number of survey points) pushing the plateau points toward larger network size which is not desirable as it adds to computational complexity.

Where interpolation is implemented, smaller calibration size is desired to reduce the data storage requirement. Interpolation would be a better option over RBFN if higher accuracy with smaller training dataset is achieved. However, RBFN (NET-1) shows better results for the 300- and 600-point calibration datasets. Furthermore, focal length corrections for sensor temperature and star (color) spectrum can be effective in lowering errors and these corrections would be difficult to integrate with the interpolation without repeat of the calibration process.

Knowing the accuracy of the discussed prediction models (refer to Table III) may not be sufficient, as computational

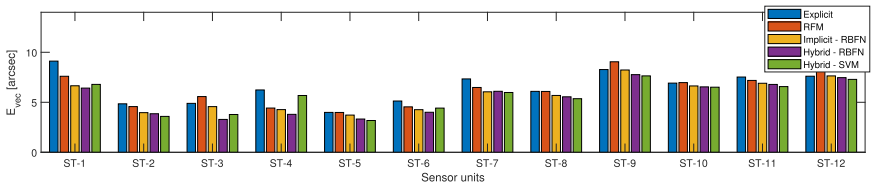

Fig. 7. Star vector accuracy for the historical validation dataset of multiple sensor units.

cost is also an important factor to consider during development of the star tracker sensor. The interpolation method requires a storage of a lookup table containing 300 centroid location and their corresponding 2 × 300 vector components (total of 1200 values). On the other hand, implantation of an RBFN with a single hidden layer of size 25 would be more desirable as it requires storage of 150 values (network weights and biases). Network size affects both training time and execution. As ML problems go, this application is not particularly demanding and the time required to train the networks is minimal. Thus, our primary consideration is the execution time. Reducing the network size for NET1 will a lead to a corresponding reduction in the online computations. In our trials, networks with 25 neurons seemed to offer the best balance between error and computational cost. To perform forward computations for NET1 i.e., 25 hidden layer neurons, two inputs and three outputs, 50 exponential and 75 multiplication operations are required. Of these, the exponentiation is by far the more expensive operation on most architectures. Simple benchmark tests on the Star Tracker processor (an Arm Cortex-A8) suggests that each exponential takes about 1 μs to evaluate, an equivalent of about 128 FLOPS. By comparison, the sensor takes about 226 FLOPS to calculate a centroid and 13 FLOPS to evaluate the EXPLICIT camera model.

For a full image of ten stars, the time spent evaluating the ANN is approximately 530 μs. Although this time is not negligible, it remains quite small compared with raw star detection, which typically takes 80–100 ms. Thus, for the moderate sized networks presented here, computational costs are not expected to a significant challenge.

C. Experiment-3: Unit-to-Unit Results

The ML approaches listed in Table III are applied to the historical calibration data from 11 other ST-16RT star tracker units to study the repeatability and variations in the prediction accuracy. These calibration datasets have same size (300 survey points) as the one for ST-1, so we expect similar trend to *Experiment-1* to hold true. Table IV lists the average improvements in E_{vec} and E_{pair} values compared with the EXPLICIT model. NET-2 and NET-3 networks show poor improvement results for E_{pair} values. Compared with other methods, both HYBRID-NET1 and IMPLICIT-NET1 show highest level of average improvement in validation E_{pair} and E_{vec}. Therefore, we posit that NET1 is the preferred network architecture for star tracker calibration.

In general, E_{pair} improvements are lower than E_{vec}, because the residual errors are not uniform along the FOV, and the paired vectors selected to compute the value of E_{pair} are nonredundant (see Enright et al. [20] for more details). Furthermore, the ANN networks are optimized by minimizing E_{vec}. In case of RFM where E_{pair} was optimized, E_{pair} improvements are better than E_{vec}.

TABLE IV
EXPERIMENT-3: MEAN CALIBRATION IMPROVEMENT (ANN-BASED VERSUS EXPLICIT)

Calibration Method	Average improvement Calibration E_{vec}	Calibration E_{pair}	Validation E_{vec}	Validation E_{pair}
RFM	-0.91%	8.77%	1.90%	6.75%
IMPLICIT-NET1	33.33%	29.52%	14.98%	11.20%
HYBRID-NET1	35.34%	31.14%	16.52%	12.83%
IMPLICIT-NET2	19.75%	8.85%	10.81%	0.44 %
HYBRID-NET2	6.53%	-13.11%	-1.06%	-30.93%
IMPLICIT-NET3	22.30%	14.36%	5.99%	3.02%
HYBRID-NET3	34.80%	22.61%	12.24%	-1.43%
GRNN	97.49%	99.28%	-417%	34.06%
HYBRID-SVM	33.10%	25.73%	14.04%	6.17%

Along with Table IV which shows the averaged results, Fig. 7 illustrates the improvement for each individual sensor unit for few of the best performing ML approaches. The RMSE values shown in Fig. 7 correspond to a validation set with 300 data points which is used to test the trained models. It is notable that the HYBRID methods result in slightly higher level of improvements in the RMSE values than fully IMPLICIT. Some sensors like ST-1 show a large change between EXPLICIT and ANNs. Others like ST-10–ST-12 show smaller performance gains. This difference between sensors is not directly tied to the initial EXPLICIT performance as ST-5 with good EXPLICIT results shows better results for ANN, while ST-9-EXPLICIT has similar residuals to ST-1, but sees less benefit from ANNs.

The reasons for these disparities are not clear. It is possible that the variations in sensor noise from sensor to sensor result in different levels of centroid error. The variations in the lens assemblies may also explain some of the disparities. However, in the end, all the sensors see some improvement with NET1, and this varies from 5% to 35%.

V. CONCLUSION

In this study, we have presented an ANN-based optical calibration method for star tracker. This approach resulted in improvements of the star vector angular errors relative to the explicit calibration procedure based on the pinhole camera model with radial and decentering distortions. The RBFN approach implicitly finds a mapping from the star image centroid position on the image frame to the 3-D unit star vector. avoiding the need to find accurate camera parameters. These RBFNs are of modest size and require very little change to extant calibration survey–collection processes. Although

larger training sets are beneficial, the existing calibration procedures still deliver good results.

The explicit camera calibration methods are easier to inspect and validate by the mathematical models. However, finding accurate basis functions to model distortions can be challenging and can lead to structured residual errors. The RBFN proved to be capable of removing some of these structures observed in residual plots. The average validation improvements of 12% is observed.

Instrument calibrations can shift due to temperature changes, vibration, and component aging, affecting the overall instrument accuracy on-orbit. It is possible to use a batch estimator (refer to Enright et al. [20]) to recalibrate and update the RBFN parameters using on-orbit data to account for any changes caused by environmental factors.

Both HYBRID and IMPLICIT represent viable architectures for accurate star vector reconstruction. The former generally provides a slight performance improvement over the latter, at the expense of some additional computation. In addition, because the HYBRID system acts a correction to the well-understood EXPLICIT models currently in use, they arguably present less technical risk to a project.

Although some variations were observed from unit-to-unit in our experiments, almost all the sensors evaluated in this study demonstrated sizable performance gains, not just in their calibration residuals, but in processing of the separate validation datasets. This is an encouraging finding and suggests that ML calibration techniques could be productively applied not just to all ST-16-series star trackers, but to other instruments as well.

ACKNOWLEDGMENT

The authors would like to thank Sinclair Interplanetary for their donation of an engineering model star tracker and for facilitating access to on-orbit telemetry.

REFERENCES

[1] J. Wang, F. Shi, J. Zhang, and Y. Liu, "A new calibration model of camera lens distortion," *Pattern Recognit.*, vol. 41, no. 2, pp. 607–615, Feb. 2008.

[2] R. V. F. Lopes and M. D. Shuster, "Parameter interference in distortion and alignment calibration," *J. Astron. Sci.*, vol. 51, no. 3, pp. 261–277, Sep. 2003.

[3] C. V. Tao and Y. Hu, "A comprehensive study of the rational function model for photogrammetric processing," *Photogramm. Eng. Remote Sens.*, vol. 67, no. 12, pp. 1347–1357, Dec. 2001.

[4] D. M. Woo and D. C. Park, "Implicit camera calibration using an artificial neural network," in *Neural Information Processing*, vol. 4233. Berlin, Germany: Springer, 2006, pp. 641–650.

[5] J. Li and Z. Liu, "Camera geometric calibration using dynamic single-pixel illumination with deep learning networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2550–2558, Aug. 2020.

[6] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE J. Robot. Autom.*, vol. RA-3, no. 4, pp. 323–344, Aug. 1987.

[7] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.

[8] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Juan, Puerto Rico, Jun. 1997, pp. 1106–1112.

[9] X. Wei, G. Zhang, Q. Fan, J. Jiang, and J. Li, "Star sensor calibration based on integrated modelling with intrinsic and extrinsic parameters," *Measurement*, vol. 55, pp. 117–125, Sep. 2014.

[10] S. Ramalingam, P. Sturm, and S. K. Lodha, "Towards complete generic camera calibration," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Jun. 2005, pp. 1093–1098.

[11] J. L. Xiong, J. Y. Xia, X. Q. Xu, and Z. Tian, "A novel method of stereo camera calibration using BP neural network," *Appl. Mech. Mater.*, vols. 29–32, pp. 2692–2697, Aug. 2010.

[12] R. Y. Tsai, "An efficient and accurate camera calibration technique for 3D machine vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami Beach, FL, USA, May 1986, pp. 364–374.

[13] M. T. Ahmed, E. E. Hemayed, and A. A. Farag, "Neurocalibration: A neural network that can tell camera calibration parameters," in *Proc. 7th IEEE Int. Conf. Comput. Vis.*, Kerkyra, Greece, Sep. 1999, pp. 463–468.

[14] Y. Wu, H. Wang, B. Zhang, and K.-L. Du, "Using radial basis function networks for function approximation and classification," *ISRN Appl. Math.*, vol. 2012, pp. 1–34, Mar. 2012.

[15] D. F. Specht, "A general regression neural network," *IEEE Trans. Neural Netw.*, vol. 2, no. 6, pp. 568–576, Nov. 1991.

[16] R. Mohamed, A. Ahmed, A. Eid, and A. Farag, "Support vector machines for camera calibration problem," in *Proc. Int. Conf. Image Process.*, Atlanta, GA, USA, Oct. 2006, pp. 1029–1032.

[17] C.-Y. Yang, G. E. Jan, and Y.-Y. Chen, "Robust CMOS camera module lens calibration by support vector machine regression," *Proc. SPIE*, vol. 8759, Jan. 2013, Art. no. 875949.

[18] T. Sun, F. Xing, and Z. You, "Optical system error analysis and calibration method of high-accuracy star trackers," *Sensors*, vol. 13, no. 4, pp. 4598–4623, Apr. 2013.

[19] W. Tan, D. Dai, W. Wu, X. Wang, and S. Qin, "A comprehensive calibration method for a star tracker and gyroscope units integrated system," *Sensors*, vol. 18, no. 9, p. 3106, Sep. 2018.

[20] J. Enright, I. Jovanovic, and B. Vaz, "Autonomous recalibration of star trackers," *IEEE Sensors J.*, vol. 18, no. 18, pp. 7708–7720, Sep. 2018.

[21] D. Sinclair, J. Enright, T. Dzamba, and T. Sears, "Custom optics vs modified COTS for small spacecraft: The build vs. rebuild decision," in *Proc. AIAA/USU Conf. Small Satell. (SSC)*, 2015. [Online]. Available: https://digitalcommons.usu.edu/smallsat/2015/all2015/47/

[22] C. C. Liebe, "Accuracy performance of star trackers—A tutorial," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 2, pp. 587–599, Apr. 2002.

[23] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: A survey," *J. Electron. Imag.*, vol. 11, no. 2, pp. 157–176, 2002.

[24] F. Schwenker, H. A. Kestler, and G. Palm, "Three learning phases for radial-basis-function networks," *Neural Netw.*, vol. 14, nos. 4–5, pp. 439–458, May 2001.

[25] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 2000.

[26] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," Microsoft, Redmond, WA, USA, Tech. Rep. MSR-TR-98-14, Apr. 1998.

Shaghayegh Khodabakhshian (Graduate Student Member, IEEE) received the B.Eng. and M.A.Sc. degrees in aerospace engineering from Toronto Metropolitan University (formerly Ryerson University), Toronto, ON, Canada, in 2018 and 2020, respectively, where she is currently pursuing the Ph.D. degree with the Space Avionics and Instrumentation Laboratory (SAIL).

Her current research interests include improving performance of star trackers for nanosatellite applications.

John Enright (Member, IEEE) received the B.A.Sc. degree in engineering science from the University of Toronto, Toronto, ON, Canada, in 1997, and the M.S. and Ph.D. degrees in aerospace systems from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1999 and 2002, respectively.

He is currently an Associate Professor in aerospace engineering with Toronto Metropolitan University (formerly Ryerson University), Toronto, where he is also a Principal Investigator with the Space Avionics and Instrumentation Laboratory (SAIL). He has developed algorithms and flight software in wide use on commercial star trackers and sun sensors.

Dr. Enright is a member of the American Institute of Aeronautics and Astronautics (AIAA) and Canadian Aeronautics and Space Institute (CASI).