Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Low-earth Satellite Orbit Determination Using Deep Convolutional Networks with Satellite Imagery

ROHIT KHORANA[1]

[1]Saint Francis High School, Mountain View, CA 94040 USA (e-mail: rohit.khorana.email@gmail.com)

Corresponding author: Rohit Khorana (e-mail: rohit.khorana.email@gmail.com).

ABSTRACT It is increasingly common for satellites to lose connection with the ground stations on Earth with which they communicate, due to signal interruptions from the Earth's ionosphere and magnetosphere. Given the important roles that satellites play in national defense, public safety, and worldwide communications, finding ways to determine satellite trajectories in such situations is a crucially important task. In this paper, we demonstrate the efficacy of a novel computer-vision-based approach, which relies on earth imagery taken by the satellite itself, to determine the orbit of a satellite that has lost contact with its ground stations. We empirically observe significant improvements, by more than an order of magnitude, over the present state-of-the-art approach, namely, the Gibbs method for an initial orbit estimate with the Kalman filter for differential error correction. We further investigate the performance of the approach by comparing various neural networks, namely, ResNet-50, ResNet-101, VGG-19, VGG-16, AlexNet, and CoAtNet-4.

INDEX TERMS kalman filter, neural network, orbit determination, satellites.

I. INTRODUCTION

A. TRENDS WITH SATELLITES

The number of satellites in space is increasing, most notably owing to SpaceX's Starlink constellation [1, 2], which aims to provide high-speed Internet services to greater parts of the world, particularly in underdeveloped or rural regions. As more satellites are being launched, the probability of collisions between them also increases. Such collisions lead to an increase in space debris, causing difficulties for ground-based astronomical observations. Moreover, space debris contribute to pollution on Earth [3]. Thus, it is important to effectively track satellite trajectories.

B. WHAT IS ORBIT DETERMINATION?

Satellite orbit determination is the computational process of determining the state of a satellite (state vector, state, ephemeris) as a function of time using sets of measurements collected on board the satellite or by ground-based tracking stations [4, 5]. Current ground-based tracking systems make use of active radar calibrators (ARCs), which are placed at accurately known locations on Earth and provide range and range-rate measurements [6]. These measurements are converted into the state vector of the satellite through the use of statistical estimation and orbit propagation techniques [5]. For satellites in orbit, however, disturbances such as those due to solar radiation, atmospheric phase screening, and azimuth phase noise interfere with communications between satellites and ground stations and can even result in a loss of connection [7]. One notable instance of this occurred with the European Space Agency's trio of satellites known as Swarm, which were launched in 2013 with the aim of better understanding the Earth's magnetic field. Swarm itself lost connection 166 times within its first two years of operation. It was eventually found that in certain regions of the satellite orbits, Earth's ionosphere and magnetosphere had interrupted the signals between the satellites and ground stations, and it was impossible to locate the satellites for a few hours [8]. Therefore, it is necessary to have techniques available that are able to locate satellites when such situations arise.

C. A BRIEF HISTORY OF ORBIT DETERMINATION

The field of orbit determination has evolved substantially over the past fifty years. In general terms, orbit determination refers to the estimation of the orbits of objects in space. In the context of this paper, these objects are specifically satellites. Orbit determination algorithms rely on ground-based or onboard observations of a satellite as input and have been continually refined over the years. Following the launch

of Sputnik in 1957, orbit determination began to capture the attention of many astronomers worldwide. During the 1960s, the primary methods for orbit determination involved the use of camera tracking and radio Doppler tracking systems [9], both of which relied on data stemming from observations taken on Earth, with Baker, Nunn, Schmidt, and Hewitt cameras [5]. These methods had an accuracy of around 0.5 km, which can be considered to be quite low [9]. However, in the 1970s, with the introduction of laser-ranging technology, the accuracy of observations improved to roughly 5–10 m [5]. The methods used today are the results of many refinements and improvements in radio tracking techniques, force modeling, and laser technology.

D. THE STATE OF THE ART IN ORBIT DETERMINATION

The current state-of-the-art technique for obtaining orbital data relies on systems of well-connected ARCs that provide range and range-rate measurements. From these measurements, the position and velocity of a satellite are calculated [5, 6], and the initial state is refined by means of differential correction techniques such as least squares or Kalman filtering techniques. ARCs are used with synthetic aperture radar (SAR), where the primary objective is to obtain high-resolution images. ARCs send pulses regularly from antennas and store the resulting echoes in a two-dimensional (2D) matrix, which is referred to as the raw data. Then, through the use of an SAR processor, a focused image is obtained after two compressions have been performed on the received raw data [6, 10]. The first compression is performed along the range direction, using a matched filter, and the second is performed along the azimuth direction, with the signal being compressed by an SAR algorithm. The primary issue with the use of SAR, and ARC, however, is that the lateral resolution degrades with increasing operating distance [6], and other factors such as frequency shift can exacerbate the inaccuracies even further [11].

After the data provided by the ARC has been obtained, the next step in determining the orbit of a satellite is to apply initial orbit determination (IOD) techniques. The two most common techniques for IOD are trilateration and the Gibbs method. Trilateration accepts the range and range-rate measurements, taken at the same time, provided by three connected ground bases, as well as each of their coordinates: geodetic latitude, geodetic longitude, and altitude [6, 12]. Through the application of geometric principles, trilateration is able to produce a satellite state vector. In the scenario where only range measurements can be provided, trilateration can be implemented sequentially, although it then becomes quite complicated. Thus, the far simpler Gibbs method is employed in such situations [6]. This method takes as input three geocentric position vectors taken at three different times (for which the position vectors are provided by trilateration) and then calculates the state vector at the second timestamp [13]. It should be noted, however, that the Gibbs method requires all input position vectors to be coplanar [13].

This preliminary orbit can then be refined through the use of differential correction techniques, the two most common being least squares and Kalman filtering. The primary difference between these is that the least squares approach involves using systems of linear equations to estimate parameters [14], whereas the Kalman filtering approach uses measurements in tandem with an equation describing the motion of the system [15].

The methodology described in this paper represents a paradigm shift in orbit determination: our approach uses patterns and machine learning instead of a linear-check relation, and it takes into account the external factors influencing the motion of satellites orbiting the Earth.

It is increasingly common to attach cameras to satellites for remote sensing purposes. These satellites use these cameras to provide important information: the military uses such information for defense and intelligence purposes, commercial companies use it for communication and entertainment services, and GPS uses it for mapping.

E. A HIGH-LEVEL OVERVIEW: OUR APPROACH

We take a distinctive approach to performing orbit determination, through the provision of an on-premise computer vision solution to the problem. As previously mentioned, satellites carry cameras, and these can be used for an additional task, namely, to take images of the Earth. We make use of these images to locate the satellite with a level of accuracy higher than that achievable with the current state of the art approach when a satellite loses contact with its ground-based operating systems.

II. METHODS

We begin by collecting various images of the Earth taken by satellites in conjunction with the corresponding positions of the satellites and the times at which the images were acquired. We then construct a dataset consisting of images mapped to flattened position vectors, and we train our model on this dataset. We record the resulting root mean square error (RMSE) across various models and report the test set accuracy as well as the graphs of the training and test set errors. We compare this novel method with the current state of the art. We do so by implementing the Gibbs method and Kalman filtering, and simulating their ability to predict the future state vectors given the initial vectors. We then determine the RMSE between the predictions and actual positions.

A. DATASET AND PREPROCESSING

Our dataset consists of a training image dataset containing 8,000 images of the Earth, a validation set of 1,300 images, and a test set of 700 images, all of which were taken from Landsat 7 and Landsat 8 and made publicly available by Google Research. The Landsat program, which is part of the U.S. Geological Survey's National Land Imaging (NLI) Program, involved sending eight satellites into low-Earth orbit (LEO) periodically, with Landsat 1 being launched in

1972 and Landsat 8 in 2013, to collect data and image the planet [16].

In particular, we collect images as well as the corresponding metadata from Landsat 7 and Landsat 8 on Bands 3 and 4, respectively. The different bands correspond to various ranges of frequencies in the electromagnetic spectrum, and were designed such that the sensors aboard the satellites produce images with an emphasis on a particular aspect of the Earth. Specifically, the wavelength range of Band 3 (0.64–0.67 µm) on Landsat 7, and that of Band 4 (0.64–0.67 µm) on Landsat 8 discriminate vegetation-covered slopes from other terrain. Furthermore, the image dataset that we use is diverse, containing images of every continent and major landmass on Earth.

For every image in the dataset, we downsize the original image to $256 \times 256 \times 3$ to save space. We then extract the corresponding ephemeris Earth-centered, Earth-fixed (ECEF) coordinate data, or the position vector of a satellite in the ECEF reference frame for each image from the metadata. We ensure consistency in the dataset by converting all vectors to tensors with fixed data type and normalize by dividing every value by 10^4. To be precise, for all images i in our dataset, we have a corresponding position vector $v \in \mathbb{R}^{165\times 1}$. We then train our model to associate the images with the position vectors, from which the orbit can be exactly determined by simply adding the time-stamp.

B. MODEL ARCHITECTURES AND IMPLEMENTATION

We have tested various neural nets and have observed consistent phenomena. We compare the performance of these neural networks with the Gibbs method and Kalman filtering. We discuss each of the model architectures and our implementation.

1) ResNet-101 and ResNet-50

ResNet-50 is a type of residual convolutional neural network (CNN) with 50 layers. ResNet-101 is a type of residual CNN developed by He et al. [17]. Its architecture is identical to that of ResNet-50 except for the addition of three more layer blocks, leading to a total of 101 layers. Central to the architecture of the ResNet model is the notion of skip connections and identity mapping. In essence, given input x and desired underlying mapping $H(x)$, He et al. [17] let $F(x) := H(x) - x$ and recast the initial mapping to $F(x) + x$. They argued that it is easier to optimize the residual mapping than the original un-referenced mapping. This formulation $F(x) + x$ is realized through skip connections, which skip one or more layers.

We implement ResNet-50 using the ImageNet weights simply by importing both the model and the weights. We feed the output of ResNet-50 through an additional GlobalAveragePooling2D layer, two dense layers with ReLU activations, and a final dense layer with linear activation. The loss function we seek to minimize is the RMSE, and the optimizer we use is stochastic gradient descent. We train the model for a total of 200 epochs and evaluate its performance on the test set. We do the same for ResNet-101.

2) VGG-19 and VGG-16

VGG-16 is a convolutional neural network with 16 weight layers, in which the image is passed through a stack of convolutional layers that have a small receptive field, and these are followed by max-pooling layers and several fully connected layers [18]. The hidden layers use the ReLU activation function. VGG-19 has a similar architecture to VGG-16, with the notable difference of having 19 weight layers [18].

We implement VGG-19 and VGG-16 using the ImageNet weights simply by importing both the model and the weights. We feed the output of the VGG-19 through an additional GlobalAveragePooling2D layer, two dense layers with ReLU activations, and a final dense layer with linear activation. The loss function we seek to minimize is the RMSE, and the optimizer we use is stochastic gradient descent. We train the model for a total of 200 epochs and evaluate its performance on the test set. We do the same for VGG-16.

3) AlexNet

AlexNet was devised by Krizhevsky et al. [19]. The model consists of eight layers with weights, namely, five convolutional layers and three fully connected layers. The first convolutional layer filters the image with a stride of four pixels, and the result is fed to a max-pooling layer. The second convolutional layer receives the output from the max-pooling layer and filters it with 256 kernels, and the remaining convolutional layers are connected without pooling or normalization layers in between [19].

We implement the model as per the original paper [19], followed by a final dense layer with linear activation.

4) CoAtNet-4

CoAtNet is a CNN that combines the strengths of convolutional neural nets and self-attention. In CoAtNet, Dai et al. [20] unified depthwise convolution and self-attention by stacking convolution layers and attention layers in a natural way. Their key insight came from noticing that depthwise convolution and self-attention can be expressed as a per-dimension weighted sum of values in a predefined receptive field. Specifically, for input x_i, output $y_i \in \mathbb{R}^D$, weights w, position i, and neighborhood $\mathcal{L}(i)$, the depthwise convolution can be described as

$$y_i = \sum_{j \in \mathcal{L}(i)} w_{i \leftarrow j} \odot x_j. \tag{1}$$

Similarly, self-attention, for global spatial space $\mathcal{G}$, can be defined as

$$y_i = \sum_{j \in \mathcal{G}} \frac{\exp(x_i^\top x_j)}{\sum_{k \in \mathcal{G}} \exp(x_i^\top x_k)} x_j. \tag{2}$$

Dai et al. [20] combined these ideas by computing the sum of a global static convolution kernel with an adaptive attention matrix. Essentially, after a softmax, one can state

$$y_i^{\text{post}} = \sum_{j \in \mathcal{G}} \left(\frac{\exp(x_i^\top x_j)}{\sum_{k \in \mathcal{G}} \exp(x_i^\top x_k)} + w_{i \leftarrow j} \right) x_j, \quad (3)$$

with a slightly different construction in the event that one wishes to perform prenormalization. Dai et al. used prenormalization relative attention and the transformer block to produce their model.

C. KALMAN FILTERING AND THE GIBBS METHOD

1) General Kalman filter

a: Description of Kalman filtering

The basic Kalman filter takes as inputs the initial state vector, the initial state error, the covariance of the process noise, the covariance of the observation noise, and measurements taken from sensors [21]. The Kalman filter is a recursive filter with two phases: prediction and update. In the prediction phase, the Kalman filter estimates the state at a later time, using the state transition matrix, which is derived from the Taylor series of the state at a certain time t [21]. In addition, a new covariance will be produced, to approximate the uncertainty of the said prediction. In the update phase, a measurement of the state is taken via sensors. However, this measurement comes with some error, and the covariance of this measurement relative to that of the prediction is used to calculate the Kalman gain [22]. The Kalman gain represents the scaling factor, and it determines the relative impact of the sensor's measurement and predicted state on the updated state.

b: Mathematical underpinnings

Mathematically, Kalman filtering is based on linear dynamical systems discretized in the time domain. To use the Kalman filter to estimate the internal state of a process given only a sequence of noisy observations for each time step, we specify the given time domain T and $\forall t_i \in T$, the state transition model F_{t_i}, the observation model H_{t_i}, the covariance of the process noise Q_{t_i}, the covariance of the observation noise R_{t_i}, and the control vector u_{t_i}. The Kalman filter supposes that the state at time t_i is dependent upon the state at time t_{i-1} according to $x_k = F_{t_i} x_{t_i - 1} + B_{t_i} u_{t_i} + w_{t_i}$. Note that process noise $w_{t_i} \sim \mathcal{N}(0, Q_{t_i})$. At time t_i, an observation or measurement z_{t_i} of the true state x_{t_i} is made according to $z_{t_i} = H_{t_i} x_{t_i} + v_{t_i}$, where H_{t_i} is the observation model and v_{t_i} is the observation noise.

2) Kalman filter for orbit determination

a: Extended Kalman filtering and current techniques for orbit determination

Current approaches to the determination of satellite orbits generally use extended Kalman filtering, which is one of the most widely used estimators for nonlinear problems like orbit determination [23]. The extended Kalman filter differs from the standard Kalman filter is that it first linearizes the problem at hand and then applies the linear Kalman filter to the resulting linear system [23].

The extended Kalman filter constitutes a state-of-the-art estimation algorithm for orbit determination [24], or, more specifically, for predicting the future state vector of a satellite. In this context, continuous measurements are taken by GPS units, and so the extended Kalman filter's estimations are repeatedly refined. In other words, at every time step, the satellite position must be used to recalculate the state matrix and state transition matrix. Note that this algorithm is implemented in the discrete time domain. We let $x_t = f(x_{t-1}, u_t) + w_t$ and $z_t = h(x_t) + v_t$, where u_t is the control vector, $w_t \sim \mathcal{N}(0, Q_t)$ and $v_t \sim \mathcal{N}(0, Q_t)$ are the process and observation noises, both of which are assumed to be zero-mean multivariate Gaussian with covariance matrices Q_t and R_t, f is used to compute the predicted state from the previous estimate, and h is used to compute the predicted measurement from the predicted state. In practice, f and h cannot be applied directly to the covariance, and the Jacobian must be used instead. The algorithm is shown as Algorithm 1.

b: Extended Kalman filtering and connection problems

In this paper, we present a novel computer-vision-based methodology for orbit determination in which images are used to determine the ECEF vectors and thus the position at later time steps. The need for such an approach arises when a satellite loses contact with its ground station, which, as already mentioned, is a frequent occurrence. Given the nature of this problem, when contact is lost between satellite and ground station, extended Kalman Filtering cannot work as intended. In this scenario, one has only two choices: to use the last received GPS position vector or not to use GPS information in the Kalman update step at all. We attempt both approaches and empirically demonstrate that no matter what choice is made, the current state-of-the-art approach, using the Gibbs method in tandem with extended Kalman filtering, performs significantly worse than ResNet50, and slightly worse than many CNNs. The second choice will seem quite familiar to those well acquainted with orbital mechanics, since this algorithm essentially solves the differential equations describing satellite motion using Cowell's approach through Runge–Kutta methods [25, 26].

Suppose that we make the first choice and use the last received GPS position vector. Whenever a satellite travels in a region with high latency or close to Earth's magnetosphere and loses connection with its ground station, future GPS measurements cannot be received. The appropriate algorithm then takes the form shown as Algorithm 2. Henceforth, we shall call this method extended Kalman filtering with fixed GPS coordinates, or EKFFG for short.

Suppose now that we make the only other choice, namely, we do not use the last received GPS position vector. In this case, only the initial state, the initial state error, the covariance of the process noise, and the covariance of the observation noise are taken as inputs. Thus, the filter does not have any measurements from sensors to aid its estima-

Algorithm 1 Current algorithm for Extended Kalman filter

Ensure: for every time step $t \in T$

Let $x_{t|i}$ denote the estimate of state x at time t using observations up to time $i \leq t$

Let $F_t = \left.\frac{\partial f}{\partial x}\right|_{x_{t-1|t-1}, u_t}$

Let $H_t = \left.\frac{\partial f}{\partial x}\right|_{x_{t|t-1}}$

Prediction:

Compute predicted state estimate $x_{t|t-1} = f(x_{t-1|t-1}, u_k)$

Compute predicted state estimate $x_{t|t-1} = f(x_{t-1|t-1}, u_k)$

Compute predicted covariance estimate $P_{t|t-1} = F_t P_{t-1|t-1} F_t^\top + Q_t$

Update:

Compute measurement residual $y_t = z_t - h(x_{t|t-1})$

Compute covariance residual $S_t = H_t P_{t|t-1} H_t^\top + R_t$

Compute Kalman gain $K_t = P_{t|t-1} H_t^\top S_t^{-1}$

Update state estimate $x_{t|t} = x_{t|t-1} + K_t y_t$

Update covariance estimate $P_{t|t} = (I - K_t H_t) P_{t|t-1}$

Algorithm 2 EKFFG

Ensure: for every time step $t \in T$:

Let $x_{t|i}$ denote the estimate of state x at time t using observations up to time $i \leq t$

Let $F_t = \left.\frac{\partial f}{\partial x}\right|_{x_{t-1|t-1}, u_t}$

Let $H_t = \left.\frac{\partial f}{\partial x}\right|_{x_{t|t-1}}$

Let z be fixed as the last received GPS position vector

Prediction:

Compute predicted state estimate $x_{t|t-1} = f(x_{t-1|t-1}, u_k)$

Compute predicted covariance estimate $P_{t|t-1} = F_t P_{t-1|t-1} F_t^\top + Q_t$

Update:

Compute measurement residual $y_t = z - h(x_{t|t-1})$

Compute covariance residual $S_t = H_t P_{t|t-1} H_t^\top + R_t$

Compute Kalman gain $K_t = P_{t|t-1} H_t^\top S_t^{-1}$

Update state estimate $x_{t|t} = x_{t|t-1} + K_t y_t$

Update covariance estimate $P_{t|t} = (I - K_t H_t) P_{t|t-1}$

tions. The appropriate algorithm is then Algorithm 3. This is essentially Cowell's method propagated forward in time. Henceforth, we shall refer to this algorithm as a Cowell propagator. The standard definition of a Cowell propagator does not require that account be taken of drag forces [27]. In our implementation, we do account for drag, but, if desired, it can be ignored, albeit at the expense of decreased accuracy.

III. RESULTS

We have tested various neural networks and have observed consistent phenomena. We now compare the performance of these neural networks with the Gibbs method and Kalman filtering. Results on the performance of each approach are presented in Tables 1 and 2. We trained each model a total of 25 times and have reported here the best observed RMSE on the test set across all 25 runs. For every model listed, we use ImageNet weights.

IV. DISCUSSION

A. PROBLEM STRUCTURE

1) Why CNNs are a natural choice

We believe that to better understand the performance of the various models, it is essential to understand the structure of the problem, which is that of finding a relationship between an image of dimension $256 \times 256 \times 3$ and a corresponding vector of ephemeris coordinates of dimension 1×165. CNNs are approximately translationally equivariant. Hence, for images in which a major landmark, continent, or body of water is shifted relative to some example previously used for training the network, the CNN is capable of reasonably predicting the ECEF, since it can rely on the approximate translational equivariance. In essence, the translational equivariance, in tandem with the template matching capabilities provided by

Algorithm 3 Cowell propagator

Ensure: for every time step $t \in T$:

Prediction:

Compute predicted state estimate $x_{t|t-1} = f(x_{t-1|t-1}, u_k)$ via Cowell's method

TABLE 1. CNN model performance

Model	Best test-set RMSE	Parameters ($\times 10^6$)
ResNet-101	29.5810	44.6
ResNet-50	32.9029	25.6
VGG-19	∞	143.6
VGG-16	NaN	138.4
CoAtNet-4	NaN	203.4
AlexNet	146.7944	62.3

TABLE 2. Extended Kalman filter performance

Model	Average RMSE	Best observed RMSE	Top 25% RMSE
EKFFG (extended Kalman 1)	9,170.4000	426.7865	447.8721
Cowell propagator (extended Kalman 2)	413.1018	176.5121	339.9398

the feature sharing, gives the CNN the ability to predict ECEF well. Put more simply, aspects of Earth's geography can be learned by local feature filters and combined in a meaningful way. Suppose for the sake of illustration that one has a local feature filter that detects mountains, another that detects lakes, etc. It is then evident that CNNs are able to exploit the geography of the surface of the Earth to make reasonable predictions. We thus argue that aspects of the problem structure make CNNs a natural choice.

2) Why is there high variance in RMSE across the models?

The reason for the superior performance of ResNet-101 and ResNet-50 lies in their residual connections. Given that for any component v_i of any normalized ephemeris vector v, we know that v_i is a relatively large real number, $v_i \approx \pm a \times 10^4$, where $|a| > 1$ and $a \in \mathbb{R}$, it can be seen that for deep neural networks, the gradient is capable of exploding, meaning that gradient descent becomes unstable. The traditionally accepted solution for an exploding gradient is to use residual connections. None of the other selected models, namely, VGG-19, VGG-16, AlexNet, and CoAtNet-4, use residual connections. Therefore, despite their ImageNet performance, their results do not generalize well to this image-based problem. As mentioned earlier, for normalization, we have simply chosen to divide each component v_i, for every v, by 10^4. To ensure that we do not lose a great deal of precision when we have to predict on the basis of new data and then inevitably scale up, we have chosen to only divide by 10^4 rather than by 10^7 and store each value as a float64.

B. WHY EXTENDED KALMAN FILTERING IS NOT RELIABLE ALL THE TIME

As shown previously, when a satellite loses access to its ground stations, using the last-received GPS coordinates is insufficient for Kalman filtering to get an accurate estimate of the satellite's position. Since satellites travel at very high speeds, of the order of magnitude of 7000 m/s, a position estimated by GPS (even if it was taken a few seconds earlier) is going to be quite far off the actual position.

Thus, if used during the update step, when the Kalman filter performs the weighted average between the estimated position from Cowell's method and the GPS coordinates, the updates made to the initial estimate are likely to decrease the accuracy of the initial orbit estimate. This problem is overcome by our implementation of the extended Kalman filter, where the last received GPS position vector is used as the measurement during the update phase of the filter.

V. CONCLUSION

We have explored how to tackle the increasingly common problem of satellites losing connection to the ground stations on Earth with which they communicate. We have demonstrated empirically that CNNs outperform extended Kalman filtering for orbit determination in such situations. We have further investigated the performance of the CNN-based approach by comparing various neural networks and have concluded that ResNets are most effective at predicting the ephemeris ECEF of a satellite in LEO. In addition, we have explained the variations in model performance, as well as the reasons behind the particular efficacy of ResNet-101. In the future, further improvements may be achieved by exploiting the symmetries present in the problem.

ACKNOWLEDGMENT

We would like to acknowledge Google for making available Landsat Data. We would also like to acknowledge Nilesh Chaturvedi, Arya Das, and Alexandros Kazantzidis for making their Gibbs method and Kalman filter code publicly available via GitHub [28]

REFERENCES

[1] J. C. McDowell, "The low Earth orbit satellite population and impacts of the SpaceX Starlink constellation," *Astrophys. J. Lett.*, vol. 892, no. 2, p. L36, 2020.

[2] Union of Concerned Scientists, "In-depth details on the 5,465 satellites currently orbiting Earth, including their country of origin, purpose, and other operational details." https://www.ucsusa.org/resources/satellite-database, 2022.

[3] M. Clormann and N. Klimburg-Witjes, "Troubled orbits and earthly concerns: Space debris as a boundary infrastructure," *Sci. Technol. Hum. Values*, vol. 47, no. 5, pp. 960–985, 2022.

[4] B. Schutz, B. Tapley, and G. H. Born, *Statistical Orbit Determination*. Elsevier, 2004.

[5] J. R. Vetter, "Fifty years of orbit determination," *Johns Hopkins APL Tech. Dig.*, vol. 27, no. 3, p. 239, 2007.

[6] M. Fernandez Uson, "Geosar mission: Orbit determination methods and techniques," Universitat Politècnica de Catalunya, 2016.

[7] H. D. Curtis, *Preliminary Orbit Determination*, 3rd ed. Boston: Butterworth-Heinemann, 2014, ch. 5, pp. 239–298.

[8] European Space Agency, "Swarm reveals why satellites lose track," https://www.esa.int/Applications/Observing_the_Earth/FutureEO/Swarm/Swarm_reveals_why_satellites_lose_track, 2016.

[9] W. H. Guier and G. C. Weiffenbach, "A satellite Doppler navigation system," *Proc. IRE*, vol. 48, no. 4, pp. 507–516, 1960.

[10] Y. K. Chan and V. Koo, "An introduction to synthetic aperture radar (SAR)," *Prog. Electromagn. Res. B*, vol. 2, pp. 27–60, 2008.

[11] M. Shimada, H. Oaku, and M. Nakai, "SAR calibration using frequency-tunable active radar calibrators," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 1, pp. 564–573, 1999.

[12] M. E. Hough, "Precise orbit determination using satellite radar ranging," *J. Guid. Control Dyn.*, vol. 35, no. 4, pp. 1048–1058, 2012.

[13] A. V. Schaeperkoetter, "A comprehensive comparison between angles-only initial orbit determination techniques," Ph.D. dissertation, Texas A & M University, 2012.

[14] G. J. Bierman, *Review of Least Squares Data Processing and the Kalman Filter Algorithm*. Elsevier, 1977, ch. II, pp. 13–32.

[15] W. Menke, "Links between Kalman filtering and data assimilation with generalized least squares," *Appl. Math.*, vol. 13, no. 6, pp. 566–584, 2022.

[16] USGS, "Landsat satellite missions," https://www.usgs.gov/landsat-missions/landsat-satellite-missions, 2021.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Preprint at https://arxiv.org/abs/1409.1556, 2014.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[20] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "CoAtNet: Marrying convolution and attention for all data sizes," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 3965–3977, 2021.

[21] E. M. Keil, "Kalman filter implementation to determine orbit and attitude of a satellite in a Molniya orbit," Ph.D. dissertation, Virginia Tech, 2014.

[22] Y. Kim and H. Bang, "Introduction to Kalman filter and its applications," in *Introduction and Implementations of the Kalman Filter*, F. Govaers, Ed. London: IntechOpen, 2018, pp. 1–16.

[23] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," *Proc. SPIE*, vol. 3068, pp. 182–193, 1997.

[24] M. A. Akram, P. Liu, M. O. Tahir, W. Ali, and Y. Wang, "A state optimization model based on Kalman filtering and robust estimation theory for fusion of multi-source information in highly non-linear systems," *Sensors*, vol. 19, no. 7, p. 1687, 2019.

[25] J. Mauty and G. Brodsky, "Cowell type numerical integration as applied to satellite orbit computations," NASA-TM-X-63542, 1969.

[26] J.-C. Yoon, B.-S. Lee, and K.-H. Choi, "Spacecraft orbit determination using GPS navigation solutions," *Aerospace Sci. Technol.*, vol. 4, no. 3, pp. 215–221, 2000.

[27] R. Flores, B. M. Burhani, and E. Fantino, "A method for accurate and efficient propagation of satellite orbits: A case study for a Molniya orbit," *Alexandria Eng. J.*, vol. 60, no. 2, pp. 2661–2676, 2021.

[28] N. Chaturvedi, A. Das, and A. Kazantzidis, "orbitdeterminator," https://github.com/aerospaceresearch/orbitdeterminator, 2022.