Blind Compensation of Angle Jitter for Satellite-Based Ground-Imaging Lidar

Ethan Phelps and Charles A. Primmerman

***Abstract*— Space-based ground-imaging lidar has become increasingly feasible with recent technological advances. Compact fiber-optic lasers and single-photon-sensitive Geiger-mode detector arrays push designs toward low pulse energies and high pulse rates. A challenge in implementing such a system is imperfect pointing knowledge caused by angular jitter, exacerbated by long distances between satellite and ground. Without mitigation, angular jitter would cause significant blurring of the 3-D data products. Reducing the error in pointing knowledge to avoid such problems might require extreme mechanical isolation, advanced inertial measurement units (IMUs), star trackers, or auxiliary passive optical sensors. These mitigations can increase cost and size, weight, and power considerably. An alternative approach is demonstrated, in which the two-axis jitter time series is estimated using only the lidar data. Simultaneously, a single-surface model of the ground is estimated as nuisance parameters. Expectation–maximization is used to separate signal and background detections while maximizing the joint posterior probability density of the jitter and surface states. The resulting estimated jitter, when used in coincidence processing or image reconstruction, can reduce the blurring effect of jitter to an amount comparable to the optical diffraction limit.**

***Index Terms*— Angle-jitter, Bayesian, expectation–maximization (EM), Geiger-mode detector, gradient descent, image correction, imaging, lidar, satellite, surface estimation.**

I. Introduction

MIT Lincoln Laboratory, Lexington, MA, USA, has pioneered the development of high-performance 3-D-lidar systems using photon-counting avalanche photodiode (APD) arrays [1]. We have successfully fielded several airborne 3-D-lidar systems. Currently, we are designing a space-based 3-D lidar.

The basic concept of a space-based ground-imaging lidar is shown in Fig. 1. A short-pulse (∼1 ns) high-repetition-rate laser illuminates a small patch on the ground (of order 70 m × 70 m). The backscattered light is imaged on a moderate-sized (128 × 128 pixel) APD array. The APD pixels detect individual photons and make precise timing measurements, from which surface heights can be calculated. To generate a surface-height map over a large area (of order 10 km^2), the laser beam and APD-array field of view (FOV) are rapidly scanned across the ground.

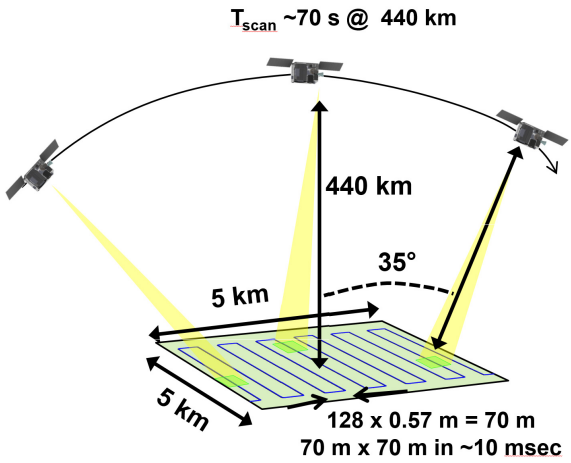

Fig. 1. Nominal concept for space-based ground-imaging 3-D lidar. The satellite-mounted lidar does a raster scan over a region of interest on the ground, uniformly illuminating a 70 m × 70 m area with each pulse, which projects onto a 128 × 128 array of Geiger-mode detector pixels. The dwell time for the FOV is on the order of 10 ms.

A requirement for developing an accurate surface-height map over a large area is that the majority of the individual height measurements are registered to a small fraction of a pixel (say < 1/3 pixel). We have achieved this registration accuracy for airborne systems, but achieving it in a space-based system is much more challenging for three reasons.

First, the airborne systems did not need to operate diffraction-limited, whereas the much longer-range space lidar will need to operate close to the diffraction limit to get adequate cross-range resolution. For the airborne systems, the ground sampling distance (GSD) was greater than the diffraction-limited ground resolved distance (GRD). This meant that the jitter could be larger than the GRD, but would not be significant because it was smaller than the GSD. For the space-based system, on the other hand, we need to have GSD ≈ GRD, and therefore, we will be more sensitive to jitter.

Second, for the airborne systems, the range was a few kilometers; for a space-based system, the range will be a few hundred kilometers. This difference in range means that, for equivalent ground resolution, a space-based system will be 100 times more sensitive to jitter than an airborne system. For an airborne system, a pointing jitter of 100 μrad might be adequate; a space-based system may need a pointing jitter < 1 μrad.

Manuscript received March 6, 2019; revised July 1, 2019, September 12, 2019, and September 23, 2019; accepted September 28, 2019. Date of publication October 30, 2019; date of current version January 21, 2020. This work was supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract under Grant FA8702-15-D-0001. *(Corresponding author: Ethan Phelps.)*

The authors are with the MIT Lincoln Laboratory, Lexington, MA 02421 USA (e-mail: ethan.phelps@ll.mit.edu; primmerman@ll.mit.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2019.2946752

Third, for our airborne systems, the pointing knowledge was provided by highly precise pointing mirrors, but for a space-based system, it would be desirable to eliminate the need for large, heavy, and expensive pointing mirrors.

On the other hand, satellites already exist that do passive imaging from space. Some of these satellites have small form factors, yet produce high-resolution imagery. They use solar illumination typically within the visible band, which has an irradiance at Earth's surface of about 500 W/m^2 [2]. This is about 25 000 times the surface irradiance of our lidar. The vastly higher signal rates allow them to produce image frames at a high rate (e.g., 2.5 kHz), which reduces jitter-induced blurring. They can then use frame-to-frame image registration [3]. This approach would not work well for our much lower signal rate.

To achieve the required registration accuracy, we have been investigating a blind approach, which estimates the two-axis jitter time series, and simultaneously, the surface height—viewed here as nuisance parameters. This approach has some similarities to blind deconvolution [4]. Some advantages of the approach are that it uses only the available data without requiring additional sensors, and it can potentially compensate even very high-frequency components of jitter that are comparable to the pulse rate. A challenge is the very low signal rate, which necessitates a long dwell time, thus increasing the amount of jitter observed and its blurring effect.

Although our jitter-compensation scheme was developed with a particular space-based lidar system in mind, we note that the scheme is completely general and could be implemented for many scanned 3-D lidar systems. This article describes the development and initial testing of this novel algorithm.

II. Design Considerations

As the satellite passes over a region of interest, the lidar does a raster scan over the region. The transmitted beam illuminates the whole FOV of the receive array. A narrower beam would make the signal less challenging to integrate but would require faster scanning to achieve the same area coverage rate, increasing the system complexity and mechanical requirements (e.g., larger reaction wheels and/or more vibration). A long and narrow beam would allow scanning at the same rate but would have worse registration errors. There may be an optimum beam aspect ratio, but we demonstrate good performance even in the challenging case of a square beam. The raster scan can be divided into multiple lateral sweeps or smaller segments that can be processed independently.

Table I lists the notional parameters of a space-based lidar system. Based on these parameters, the GSD at zenith is $(\lambda/D)(R/Q) = 0.57$ m, where λ is the wavelength, D is the aperture diameter, R is the range, and Q is the parameter described by Fiete [5], which we call the oversampling factor. The per-pulse photoelectron (PE) return rate across the whole array is $E_{\text{pulse}}(\lambda/hc)(\rho/\pi R^2)\pi(D/2)^2\gamma = 69$ PE/pulse, where additionally E_{pulse} is the pulse energy in Joules, h is Planck's constant, c is the speed of light, ρ is the target reflectivity, and γ is the receive efficiency, which combines quantum and optical efficiency. When distributed over 128×128 pixels, this return rate becomes 4.2×10^{-3} PE/pixel/pulse. At this low signal level, the necessary dwell time is long enough that jitter would cause significant blurring. A dwell time of 12.5 ms (or 2500 pulses) results in an average of about ten detections per GSD, which is roughly the requirement for forming a high-quality height image. Some image-formation algorithms, such as [6], seem to require as few as one detection per GSD, but that impressively low number is achieved by sacrificing angular resolution. This tradeoff is undesirable for a space-based system.

TABLE I
Example System Parameters

Parameter	Value
Array size (fully illuminated)	128 × 128 pixels
PRF	200 kHz
Pulsewidth (FWHM)	2 ns
Pulse energy	0.5 mJ
Wavelength	1 μm
Receive aperture diameter	1 m
Receive efficiency	0.2
Oversampling factor, Q	0.82
Background count rate	2 kHz
Surface reflectivity	0.1
Altitude	440 km
Jitter knee frequency	20 Hz
Long term jitter std. dev.	8.9 μrad
Dwell time per FOV	12.5 ms

Blurring and registration errors caused by jitter are more precisely caused by an error in pointing knowledge, which we call residual jitter. Random disturbances in pointing often have similar characteristics to Brownian motion, with a power spectral density (PSD) that is inversely proportional to the square of frequency. The error in angle measurements made by inertial measurement units (IMUs) tends to be dominated by angle random walk, which is also Brownian motion. Typically, accurate pointing information, such as from star observations, is received at a low rate, which limits what would otherwise be unbounded power in the residual jitter at low frequencies. In addition, frequencies that are much lower than the reciprocal of the dwell time do not contribute significantly to blurring. They do contribute to absolute registration errors, but we view image quality as more important. For these reasons, we model jitter via a first-order Gauss–Markov process, which has the PSD shown in Fig. 2. This random process is equivalent to Gaussian white noise passed through a single-pole low pass filter [7]. The Gauss–Markov PSD is approximately flat below its knee frequency of 20 Hz and approximates the Brownian motion PSD above 20 Hz. The 20-Hz knee frequency was somewhat arbitrarily chosen as a factor of 4 below the 80 Hz reciprocal of dwell time. The scaling of the Brownian motion PSD and consequently of the Gauss–Markov PSD was chosen to approximately match one of the largest PSDs among a set representing previous satellite systems.

Many pointing systems have resonances at certain frequencies, with corresponding peaks in the jitter PSD. This is a

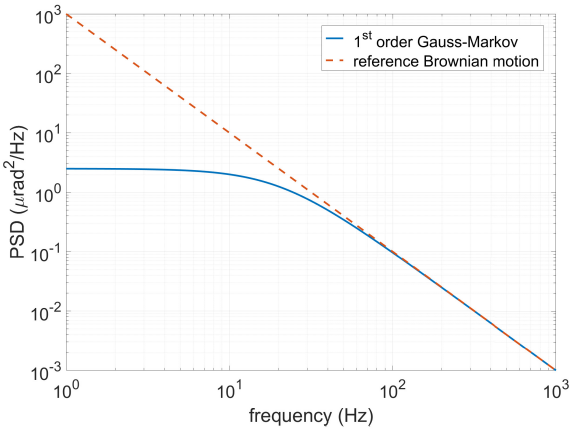

Fig. 2. Modeled PSD for residual angular jitter (solid line), representing a first-order Gauss–Markov process, and reference PSD (dashed line). The reference PSD represents Brownian motion, which is a common behavior in pointing systems. The modeled PSD has a knee at 20 Hz and fits the reference asymptotically at high frequencies.

mismatch to our assumed form of PSD, but we believe our approach would be tolerant to some peaks in the PSD if the parameters of the assumed PSD are chosen to approximately fit the upper envelope of the actual PSD. Since the algorithm is statistical in nature, it is naturally robust to some amount of mismatch.

A small-angle approximation is used to translate the angular jitter into the lateral motion of the optical axis on the surface, i.e., $\Delta x \approx R\theta$. We expect the algorithm to be quite robust to errors in this approximation partly because the estimated surface serves only to help estimate the jitter. Any warping of the estimated surface would only indirectly affect the jitter estimates. Using the parameters from Table I, the long-term RMS jitter per axis at the surface would be $\Delta x \approx 440\ \text{km} \times 8.9\ \mu\text{rad} = 3.9$ m. Since the dwell time is limited, the standard deviation of the observed jitter is smaller, at about 2.4 m. This represents very significant blur of about 4 pixels along the x- and y-axes. The relationship between dwell time and jitter fluctuation is explored in [8].

To make this material easier to communicate and to understand, the rest of this article will represent the application of a staring sensor rather than a continuously scanning one. This simplifies some aspects of describing the algorithm and makes the results easier to quantify. One could consider a "step and stare" mode that jumps by 1 FOV (70 m) after each dwell time (12.5 ms), while a scanning sensor would slew at the same average rate, but smoothly. Most aspects of the algorithm remain the same whether the sensor is staring or scanning.

III. Problem Formulation

A. Overview

The proposed algorithm is a very direct (but carefully implemented) solution to an inverse problem, based on the principle of Bayes' rule. It starts with a high-fidelity statistical model of the relevant physics. Bayes' rule is applied to determine the posterior probability density of the desired parameters. Because of the high dimensionality of the state space, we cannot represent the full probability density; therefore, we seek a maximum-likelihood solution. Expectation–maximization (EM) is used to provide a more tractable approximation to evaluating the posterior at a point in state space. Gradient descent is an optimization technique that works for maximizing the approximated posterior in a high-dimensional space. Sparsity and locality are exploited in the computations whenever possible, and array programming is used for parallelization.

B. Physical Modeling

The projection of the optical axis onto the ground is known to limited precision from IMU measurements and instrumentation, such as angle encoders. An orthographic projection model is used, which assumes that the change in angular pointing is small. The region of interest can be divided into sufficiently small pieces that this assumption is valid. With the orthographic projection model, angles are replaced by their projection onto the observed scene.

The satellite position and the projection of the optical axis, as determined from the IMU and instrumentation, are combined into a single 3-D vector function of time

$$g(t) = \begin{bmatrix} g_x(t) \\ g_y(t) \\ g_z(t) \end{bmatrix}. \tag{1}$$

The coordinate frame is fixed relative to the ground, with the xy plane orthogonal to the line of sight, and the z-axis pointing along the line of sight toward the satellite. The z component can be considered the range, due to the orthographic projection approximation. The xy plane, also called the reference plane, is defined such that it is close to the surface, notionally at the average surface level. The pixels of the detector array project onto the reference plane in a grid with spacing equal to one GSD.

The 3-D scene is modeled as the height of a single surface relative to the reference plane, as a function of the lateral coordinates, i.e., $h(x, y)$. This type of function is sometimes referred to as a 2.5-D image. Structures, such as bridges or cantilevered buildings, do not necessarily challenge the single-surface model (regardless of viewing angle), though they do result in abrupt changes in height. Real 3-D scenes can, of course, be more complex, with multiple surfaces per GSD and fine-grained structure, such as foliage, that allows lidar returns from a variety of ranges over a small line of sight change. Nevertheless, we believe this model to be a good approximation, as there is often a strong return from the first surface, and it is likely that a large portion of the scene will be well represented by the model. The goal is to represent the surface accurately enough to facilitate estimation of the jitter.

The jitter is modeled as a 2-D function of time that represents a lateral shift of the optical axis along the reference plane

$$j(t) = \begin{bmatrix} j_x(t) \\ j_y(t) \end{bmatrix}. \tag{2}$$

A 2-D representation of rotation is adequate because the third component of rotation—about the optical axis—does not get

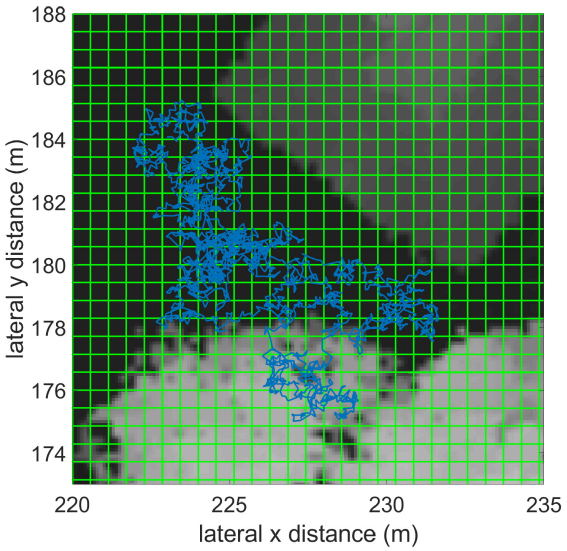

Fig. 3. Cropped view of truth height image (grayscale) with pixel grid (green) and jitter path (blue) overlaid. Over the dwell time, the center of the pixel grid follows the jitter path, leading to blurred imagery if this effect is not compensated. The jitter spans a large number of pixels and shows the scale of the blurring effect.

scaled by range, and therefore, its effect is over two orders of magnitude smaller. The x and y jitter functions are assumed to be independent first-order Gauss–Markov random processes.

Fig. 3 shows the surface height with the projected pixel grid and the jitter path. Over the dwell time interval, the center of the pixel grid would traverse the jitter path.

C. Measurement Process

The measurement process for each pixel is independent of the others, given the surface and jitter, which for the purpose of simulation are known quantities. The single-pixel measurement process is a Poisson process with a time-varying rate, which is the sum of contributions from noise and signal. This can be represented as two independent Poisson processes with rate functions that add up to the original. We model the noise with a rate function that is constant in time, and the signal with a rate function that is shaped like the returning pulse train after reflecting off the surface. The sensor records the time of each PE detection event, which may be due to noise or signal.

Low per pulse detection rates are assumed for both noise and signal, in order to ignore the effects of blocking. Blocking is a characteristic of a Geiger-mode detector (i.e., pixel) in which the detector takes some time (e.g., 2 μs) to reset, after making a detection. Any photons arriving within this time interval would not be detected—they would be "blocked" by the preceding detection. The reduction in detection rate is called blocking loss, which becomes significant as the rate increases toward the reciprocal of the reset time. An indirect effect of blocking is a skewing of the apparent pulse shape due to detections from the early part of the pulse preventing subsequent detections. Another effect is that statistical independence between detection times is broken. Low rates let us ignore these complications.

TABLE II
PROPERTIES OF PULSES

Variable	Description
$t_{TX,i}$	transmit time (center of pulse)
$e_{TX,i}$	transmit energy
$t_{REF,i}$	nominal reflection time (center of pulse)
$r_{REF,i}$	nominal reflection range
$t_{RX,i}$	nominal receive time (center of pulse)
$e_{RX,i}$	nominal receive energy
d_{il}	detection indicator for pixel l

The first step in modeling the signal process involves simulating the transmitted pulses. The transmit time of each pulse is recorded, which represents the center of the pulse. If the pulse energy is varied, then this is also recorded. A transmitted pulse travels at the speed of light toward the surface, reflects off the surface, and a fraction of the energy returns toward the lidar. It is convenient to define the time $t_{\mathrm{REF,i}}$ and range $r_{\mathrm{REF,i}}$ at which the i^{th} pulse would reflect off the reference plane. We call these the nominal reflection time and nominal reflection range. These are found by taking the intersection of the outgoing pulse's range versus time curve and the range versus time function $g_z(t)$ of the reference plane.

The nominal reflection time and range for the i^{th} pulse are related by

$$t_{\mathrm{REF,i}} = t_{\mathrm{TX},i} + \frac{r_{\mathrm{REF,i}}}{c} \tag{3}$$

where c is the speed of light, and the remaining variables are defined in Table II. Given the nominal reflection time, the nominal receive time is

$$t_{\mathrm{RX},i} = t_{\mathrm{TX},i} + 2(t_{\mathrm{REF,i}} - t_{\mathrm{TX},i}). \tag{4}$$

The nominal receive energy is

$$e_{\mathrm{RX},i} = \frac{\rho_{\mathrm{surface}} A \gamma}{\pi} \frac{e_{\mathrm{TX},i}}{r^2_{\mathrm{REF,i}}} \tag{5}$$

where ρ_{surface} is the surface reflectivity (assumed constant), A is the receive aperture area, and γ is the combined optical efficiency and quantum efficiency. Lambertian scattering is assumed. If $e_{\mathrm{RX},i}$ is expressed in units of counts, then it is the expected number of detections for the i^{th} pulse, across all pixels.

In simulating measurement data, a binary indicator variable d_{il} is randomly generated, which represents whether pulse i generates a detection in pixel l. This variable has a Bernoulli distribution, with parameter $e_{\mathrm{RX},i}/N_{\mathrm{pixels}}$ expressed in units of counts. Low per pixel signal rate is assumed here, i.e., $e_{\mathrm{RX},i}/N_{\mathrm{pixels}} \ll 1$. Another assumption here is uniform spatial illumination across pixels, though a shaped beam could be easily accommodated. The indicator variable is generated for every pulse and for every pixel.

Measurements are simulated over the time interval $[0, T]$. The sensor data contains the time and pixel number of each detection, which may arise from signal or from background noise. For simulation of signal detections for pixel l, the detection indicator variable d_{il} determines whether pulse i causes

a detection. For each pulse with $d_{il} = 1$, the detection time is modeled as

$$\zeta_{k|\text{signal}} = t_{\text{RX},i(k)} - \frac{2}{c}h(x_k, y_k) + v_k \tag{6}$$

where k is the detection index and $i(k)$ is the corresponding pulse index. v_k represents when the received photon was emitted relative to the center of the pulse, which is distributed as

$$v_k \sim \text{Gaussian}\left(0, \sigma_{\text{PW}}^2\right) \tag{7}$$

where σ_{PW} is the standard deviation representing the spread of the Gaussian pulse in time. Low per pixel signal rate is assumed here, i.e., $e_{\text{RX},i}/N_{\text{pixels}} \ll 1$, to avoid blocking-induced skewing of the probability density function (pdf) relative to the pulse shape.

The reflection coordinates at which the surface height is evaluated are defined as

$$x_k = g_x(t_{\text{RX},i(k)}) + j_x(t_{\text{RX},i(k)}) + o_x(n_k) + a_{xk} + b_{xk} \tag{8}$$

$$y_k = g_y(t_{\text{RX},i(k)}) + j_y(t_{\text{RX},i(k)}) + o_y(n_k) + a_{yk} + b_{yk}. \tag{9}$$

Both of these equations are the sum of the following five terms:

1) Reported projection of the optical axis at the nominal receive time ($g_x(t_{\text{RX},i(k)})$ and $g_y(t_{\text{RX},i(k)})$);
2) Jitter at the nominal receive time ($j_x(t_{\text{RX},i(k)})$ and $j_y(t_{\text{RX},i(k)})$);
3) Offset of the pixel center relative to the optical axis ($o_x(n_k)$ and $o_y(n_k)$);
4) Random location of photon arrival within the pixel (drawn from a uniform distribution function) (a_{xk} and a_{yk});
5) Optical blur (b_{xk} and b_{yk}).

The functions $o_x(n_k)$ and $o_y(n_k)$ return the x and y coordinates, respectively, of the center of pixel n_k relative to the optical axis. a_{xk} and a_{yk} are uniformly distributed random variables for the location of photon arrival within the pixel, relative to its center

$$a_{xk} \sim \text{Uniform}\left(\frac{-\Delta x_{\text{pixel}}}{2}, \frac{+\Delta x_{\text{pixel}}}{2}\right) \tag{10}$$

$$a_{yk} \sim \text{Uniform}\left(\frac{-\Delta y_{\text{pixel}}}{2}, \frac{+\Delta y_{\text{pixel}}}{2}\right) \tag{11}$$

where Δx_{pixel} and Δy_{pixel} are the pixel widths, which will be equal for square pixels. b_{xk} and b_{yk} are Gaussian distributed random variables representing the blur due to the diffraction pattern of the optics

$$b_{xk} \sim \text{Gaussian}\left(0, \sigma_{\text{blur}}^2\right) \tag{12}$$

$$b_{yk} \sim \text{Gaussian}\left(0, \sigma_{\text{blur}}^2\right). \tag{13}$$

The Gaussian distribution approximates an Airy function, using $\sigma_{\text{blur}} = 0.42\lambda/D$, where λ is the wavelength and D is the aperture diameter.

Background noise can come from dark counts or from sources outside the sensor that are uncorrelated with the pulses. Background noise is modeled as a Poisson point process, and therefore, the number of detections within the dwell time for a given pixel is Poisson distributed

$$N_{\text{noise},l} \sim \text{Poisson}(f_{\text{noise},l}T) \tag{14}$$

where $f_{\text{noise},l}$ is the background count rate for pixel l and T is the dwell time. The $N_{\text{noise},l}$ individual detection times $\zeta_{k|\text{noise}}$ are each independent and uniformly distributed over the dwell time

$$\zeta_{k|\text{noise}} \sim \text{Uniform}(0, T). \tag{15}$$

This is repeated to generate the noise for all pixels.

Once all signal and noise detections have been generated, they are merged by concatenating the two sequences and then sorting into time order. Repeating this process for all pixels results in the measurement data, which is the sequence of pixel number—detection time pairs for all detections within the dwell time—$\{n_k, \zeta_k\}_{k=1:N}$.

TABLE III
PROPERTIES OF DETECTIONS

Variable	Description
ζ_k	detection time (time of detection of a photoelectron)
n_k	pixel number
s_k	source indicator (noise or signal)
x_k	x coordinate of surface reflection of photon (valid only for signal detections)
y_k	y coordinate of surface reflection of photon (valid only for signal detections)

IV. ESTIMATION APPROACH

A. State Representation

The state is defined as a vector with three parts, representing the x jitter, y jitter, and surface height

$$\theta = \begin{bmatrix}\theta_x \\ \theta_y \\ \theta_h\end{bmatrix}. \tag{16}$$

The jitter and surface height functions are defined in a parametric form as the weighted sum of regularly spaced kernel functions. Each of the three parts of the state vector is a set of weights. The x and y jitter functions are

$$j_x(t;\theta_x) = \sum_i \theta_x(i)\phi_{\text{jitter}}\left(\frac{t - t_{\text{grid}}(i)}{\Delta t_{\text{grid}}}\right) \tag{17}$$

$$j_y(t;\theta_y) = \sum_i \theta_y(i)\phi_{\text{jitter}}\left(\frac{t - t_{\text{grid}}(i)}{\Delta t_{\text{grid}}}\right) \tag{18}$$

where t_{grid} is a vector of uniformly spaced times with interval Δt_{grid}. The jitter kernel is the triangular function, which makes the jitter functions equivalent to linear interpolation of the state vector elements

$$\phi_{\text{jitter}}(t) = \begin{cases}1 - |t|, & |t| < 1 \\ 0, & \text{otherwise.}\end{cases} \tag{19}$$

The sampling interval Δt_{grid} for defining the jitter sequences is chosen to be 10 μs, which is two times the pulse interval.

TABLE IV
SAMPLING GRIDS AND INTERPOLATION KERNELS

State elements	Dimension	Sampling interval	Kernel
x jitter (θ_x)	1251	10 μs	triangular (linear interpolation)
y jitter (θ_y)	1251	10 μs	triangular (linear interpolation)
surface (θ_h)	53 × 53	3 GSD	2D Gaussian (σ = 1.5 GSD)

This seems to be a good tradeoff between accurately representing the jitter and reducing the size of the state vector.

The height function is

$$h(x, y; \theta_h) = \sum_{i,k} \theta_h(i,k) \phi_{\text{surface}} \left(\frac{x - x_{\text{grid}}(i)}{\Delta x_{\text{grid}}}, \frac{y - y_{\text{grid}}(k)}{\Delta y_{\text{grid}}} \right) \quad (20)$$

where x_{grid} and y_{grid} are uniformly spaced distance vectors with intervals Δx_{grid} and Δy_{grid}, respectively. The double indexing of the state vector should be interpreted as if the vector were reshaped into the size of the 2-D grid. The surface kernel is a 2-D Gaussian function. A Gaussian kernel shape was chosen because it is smooth and convenient for taking derivatives, but it decays rapidly. The rapid decay simplifies the evaluation of the height function at a point by only requiring the evaluation of kernels within a small vicinity. The 2-D Gaussian kernel can be expressed as the product of two 1-D Gaussian functions

$$\phi_{\text{surface}}(x, y) = \phi_G(x) \phi_G(y) \quad (21)$$

$$\phi_G(w) = \exp\left(-\frac{1}{2} \left(\frac{w}{\sigma} \right)^2 \right). \quad (22)$$

The kernel width is determined by the standard deviation, which is chosen to be $\sigma = 1/2$. The lateral sampling interval (Δx_{grid} and Δy_{grid}) for defining the surface is chosen to be three GSD's. This represents a compromise between good performance and fast/reliable convergence. A finer sampling grid would allow more accurate representation of the surface, which would potentially improve estimation performance at the expense of increased compute time. However, it has been observed that too fine of a sampling grid can greatly hinder algorithm convergence. Table IV describes the size and spacing of the sampling grids, and therefore, the sizes of the state vector components. The state vector is quite large, with multiple thousands of elements, and therefore, care must be taken in the algorithm implementation.

B. Priors

The jitter and surface height functions are expected to be slowly varying, and this represents prior information that can be exploited. The jitter functions are modeled as two independent uniformly sampled first-order Gauss–Markov processes; and therefore, the θ_x and θ_y parts of the state vector are i.i.d. Gaussian random vectors with zero mean

$$\theta_x \sim \text{Gaussian}(0, Q_x) \quad (23)$$

$$\theta_y \sim \text{Gaussian}(0, Q_y). \quad (24)$$

The covariance matrix is large and nonsparse, but its inverse is conveniently sparse [9]–[12]. It is a tridiagonal matrix with only three unique nonzero values, given by

$$Q_x^{-1} = Q_y^{-1} = \begin{cases} \dfrac{1}{(1-\rho^2)\sigma^2}, & \text{first and last diagonal elements} \\ \dfrac{1+\rho^2}{(1-\rho^2)\sigma^2}, & \text{elsewhere on diagonal} \\ \dfrac{-\rho}{(1-\rho^2)\sigma^2}, & \text{on first off-diagonals} \\ 0, & \text{elsewhere} \end{cases} \quad (25)$$

where ρ is the correlation coefficient between adjacent samples, given by

$$\rho = \exp(-2\pi f_{\text{knee}} \Delta t_{\text{grid}}). \quad (26)$$

The long-term standard deviation σ and knee frequency f_{knee} are matched to the jitter PSD with the parameters in Table I. The covariance and inverse covariance are shown in Fig. 4.

The surface height function is modeled as a 2-D uniformly sampled first-order Gauss–Markov process (Markov random field). The vectorized array of height coefficients θ_h is also a Gaussian random vector

$$\theta_h \sim \text{Gaussian}(\bar{\theta}_h, Q_h). \quad (27)$$

The mean, $\bar{\theta}_h$ might be set by some preexisting knowledge of the surface, such as a digital elevation model, or it might be zero. The covariance is a Kronecker product of two matrices

$$Q_h = \left(\frac{1}{\sigma} Q_{hy} \right) \otimes \left(\frac{1}{\sigma} Q_{hx} \right). \quad (28)$$

Here, Q_{hx} and Q_{hy} are the covariance matrices of 1-D slices of the height coefficient array along the x- and y-directions, respectively. They have the same form as the jitter prior covariances Q_x and Q_y but different values for σ and ρ. Based on the statistics of the target scene, we choose $\sigma = 8$ meters and $\rho = 0.87$. Q_{hx} and Q_{hy} should share the same parameters (σ and ρ), but will differ in size if the height coefficient array is not square. The composite covariance Q_h is again nonsparse, but its inverse is sparse. Q_h^{-1} is a nine-banded array with six unique nonzero values. Since the inverse of a Kronecker product is the Kronecker product of the inverses, the inverse covariance can be written as

$$Q_h^{-1} = \left(\sigma Q_{hy}^{-1} \right) \otimes \left(\sigma Q_{hx}^{-1} \right). \quad (29)$$

Since the three parts of the state vector are independent (in the absence of measurements), the total prior can be written as the product $p(\theta) = p(\theta_x) p(\theta_y) p(\theta_h)$. The product and each of the three terms is a multivariate Gaussian with the form

$$p(\theta) = \text{Gaussian}(\bar{\theta}, Q) = \exp\left(-\frac{1}{2} (\log(|2\pi Q|) + (\theta - \bar{\theta})^T Q^{-1} (\theta - \bar{\theta})) \right). \quad (30)$$

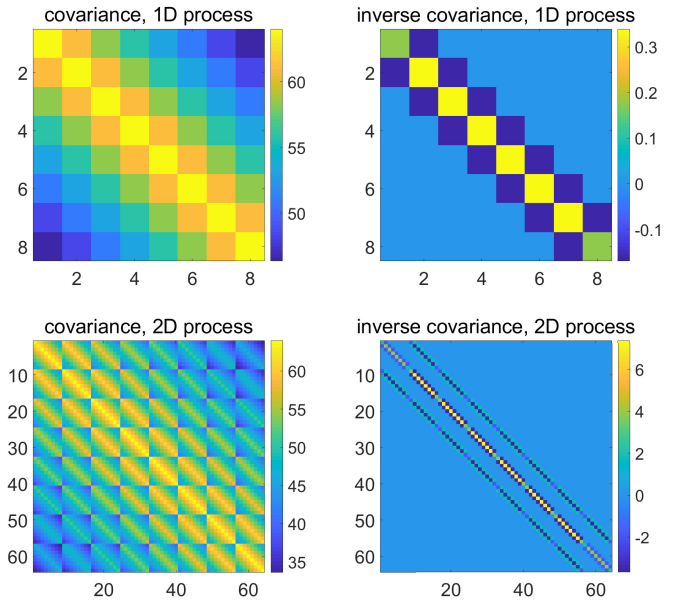

Fig. 4. Top row: Covariance and inverse covariance of eight samples of a 1-D first-order Gauss–Markov process (relevant to the jitter priors). Bottom row: Covariance and inverse covariance of an 8×8 sample array of a 2-D first-order Gauss–Markov process (relevant to the surface prior). Both inverse covariances are sparse.

C. EM Algorithm

The state estimation uses the EM algorithm [13]–[15] to deal with the uncertainty of whether each detection arose from noise or signal, in a robust way. "Noise" and "signal" are considered to be two components in a mixture pdf. The EM algorithm estimates the weights (prior probabilities) of the components and the parameters of each component. For this application, only the signal component has parameters to estimate. Since the noise component represents a Poisson process with a constant rate, it is uniform over a fixed interval. A possible extension to this algorithm could accommodate background light sources or nonuniform solar illumination by estimating the background as a function of spatial location. This would fit nicely into the EM framework and would result in the noise component having its own states. The algorithm described here is expected to be robust to spatially varying background, but it does not exploit it.

A key aspect of EM, as applied here, is the estimation of the "membership" of each measurement. This is a probability vector for each detection representing whether it is noise or signal. The steps in EM are as follows.

1) $\{w, \theta\} = initialize(\,)$.
2) $m = E\text{-}step(w, \theta)$.
3) $\{w, \theta\} = M\text{-}step(m, \theta)$.
4) If convergence criterion not met, go to step 2.

Here, w is the two-element component probability vector, θ is the estimated state vector, and m is the estimated membership array. Though not written explicitly, each step has access to the measurements and assorted parameters. After initialization, the E and M steps are cycled through repeatedly until a stopping criterion is met. The E-step estimates membership, using the simplifying assumption that the estimated state vector and component probabilities are the true values. Conversely, the M-step takes the membership as given and perturbs the state vector estimate to reduce a cost function. The membership weights the contribution of each detection to the cost function. To determine when to stop iterating, any of the following criteria or a combination of them can be used:

1) change in θ;
2) change in m;
3) change in the total likelihood of data (available in E-step);
4) number of iterations.

D. Initialization

Algorithm performance and speed are both improved by a careful initialization. The jitter states θ_x and θ_y are initialized as zeros. The two-element component weight vector w can be set to [0.5; 0.5] or can be based on the expected noise and signal rates. The height states θ_h are initialized by forming a height image without jitter compensation and then doing a fit to the image.

For each detection, the measured height is calculated as

$$h_k = -\frac{c}{2}(\zeta_k - t_{\mathrm{RX},i(k)}) \tag{31}$$

where $i(k)$ is defined as the index of the nearest pulse to the observed detection time, that is,

$$i(k) = \arg\min_i(|\zeta_k - t_{\mathrm{RX},i}|). \tag{32}$$

The expected ground sample coordinates are calculated as

$$\bar{x}_k = g_x(t_{\mathrm{RX},i(k)}) + o_x(n_k) \tag{33}$$

$$\bar{y}_k = g_y(t_{\mathrm{RX},i(k)}) + o_y(n_k) \tag{34}$$

Next, a 3-D histogram of the $(\bar{x}_k, \bar{y}_k, h_k)$ triplets is made. The "measured" height image is formed by taking the height bin with the most counts, for each (x, y) bin. The bin spacing could be selected based on the SNR, but for simplicity, the spacing can be chosen to match the 2-D grid on which the surface states are defined.

If the height image is defined over the same 2-D grid as the surface states, then an initial estimate of the surface states can be made by deconvolving the surface kernel from the height image. The deconvolution can be implemented via division in the frequency domain. An alternative approach for initialization is to do a least squares fit to the height image, with the measurement matrix representing convolution with the surface kernel. The least squares method is more general and allows the option of incorporating a prior for the surface states, but it is multiple orders of magnitude slower than deconvolution and has been found to be unnecessary.

E. E-Step

The purpose of the E-step is to update the membership of each detection. To define membership, first consider a discrete variable s_k that indicates whether a particular detection arose from noise or signal

$$s_k \in \{\text{noise}, \text{signal}\}. \tag{35}$$

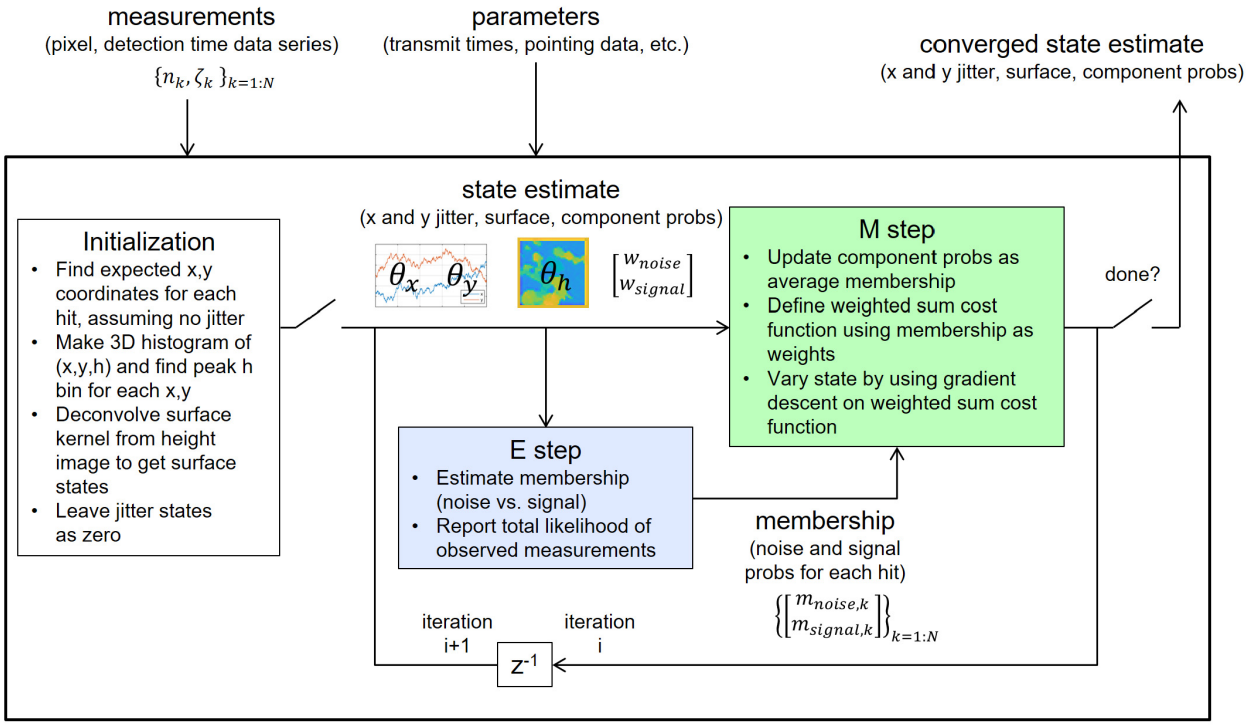

Fig. 5. EM algorithm flow. Inputs are {pixel index, detection time} pairs, and various parameters, including pulse transmit times and pointing instrumentation data. The algorithm iteratively refines estimates of the x and y jitter time series, a single-surface model of scene height versus x and y and the overall proportion of noise and signal in the data. Initialization provides a starting estimate of the height. The E-step estimates the membership of each hit (noise versus signal), and the M-step updates the estimated states by using gradient descent on a membership-weighted cost function that is closely related to the posterior pdf.

This indicator variable is not known, but can be estimated. Define the membership of detection k as a vector m_k that represents the conditional probability distribution of s_k

$$m_k = \begin{bmatrix} m_{\text{noise},k} \\ m_{\text{signal},k} \end{bmatrix} = \begin{bmatrix} P(s_k = \text{noise}|\zeta_k) \\ P(s_k = \text{signal}|\zeta_k) \end{bmatrix} = C_k \begin{bmatrix} p(\zeta_k|s_k = \text{noise})w_{\text{noise}} \\ p(\zeta_k|s_k = \text{signal})w_{\text{signal}} \end{bmatrix} \tag{36}$$

where C_k is a scalar that normalizes m_k such that the sum of its elements is one. w_{noise} and w_{signal} are the noise and signal component probabilities, respectively.

If the source is noise, then the detection time is simply uniform over the dwell time interval

$$p(\zeta_k|s_k = \text{noise}) = \frac{1}{T}. \tag{37}$$

If the source is signal, then the detection time is distributed as a Gaussian sum, assuming a Gaussian pulse shape. For pulses that are narrow relative to the pulse spacing, taking only the nearest Gaussian is a very good approximation

$$\begin{aligned} &p(\zeta_k|s_k = \text{signal}) \\ &= \sum_i u_i \exp\left(-\frac{1}{2}\left(\log\left(2\pi\sigma_{t,i}^2\right) + \left(\frac{\zeta_k - \bar{t}_i}{\sigma_{t,i}}\right)^2\right)\right) \\ &\approx u_{i(k)} \exp\left(-\frac{1}{2}\left(\log\left(2\pi\sigma_{t,i(k)}^2\right) + \left(\frac{\zeta_k - \bar{t}_{i(k)}}{\sigma_{t,i(k)}}\right)^2\right)\right). \end{aligned} \tag{38}$$

Another assumption here is that the height variation in the scene is significantly less than the pulse spacing. The weights of the Gaussian sum are the normalized receive energies

$$u_i = \frac{e_{\text{RX},i}}{\sum_i e_{\text{RX},i}}. \tag{39}$$

If the range is not changing significantly over the dwell time, and other things remain constant, then the weights would all be approximately equal to the reciprocal of the number of pulses within the dwell time interval.

The mean of the Gaussian distribution is

$$\bar{t}_{i(k)} = t_{\text{RX},i(k)} - \frac{2}{c}h(\bar{x}_k, \bar{y}_k; \theta_h) \tag{40}$$

where $\bar{x}_k$ and $\bar{y}_k$ are the mean coordinates of where the photon reflected off the surface (this time treating jitter as a known quantity)

$$\bar{x}_k = g_x(t_{\text{RX},i(k)}) + j_x(t_{\text{RX},i(k)}; \theta_x) + o_x(n_k) \tag{41}$$

$$\bar{y}_k = g_y(t_{\text{RX},i(k)}) + j_y(t_{\text{RX},i(k)}; \theta_y) + o_y(n_k). \tag{42}$$

The variance of the Gaussian distribution is the sum of the variance representing the Gaussian pulse shape and a term representing the broadening of the pulse due to the height variation over the lateral uncertainty of the measurement. The effect is that of convolving the pulse with the impulse response of the surface. The translation of lateral uncertainty into height uncertainty uses a local linear (planar) approximation to the surface, as shown in Fig. 6. Accordingly, the lateral variances

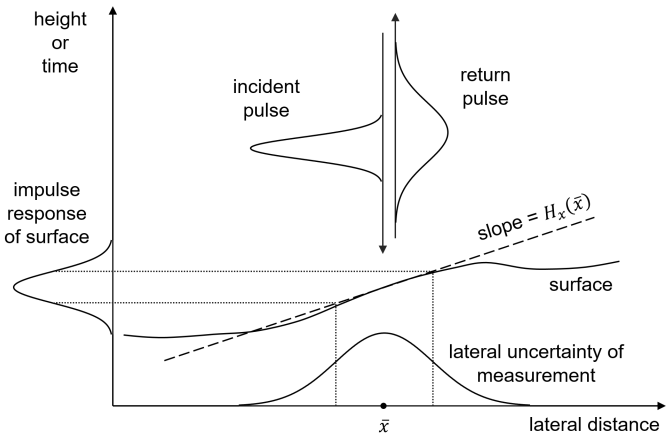

Fig. 6. Temporal pulse broadening caused by lateral uncertainty of the measurement across the single surface model. Lateral uncertainty is caused by not knowing where within a pixel the incoming photon arrived and the optical resolution of the receiver. The combination is approximated as a Gaussian, which is compressed or stretched by the slope of the surface, giving the impulse response of the surface. This convolves with the incident pulse to produce the return pulse.

are multiplied by the squared gradients of the height function. The resulting standard deviation of the detection time is

$$\sigma_{t,i(k)} \approx \sqrt{\sigma_{\mathrm{PW}}^2 + \left(\frac{2}{c}\right)^2 \left(H_x^2(\bar{x}_k, \bar{y}_k; \theta_h)\sigma_x^2 + H_y^2(\bar{x}_k, \bar{y}_k; \theta_h)\sigma_y^2\right)} \tag{43}$$

where H_x and H_y are gradients of the height function with respect to x and y, as given in the Appendix. The lateral variances are the sum of the variance of the uniformly distributed location of arrival and the variance of the optical blur

$$\sigma_x^2 = \frac{\Delta x_{\mathrm{pixel}}^2}{12} + \sigma_{\mathrm{blur}}^2 \tag{44}$$

$$\sigma_y^2 = \frac{\Delta y_{\mathrm{pixel}}^2}{12} + \sigma_{\mathrm{blur}}^2. \tag{45}$$

The pdf can alternately be expressed in terms of relative range rather than absolute time, by doing a change of variables

$$p(\zeta_k | s_k = \mathrm{signal}) \approx u_{i(k)} \frac{c}{2} \exp\left(-\frac{1}{2}\left(\log\left(2\pi\sigma_{r,i(k)}^2\right) + \left(\frac{\Delta r_k}{\sigma_{r,i(k)}}\right)^2\right)\right) \tag{46}$$

which is a function of the range difference and range uncertainty, given by

$$\Delta r_k = \frac{c}{2}(\zeta_k - \bar{t}_{i(k)}) = \frac{c}{2}(\zeta_k - t_{\mathrm{RX},i(k)}) + h(\bar{x}_k, \bar{y}_k, \theta_h) \tag{47}$$

$$\sigma_{r,i(k)} = \frac{c}{2}\sigma_{t,i(k)} = \sqrt{\left(\frac{c}{2}\sigma_{\mathrm{PW}}\right)^2 + H_x^2(\bar{x}_k, \bar{y}_k; \theta_h)\sigma_x^2 + H_y^2(\bar{x}_k, \bar{y}_k; \theta_h)\sigma_y^2}. \tag{48}$$

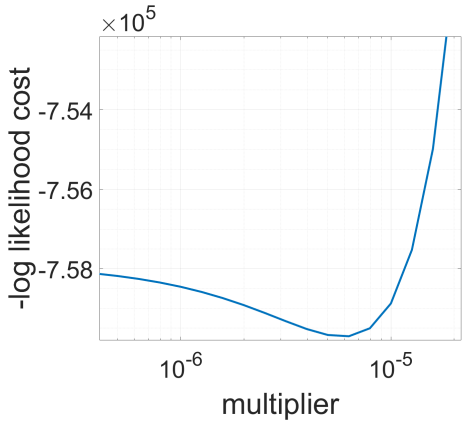

Fig. 7. Typical example of M-step cost function versus step size multiplier. The cost function is evaluated for an initial large step size multiplier. If this does not reduce the cost, then a sequence of decreasing logarithmically spaced step size multipliers is tried.

F. M-Step

The M-step updates the estimate of the state vector and component weights. The weights are updated by taking the maximum likelihood estimate, which is found by simply averaging the membership across all measurements

$$\begin{bmatrix} w_{\mathrm{noise}} \\ w_{\mathrm{signal}} \end{bmatrix} = \frac{1}{N}\sum_k \begin{bmatrix} m_{\mathrm{noise},k} \\ m_{\mathrm{signal},k} \end{bmatrix}. \tag{49}$$

In EM, each measurement can come from one of multiple components of a mixture pdf. In general, each component has a set of parameters that are updated in the M-step. In this application, only the signal component has parameters (i.e., elements of the state vector) to update.

The state vector estimate is updated by using gradient descent to reduce a weighted sum cost function, where the weights are the membership of the signal component. Gradient descent is used because of the nonlinearity of the measurement function and the high dimensionality of the state space. With ~5000 state elements (and potentially more for a scanning mode), a particle filter would be completely infeasible, and even a covariance matrix starts to become unwieldy. There are three reasons for settling for a reduction in the cost function rather than a minimization. The first is that there is no known closed-form solution. The second is that the cost function is relatively expensive to evaluate. The third is that the cost function becomes less valid as the distance from the current state estimate increases. The cost function gets refreshed by the E-step. Therefore, it makes sense to take smaller steps and update the cost function frequently. Reducing rather than minimizing the cost function in the M-step means this algorithm is technically generalized EM.

The cost function is a modified version of the negative log of the posterior probability density given that the source is signal. The modification, which is associated with the EM algorithm, is the weighting of the contribution of each measurement by its signal component membership $m_{\mathrm{signal},k}$. The cost function is

$$\Psi(\theta) = -\log(p(\theta)) - \sum_k m_{\mathrm{signal},k} \log(p(\zeta_k | \theta, s_k = \mathrm{signal})) \tag{50}$$

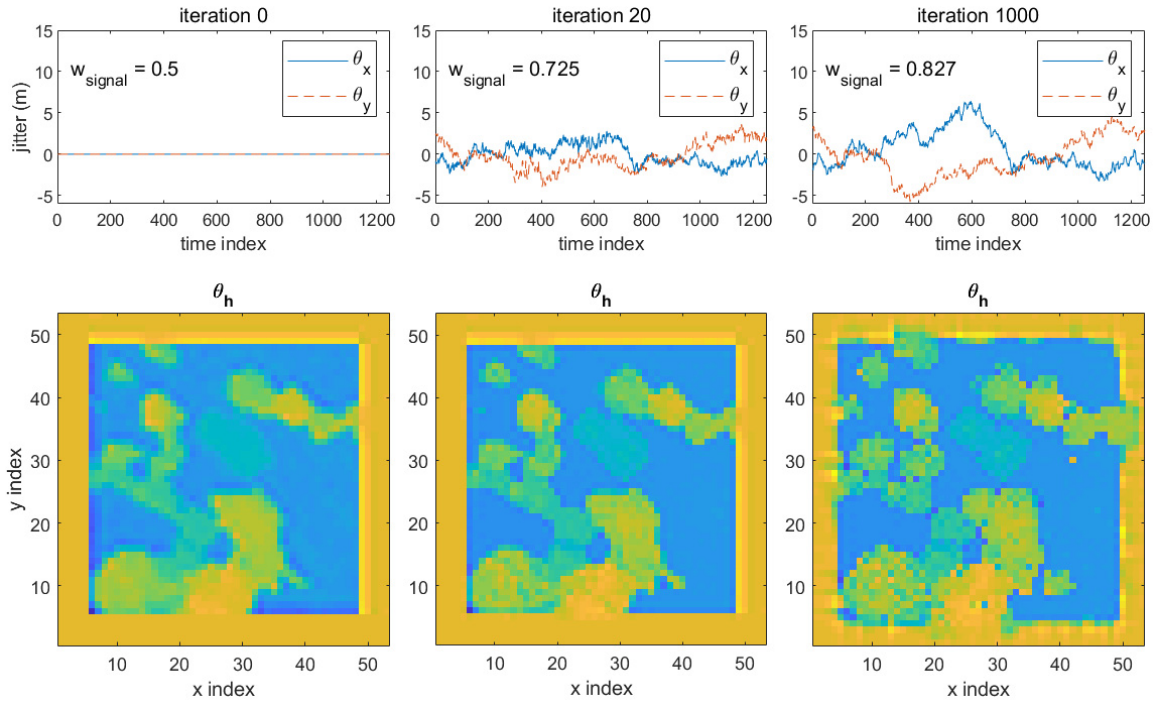

Fig. 8. Elements of the estimated state vector corresponding to x and y jitter, shown as time series, and height, shown as images. Estimates are shown at initialization (left), iteration 20—representing partial convergence (middle), and iteration 1000—representing full convergence (right). The signal-component probability is also displayed which increases as the algorithm "finds" more of the signal. Despite the coarse sampling of the height states and their indirect relationship to height, the images show scene objects becoming sharper and more resolved as the jitter and height states evolve simultaneously.

where $p(\theta)$ is the Gaussian prior pdf, as described previously. $p(\zeta_k|\theta, s_k = \text{signal})$ was also given previously in (46), but here, we make explicit the conditioning on the state vector.

Gradient descent is used to take a step that reduces the cost function. This requires the partial derivatives of the cost function with respect to the state

$$\frac{\partial \Psi}{\partial \theta} = \frac{\partial}{\partial \theta}(-\log(p(\theta))) + \sum_k m_{\text{signal},k} \frac{\partial}{\partial \theta}(-\log(p(\zeta_k|\theta, s_k = \text{signal}))). \quad (51)$$

The partial derivatives of the term representing the prior are

$$\begin{aligned}\frac{\partial}{\partial \theta}(-\log(p(\theta))) &= (\theta - \bar{\theta})^T Q^{-1} \\ &= (\theta_x - \bar{\theta}_x)^T Q_x^{-1} \frac{\partial \theta_x}{\partial \theta} + (\theta_y - \bar{\theta}_y)^T Q_y^{-1} \frac{\partial \theta_y}{\partial \theta} \\ &\quad + (\theta_h - \bar{\theta}_h)^T Q_h^{-1} \frac{\partial \theta_h}{\partial \theta} \end{aligned} \quad (52)$$

where $\bar{\theta}$ and Q are the mean and covariance of the prior, respectively. Due to sparsity, the prior and its partial derivatives can be evaluated without explicitly representing the covariance matrices or their inverses.

The partial derivatives in the sum are

$$\begin{aligned}&\frac{\partial}{\partial \theta}(-\log(p(\zeta_k|\theta, s_k = \text{signal}))) \\ &= -\frac{\partial}{\partial \theta} \log\left(u_{i(k)} \frac{c}{2}\right) \\ &\quad + \frac{1}{2}\frac{\partial}{\partial \theta}\left(\log\left(2\pi \sigma_{r,i(k)}^2\right) + \left(\frac{\Delta r_k}{\sigma_{r,i(k)}}\right)^2\right) \\ &= \frac{1}{\sigma_{r,i(k)}}\left(\left(1 - \left(\frac{\Delta r_k}{\sigma_{r,i(k)}}\right)^2\right)\frac{\partial \sigma_{r,i(k)}}{\partial \theta} + \frac{\Delta r_k}{\sigma_{r,i(k)}}\frac{\partial \Delta r_k}{\partial \theta}\right)\end{aligned} \quad (53)$$

where the partial derivatives of the range difference and range standard deviation are

$$\frac{\partial \Delta r_k}{\partial \theta} = \frac{\partial h}{\partial \theta} \quad (54)$$

$$\frac{\partial \sigma_{r,i(k)}}{\partial \theta} = \frac{1}{\sigma_{r,i(k)}}\left(\sigma_x^2 H_x \frac{\partial H_x}{\partial \theta} + \sigma_y^2 H_y \frac{\partial H_y}{\partial \theta}\right). \quad (55)$$

The height gradients and remaining partial derivatives are given in the Appendix. This completes the derivation of the gradient of the cost function.

Gradient descent is used to improve the state estimate by trying various step sizes in the opposite direction of the gradient. A step size multiplier α determines the new state vector, at which the cost function is evaluated. If i is the iteration number, a candidate state vector estimate is found as

$$\hat{\theta}_{\text{test}} = \hat{\theta}_i - \alpha \frac{\partial \Psi}{\partial \theta} \quad (56)$$

If the cost is improved, then the previous state estimate is replaced with the new one, and the M-step is complete

$$\hat{\theta}_{i+1} = \hat{\theta}_{\text{test}} \quad \text{if} \quad \Psi(\hat{\theta}_{\text{test}}) < \Psi(\hat{\theta}_i). \quad (57)$$

Otherwise, the next value of α in a descending logarithmically spaced list is tried.

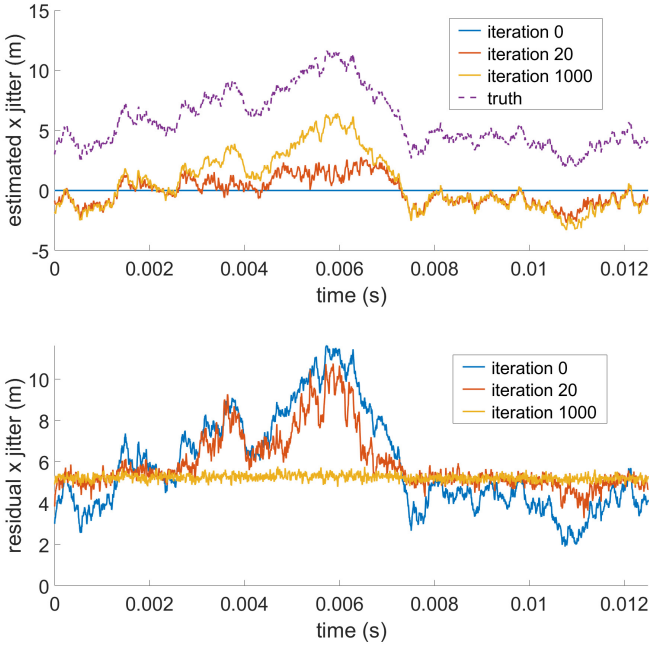

Fig. 9. Top: Estimated x jitter versus time, at initialization (blue), partial convergence (red), and full convergence (yellow), compared against truth (purple). Bottom: Residual x jitter (truth minus estimate) at the same three levels of convergence. At full convergence, the residual x jitter is reduced to an unobservable constant bias plus a small amount of mostly white noise.

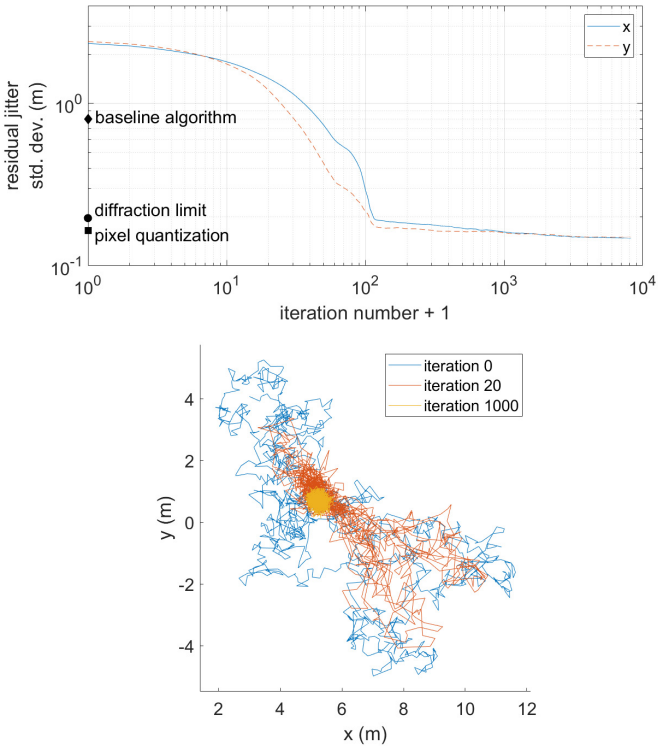

Fig. 10. Top: Standard deviation of residual jitter in x (blue) and y (red dashed) versus iteration number. This is the main result of this article since this metric neatly quantifies the blurring effect of residual jitter. At iteration 120, the standard deviation has been reduced by over an order of magnitude, to a similar size as the diffraction limit and the pixel quantization. Bottom: Residual jitter path at varying levels of convergence (initialization, partial, and full).

Fig. 7 shows the typical shape of the cost function versus α. This shape makes the optimization very stable. Too large of a step size increases the cost, and too small of a step size reduces the cost only slightly. A better minimum may exist for much larger step sizes, but we seek only to move toward the local minimum. It is not important to truly minimize the cost function at any given iteration of EM.

V. Experimental Results and Discussion

The algorithm was tested with simulated data that was generated with the parameters listed in Table I. The simulation used a high-resolution height map from the MACHETE airborne lidar of the vicinity of the Boston Common Visitor Center, Boston, MA, USA, to represent the true scene. The algorithm was written in vectorized MATLAB code, using the gpuArray functionality. To drastically improve computation time, when evaluating the height function or its gradients, only the kernels for the nearest 5×5 array of coefficients were evaluated. This is a good approximation because of the rapid decay of the 2-D Gaussian kernel. The measurements were also range-gated to isolate the returns within a 50-m interval centered on the scene. The code was run on an LLGrid node with an Intel Xeon E5 processor and NVIDIA Tesla K80 GPU. The algorithm was allowed to continue until it converged to within single-precision after 8208 iterations, running at about 8 iterations/s.

Fig. 8 shows the evolution of the jitter and height parts of the state vector as the algorithm progresses from initialization to convergence (calling 1000 iterations full convergence). The signal-component probability is also displayed and can be seen to increase with iterations, as the improved state estimates allow more signal to be pulled out of the noise. The jitter states start as all zeros and eventually acquire the precise shape of the jitter time series. Concurrently, the height states approach the shape of the scene, becoming sharper and more resolved. This is despite the height states not representing height directly, but rather the weights of the 2-D Gaussian kernels. The margin around the height images is necessary to allow for the extent of the jitter, which is only known statistically *a priori*.

Fig. 9 shows the evolution of the estimated and residual x-axis jitter, with multiple iterations plotted together for comparison. At full convergence, the estimated jitter matches the true jitter extremely well, aside from a small amount of mostly white noise and a constant bias, which represents a registration error. The constant bias is not observable from the data alone, but it could be observed if spatially varying surface height data is encoded in the prior of the height states, effectively allowing the algorithm to register against the prior knowledge of the surface. The registration problem can also be handled via the conventional method of image cross correlation. This would be a postprocessing step, and it would benefit from the greatly reduced blurring in its compensated input images.

For comparison, we also implemented a baseline algorithm that estimates the jitter via frame-to-frame registration. This approach divides the dwell time into a number of frames and forms a height image for each. The height images are formed via the same histogram-based approach used for initialization, with a spatial sampling interval of one GSD. The height image

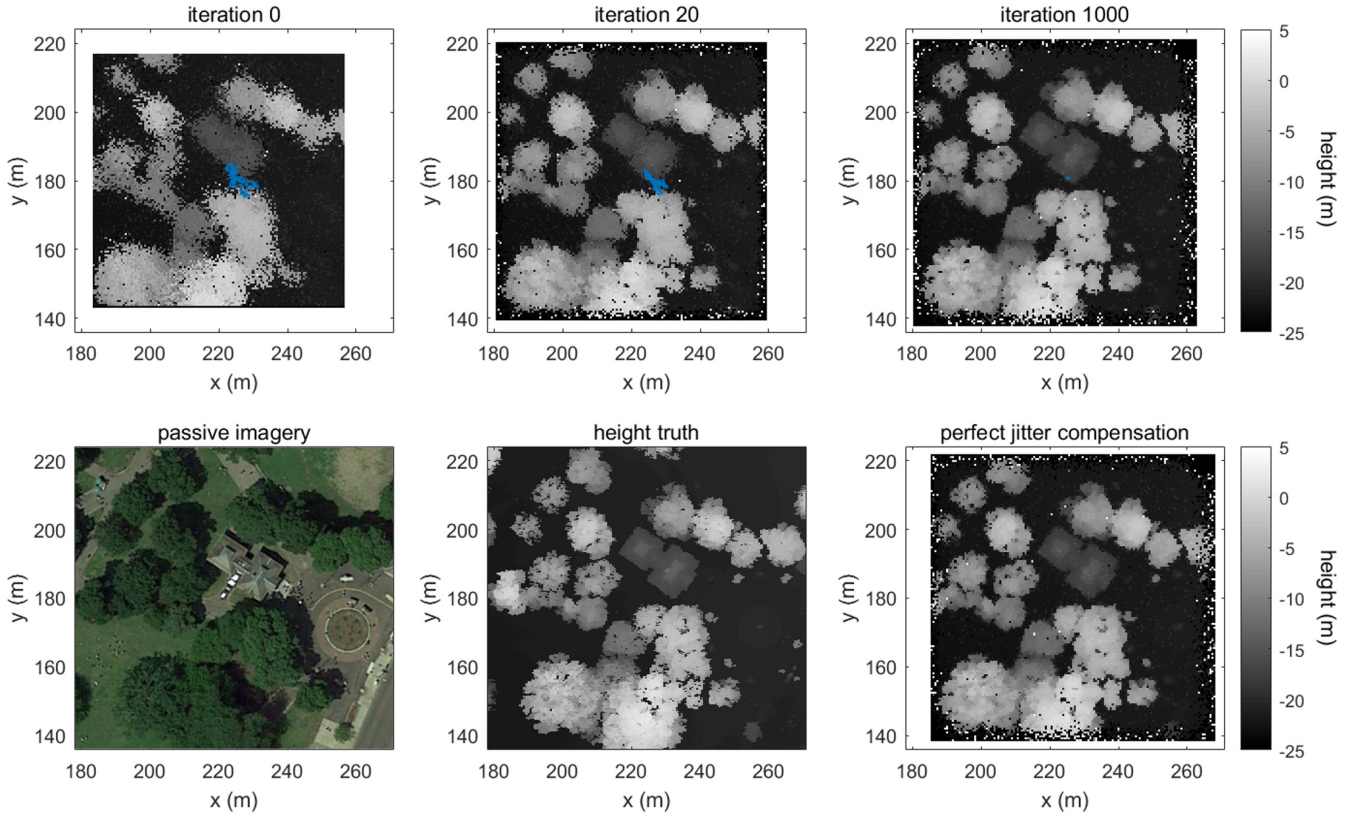

Fig. 11. Top row: Jitter-compensated height images, shown at varying levels of convergence (initialization, partial, and full), with an overlay of residual jitter (blue). Bottom row: Google Earth imagery, high-resolution height map used as truth, and jitter-compensated height image using knowledge of the true jitter. A simple histogram-based algorithm is used to form the height images. The jitter-compensated image at full convergence (top right) has greatly improved resolution and is very similar to the perfectly compensated image (bottom right), aside from the unobservable constant bias, which represents a registration error.

is then range-gated to a 40-m interval, which is approximately the height extent of the scene. Elements of the height image that are outside of the range gate are replaced with the mean of all elements within the range gate. Then, the mean height is subtracted from all elements. A cross correlation is performed for each pair of consecutive frames (using circular convolution for simplicity). The difference in jitter between the two frames is estimated as the shift that maximizes the cross correlation. The cumulative sum of these differences is taken and scaled by the GSD to produce the estimated jitter time sequence. Linear interpolation is used to evaluate this sequence at arbitrary times.

To find the most favorable parameters for the baseline algorithm, a parameter sweep was done over the number of frames to divide the dwell time into. The minimum standard deviation of the residual jitter was achieved with the dwell time divided into about 20 frames of size 625 μs or 125 pulses. With this optimal frame size, the standard deviation of the residual jitter for the baseline algorithm was about 0.8 m, which is an improvement of roughly a factor of three compared with no compensation.

Fig. 10 shows the main result of this article, which is the standard deviation over the dwell time of the residual jitter as the iterations progress. This metric nicely quantifies the blurring effect of residual jitter [8]. This is the angle-resolution-degrading quantity that adds in quadrature to the terms due to the diffraction limit and the pixel size. After reducing the residual jitter standard deviation for the x- and y-axes to about the same size (0.15–0.19 m depending on level of convergence) as the diffraction limit (0.20-m standard deviation) and the pixel width (0.16-m standard deviation), the jitter is no longer the limiting term for angular resolution. The algorithm's performance is about four times better than the baseline algorithm. The lower plot shows the x and y components of the residual jitter time series plotted against each other. These residual jitter paths are analogous to blur functions that get convolved with the true scene to produce degraded blurred height images. The converged residual jitter path is drastically smaller and more pointlike than the original.

For illustrative purposes, Fig. 11 shows jitter-compensated height images at varying levels of convergence. The same histogram method of image reconstruction is used here as was used for initialization, except that now the estimated jitter time series is used to shift the x and y coordinates of the measurements as a function of receive time. This is clearly not the most sophisticated image forming or coincidence processing algorithm, but it is simple and is adequate to demonstrate the effect of jitter compensation. For reference, below the compensated height images are images showing—passive imagery from Google Earth, the high-resolution height

map used as truth for simulating the measurements, and the result of perfectly jitter-compensating the measurements.

The uncompensated image (iteration 0) is extremely blurred, with the blurring together of multiple trees and the two sections of the building. The jitter path is overlaid for comparison and can be seen to span many pixels. At partial convergence (iteration 20) the image is much sharper, but still partially blurred. The fully converged image (iteration 1000) has drastically enhanced resolution and looks very similar to the perfectly compensated image that would result if the true jitter were known exactly. The residual jitter path (blur function) is collapsed to a point, smaller than one pixel. Small objects are resolved, which were not present in the uncompensated image. Notice that the compensated images expand as the jitter estimates improve, and the optical axis traces out a path parallel to the actual jitter. Also evident is the registration bias between the compensated images and the perfectly compensated image.

VI. Conclusion

Initial tests of the jitter-estimation algorithm have been very promising. Using the algorithm in a scenario in which the standard deviation of the combined two-axis jitter was about 6 times the pixel size, and pixel size matched the Rayleigh diffraction limit ($1.22\lambda/D$), caused the residual angular jitter to be reduced by over a factor of 10 in amplitude (100 in power). Relative to a baseline algorithm of frame-to-frame registration, the performance improvement was about a factor of 4 in amplitude (16 in power). The impact of the improved pointing knowledge was demonstrated by running a simple image reconstruction algorithm with and without compensating for the estimated jitter. The jitter-compensated image is much clearer and would be much more valuable for object recognition, whether performed by humans or computers.

The algorithm is about 1000 times slower than real-time (for 90% of full convergence) on the single computer used to run it. This can easily be improved by parallelizing over multiple time intervals or upgrading the computing hardware. Parameters, such as the step size in gradient descent, could be chosen more carefully, and other code improvements are possible. If the input jitter is smaller or if higher residual jitter can be tolerated, these can greatly reduce the required runtime. Even without any speed increase, the algorithm is practical to implement without huge expense and would be significantly enabling for a satellite-based lidar system.

Appendix
Gradients and Partial Derivatives

The E and M steps require the gradient of the height function, and the M step also requires partial derivatives of the jitter (17) and (18) and the height (20) functions with respect to the state. The jitter and height functions can be written in a compact form, as an inner product between a coefficient vector and part of the state vector

$$j_x = \Phi_j^T \theta_x \tag{58}$$

$$j_y = \Phi_j^T \theta_y \tag{59}$$

$$h = (\Phi_y \otimes \Phi_x)^T \theta_h. \tag{60}$$

The terms in the first and second gradients of the height function are

$$H_x = \frac{\partial h}{\partial x} = \left(\Phi_y \otimes \Phi_x'\right)^T \theta_h \tag{61}$$

$$H_y = \frac{\partial h}{\partial y} = \left(\Phi_y' \otimes \Phi_x\right)^T \theta_h \tag{62}$$

$$H_{xx} = \frac{\partial^2 h}{\partial x^2} = \left(\Phi_y \otimes \Phi_x''\right)^T \theta_h \tag{63}$$

$$H_{xy} = \frac{\partial^2 h}{\partial x \partial y} = \left(\Phi_y' \otimes \Phi_x'\right)^T \theta_h \tag{64}$$

$$H_{yy} = \frac{\partial^2 h}{\partial y^2} = \left(\Phi_y'' \otimes \Phi_x\right)^T \theta_h. \tag{65}$$

For the jitter functions, if the jitter time grid has L samples, the coefficient vector is

$$\Phi_j = \begin{bmatrix} \phi_{\text{jitter}}\left(\dfrac{t - t_{\text{grid}}(1)}{\Delta t_{\text{grid}}}\right) \\ \vdots \\ \phi_{\text{jitter}}\left(\dfrac{t - t_{\text{grid}}(L)}{\Delta t_{\text{grid}}}\right) \end{bmatrix}. \tag{66}$$

The coefficient vector for the height function is a Kronecker product of two vectors. If the grid of height coefficients is M by N, the two vectors and their first and second derivatives are

$$\Phi_x = \begin{bmatrix} \phi_G\left(\dfrac{x - x_{\text{grid}}(1)}{\Delta x_{\text{grid}}}\right) \\ \vdots \\ \phi_G\left(\dfrac{x - x_{\text{grid}}(M)}{\Delta x_{\text{grid}}}\right) \end{bmatrix} \tag{67}$$

$$\Phi_x' = \frac{1}{\Delta x_{\text{grid}}} \begin{bmatrix} \phi_G'\left(\dfrac{x - x_{\text{grid}}(1)}{\Delta x_{\text{grid}}}\right) \\ \vdots \\ \phi_G'\left(\dfrac{x - x_{\text{grid}}(M)}{\Delta x_{\text{grid}}}\right) \end{bmatrix} \tag{68}$$

$$\Phi_x'' = \frac{1}{\Delta x_{\text{grid}}^2} \begin{bmatrix} \phi_G''\left(\dfrac{x - x_{\text{grid}}(1)}{\Delta x_{\text{grid}}}\right) \\ \vdots \\ \phi_G''\left(\dfrac{x - x_{\text{grid}}(M)}{\Delta x_{\text{grid}}}\right) \end{bmatrix} \tag{69}$$

$$\Phi_y = \begin{bmatrix} \phi_G\left(\dfrac{y - y_{\text{grid}}(1)}{\Delta y_{\text{grid}}}\right) \\ \vdots \\ \phi_G\left(\dfrac{y - y_{\text{grid}}(N)}{\Delta y_{\text{grid}}}\right) \end{bmatrix} \tag{70}$$

$$\Phi_y' = \frac{1}{\Delta y_{\text{grid}}} \begin{bmatrix} \phi_G'\left(\dfrac{y - y_{\text{grid}}(1)}{\Delta y_{\text{grid}}}\right) \\ \vdots \\ \phi_G'\left(\dfrac{y - y_{\text{grid}}(N)}{\Delta y_{\text{grid}}}\right) \end{bmatrix} \tag{71}$$

$$\Phi_y'' = \frac{1}{\Delta y_{\text{grid}}^2}\begin{bmatrix} \phi_G''\left(\dfrac{y - y_{\text{grid}}(1)}{\Delta y_{\text{grid}}}\right) \\ \vdots \\ \phi_G''\left(\dfrac{y - y_{\text{grid}}(N)}{\Delta y_{\text{grid}}}\right)\end{bmatrix}. \tag{72}$$

The 1-D Gaussian kernel and its first and second derivatives are

$$\phi_G(w) = \exp\left(-\frac{1}{2}\left(\frac{w}{\sigma}\right)^2\right) \tag{73}$$

$$\phi_G'(w) = -\frac{w}{\sigma^2}\exp\left(-\frac{1}{2}\left(\frac{w}{\sigma}\right)^2\right) \tag{74}$$

$$\phi_G''(w) = \frac{w^2 - \sigma^2}{\sigma^4}\exp\left(-\frac{1}{2}\left(\frac{w}{\sigma}\right)^2\right) \tag{75}$$

where the width of the Gaussian kernel is set by $\sigma = 1/2$. The necessary partial derivatives of the jitter functions and the height function and its gradient are

$$\frac{\partial j_x}{\partial \theta} = \Phi_j^T \frac{\partial \theta_x}{\partial \theta} \tag{76}$$

$$\frac{\partial j_y}{\partial \theta} = \Phi_j^T \frac{\partial \theta_y}{\partial \theta} \tag{77}$$

$$\frac{\partial h}{\partial \theta} = H_x \frac{\partial j_x}{\partial \theta} + H_y \frac{\partial j_y}{\partial \theta} + \left(\Phi_y \otimes \Phi_x\right)^T \frac{\partial \theta_h}{\partial \theta} \tag{78}$$

$$\frac{\partial H_x}{\partial \theta} = H_{xx} \frac{\partial j_x}{\partial \theta} + H_{xy} \frac{\partial j_y}{\partial \theta} + \left(\Phi_y \otimes \Phi_x'\right)^T \frac{\partial \theta_h}{\partial \theta} \tag{79}$$

$$\frac{\partial H_y}{\partial \theta} = H_{xy} \frac{\partial j_x}{\partial \theta} + H_{yy} \frac{\partial j_y}{\partial \theta} + \left(\Phi_y' \otimes \Phi_x\right)^T \frac{\partial \theta_h}{\partial \theta}. \tag{80}$$

ACKNOWLEDGMENT

Mr. E. Phelps would like to thank Mr. E. Statz of MIT Lincoln Laboratory, Lexington, MA, USA, for judicious oversight of this research and many helpful discussions. The authors thank the reviewers for their valuable input.

REFERENCES

[1] M. A. Albota *et al.*, "Three-dimensional imaging laser radar with a photon-counting avalanche photodiode array and microchip laser," *Appl. Opt.*, vol. 41, no. 36, pp. 7671–7678, 2002.

[2] D. M. Gates, "Spectral distribution of solar radiation at the Earth's surface," *Science*, vol. 151, no. 3710, pp. 523–529, 1966.

[3] K. Murthy, M. Shearn, B. D. Smiley, A. H. Chau, J. Levine, and M. D. Robinson, "SkySat-1: Very high-resolution imagery from a small satellite," *Proc. SPIE*, vol. 9241, Oct. 2014, Art. no. 92411E.

[4] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Efficient marginal likelihood optimization in blind deconvolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 2657–2664.

[5] R. D. Fiete, "Image quality and [lambda] FN/p for remote sensing systems," *Proc. SPIE*, vol. 38, no. 7, pp. 1229–1241, 1999.

[6] A. Kirmani *et al.*, "First-photon imaging," *Science*, vol. 343, no. 6166, pp. 58–61, 2014.

[7] A. Gelb, *Applied Optimal Estimation*. Cambridge, MA, USA: MIT Press, 1974.

[8] D. S. Bayard, "State-space approach to computing spacecraft pointing jitter," *J. Guid., Control, Dyn.*, vol. 27, no. 3, pp. 426–433, 2004.

[9] M. Dow, "Explicit inverses of Toeplitz and associated matrices," *ANZIAM J.*, vol. 44, pp. 185–215, Jan. 2003.

[10] G. Meurant, "A review on the inverse of symmetric tridiagonal and block tridiagonal matrices," *SIAM J. Matrix Anal. Appl.*, vol. 13, no. 3, pp. 707–728, 1992.

[11] M. A. El-Shehawey, "On inverses of tridiagonal matrices arising from Markov chain-random walk I," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 2, pp. 497–508, 2008.

[12] A. Kavcic and J. M. F. Moura, "Matrices with banded inverses: Inversion algorithms and factorization of Gauss-Markov processes," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1495–1509, Jul. 2000.

[13] F. Dellaert, "The expectation maximization algorithm," Georgia Inst. Technol., Atlanta, GA, USA, Tech. Rep. GIT-GVU-02-20, 2002.

[14] G. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, vol. 382. Hoboken, NJ, USA: Wiley, 2007.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc., B (Methodol.)*, vol. 39, no. 1, pp. 1–22, 1977.

Ethan Phelps received the B.S. and M.S. degrees in electrical engineering from the University of Connecticut, Storrs, CT, USA, in 2000 and 2002, respectively.

He is currently with the Active Optical Systems Group, MIT Lincoln Laboratory, where he is involved in designing estimation and signal processing algorithms for lidar and optical systems.

Charles A. Primmerman received the B.A. degree in physics from Duke University, Durham, NC, USA, in 1968, and the M.S. and Ph.D. degrees in nuclear engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 1970 and 1975, respectively.

He is currently with the Space Systems and Technology Division, MIT Lincoln Laboratory. His research interests include high-energy lasers, laser radar, adaptive optics, and space-based electro-optical systems.

Dr. Primmerman is a fellow of the Optical Society of America.