Anisotropic Scattered Data Interpolation for Pushbroom Image Rectification

Erik Ringaby, Ola Friman, Per-Erik Forssén, Thomas Opsahl, Trym Vegard Haavardsholm, Ingebjørg Kåsen

***Abstract*—This article deals with fast and accurate visualization of pushbroom image data from airborne and spaceborne platforms. A pushbroom sensor acquires images in a line-scanning fashion, and this results in scattered input data that needs to be resampled onto a uniform grid for geometrically correct visualization. To this end, we model the anisotropic spatial dependence structure caused by the acquisition process. Several methods for scattered data interpolation are then adapted to handle the induced anisotropic metric and compared for the pushbroom image rectification problem. A trick that exploits the semi-ordered line structure of pushbroom data to improve the computational complexity several orders of magnitude is also presented.**

***Index Terms*—pushbroom, rectification, hyperspectral, interpolation, anisotropic, scattered data**

I. Introduction

Pushbroom scanners are common in multi- and hyperspectral imaging applications from satellite and airborne platforms. A pushbroom scanner has a linear array of sensor elements oriented perpendicular to the flight direction, see Fig. 1. Image data is acquired in a line-by-line fashion as the carrier platform moves forward and the footprint of each scan line therefore depends on the position, velocity, and orientation of the platform at the time of acquisition. Since the platform motion and the topography of the scene typically prevent neighboring scan lines from being parallel, a direct stacking of scan-lines generates a geometrically distorted image, see Fig. 2 top. To produce a geometrically correct *rectified* image, as shown in Fig. 2 bottom, two problems must be solved: (1) *georeferencing* and (2) *scattered data interpolation*. A georeferencing algorithm assigns a world coordinate to each acquired data point, e.g., in the standard WGS84 system [1], [2], [3], by projecting the point down to the ground. This can be done based on a camera model and navigation data from an on-board INS system [4]. Accurate georereferencing also requires a Digital Surface Model (DSM) to determine where the back-projection of a data point intersects the ground surface. The georeferenced pushbroom data constitutes a set of scattered or irregularly sampled points on the ground surface (i.e. each point has a longitude, a latitude, and an elevation). By interpolating this scattered data onto a uniform longitude-latitude sampling grid, a geometrically correct image is produced, see see Fig. 2 bottom.

E. Ringaby and P-E. Forssén are with the Department of Electrical Engineering, Linköping University, Sweden e-mail: ringaby@isy.liu.se

Ola Friman is with the Swedish Defense Research Agency, FOI.

T. Opsahl, T. Vegard Haavardsholm and I. Kåsen are with the Norwegian Defence Research Establishment (FFI), Instituttveien 20, 2027 Kjeller, Norway.

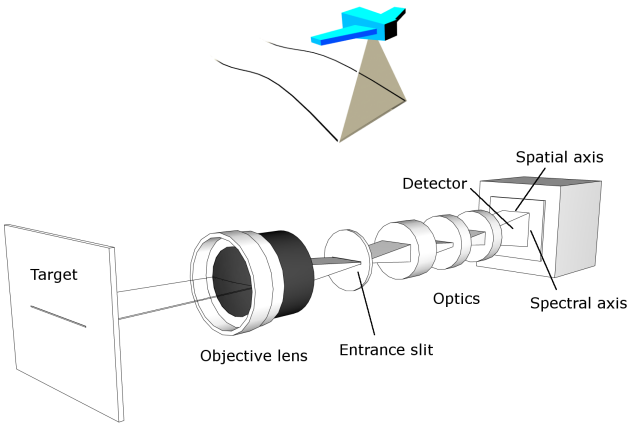

Fig. 1. Top: A pushbroom camera acquires images by scanning the ground surface in a line-wise fashion. Bottom: A hyperspectral pushbroom camera splits the light that enters through the entrance slit into different spectral bands using a dispersive element such as a prism.

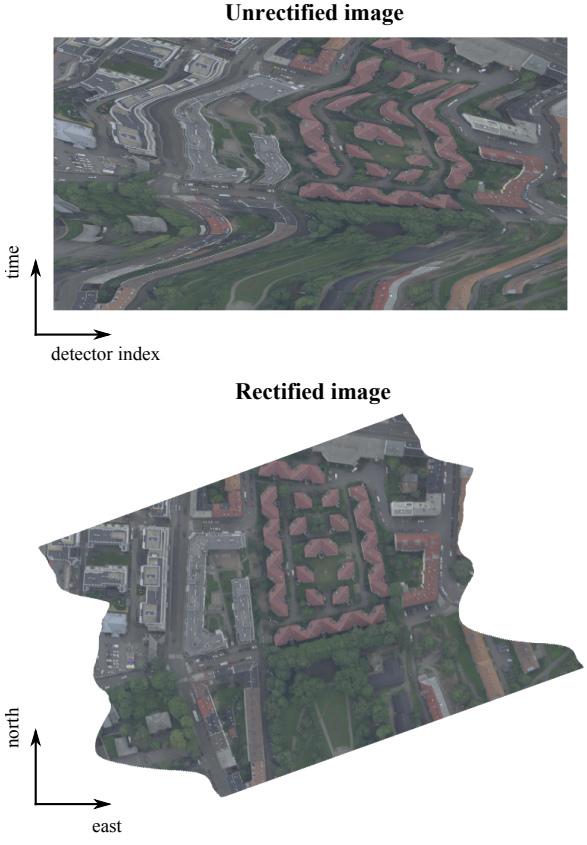

Fig. 2. Top: Example of an unrectified pushbroom swath over the city of Oslo. The image consists of stacked lines acquired over time. Bottom: Rectified version of the image in a world coordinate system.

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

The focus of this paper is on the scattered data interpolation problem. The general problem of scattered data interpolation has received much attention in the literature and several surveys are available [5], [6], [7]. Due to the varying sample density, scattered data interpolation is a much more challenging problem than interpolation of data in a uniform grid. There is also a computational aspect that is related to neighbor relations: in a uniform grid the nearest sample neighbors to an arbitrary point are quickly identified, whereas the distance to *all* sample points must be calculated in an irregularly sampled data set to find the closest sample neighbors to an arbitrary point. This difference leads to two different schemes of scattered data interpolation considered in this work: forward and inverse interpolation, see e.g. [8]. In a forward interpolation scheme, each sample point in the irregular input data is spread onto the neighbor locations in the uniform output grid. This is a fast operation as the sample neighbors are trivially located in the uniform output grid. The main disadvantage with forward mapping schemes is that it is difficult to guarantee that each point in the output grid receives any contribution, i.e., there is a risk that holes with undefined values are obtained in the interpolated image. Therefore, in image interpolation and resampling in general, inverse mapping schemes are preferred to ensure that each position in the output grid is assigned a value. An inverse mapping means that each point in the output grid is mapped back to the input domain where the interpolation takes place by a weighted sum of the neighboring input samples. However, when the input data is irregularly sampled, one is faced with the computational problem of identifying the neighbors, as discussed above. As pushbroom data sets typically are large, this computational aspect becomes an issue to consider.

This work evaluates a number of different techniques for interpolating pushbroom data to produce a rectified image. In the forward interpolation category, the Splatting method is evaluated [9]. This method performs the forward spreading of values using a radial basis function such as a Gaussian kernel. This technique has, for example, previously been used in correction of rolling shutter video [10], which utilizes an acquisition technique similar to the pushbroom acquisition. In the inverse interpolation category, Nearest Neighbor (NN), Inverse Distance Weighted (IDW), Kriging and triangulation-based interpolation methods are included in the comparison. NN interpolation simply uses the closest input sample as the interpolated value. IDW, also known as Shepard's method [11], calculates a weighted sum of the neighboring input samples with the weights equal to the inverse distances from the input samples to the point to interpolate. The Kriging interpolation similarly finds the weights by minimizing a reconstruction error in a least-squares sense, assuming knowledge of the spatial covariance function governing the statistical dependence between samples [12], [13], which typically decays monotonically with distance. Finally, in triangulation-based techniques, neighbor relations in the scattered input data are first established in a pre-processing step, e.g., using a Delaunay triangulation [7]. For triangulated data, interpolation can be made very efficient as the neighbor relations are given, but for large input data sets, the triangulation itself can be expensive. The triangulation-based technique evaluated here is the Natural Neighbor (NAT) method [14], [7]. In contrast to the previous inverse interpolation schemes, which weight input samples according to the distance to the point to interpolate, NAT uses an area-based measure to compute the weights. Other possible interpolation methods include thin-plate splines [15] and Non-Uniform-Rational-B-Spline (NURBS) surfaces [16]. These methods are typically used for smooth approximations in computer graphics and Computer Aided Design applications and require large linear equation systems to be solved, so they are not the first choice for the pushbroom rectification problem. Comparisons of methods for scattered data interpolation have been performed for different applications other than the pushbroom imaging one, see [17], [18] and references therein, with mixed conclusions as to the best method to apply. To the best of the authors' knowledge, an evaluation for the specific pushbroom imaging application has not been presented previously.

There are a number of contributions of the present work. First, the spatial dependence structure of pushbroom data is modeled and shown to be inherently anisotropic, i.e., data correlation is different in different directions. Second, five methods for scattered data interpolation are extended to handle the anisotropic nature of pushbroom data and compared for the image rectification problem. Third, a method that utilizes the semi-structured sampling pattern of a pushbroom sensor to significantly speed up inverse interpolation schemes is presented.

II. Problem description

The problem we consider in this work is to resample pushbroom data onto a uniform grid to obtain a geometrically correct rectified image. We assume that the pushbroom data has been georeferenced so that each pushbroom sample has an associated world coordinate in a coordinate frame for the Earth, e.g., the standard WGS84 system [1], [2], [3]. For a nadir looking sensor, the pushbroom samples are structured in an approximate line pattern, see Fig. 3. However, both the distance between points on a line and the distance between lines are non-uniform due to, among other factors, perspective geometry and variations in carrier platform velocity and attitude. The set of pushbroom points is therefore irregular. We denote a 2D position in world coordinates by $\mathbf{u} = (u_x, u_y)$ and the specific positions of the pushbroom samples are enumerated by a subscript $\mathbf{u}_i$. For each pushbroom point $\mathbf{u}_i$ there is a measured radiance value $z(\mathbf{u}_i)$. Multi- and hyperspectral pushbroom sensors generate multi-valued vectors with radiance values for different wavelength bands at each sample point. While both the spectral and spatial dimensions can be considered when interpolating new sample values, we focus on the spatial dimension in this work. The rectification is then performed for one wavelength band at a time.

The uniform output grid is user-defined. Typically, one defines the grid as the bounding box of all, or of a subset, of the pushbroom lines, and with a spatial spacing Δx and Δy of the same magnitude as the spacing between the input samples.

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

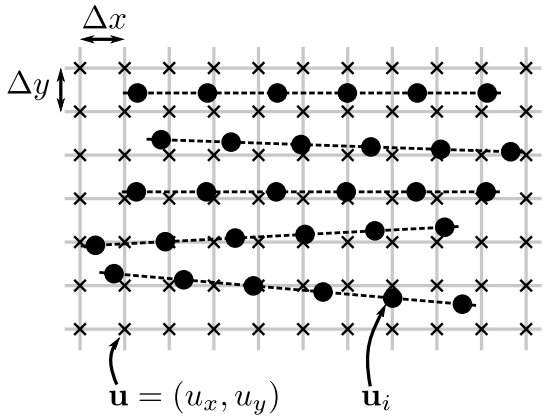

Fig. 3. Illustration of the scattered data interpolation problem. Irregularly spaced samples $\mathbf{u}_i$ (black dots) are located on pushbroom lines (dashed). The rectified image is produced by resampling onto the points in a regular output grid (crosses).

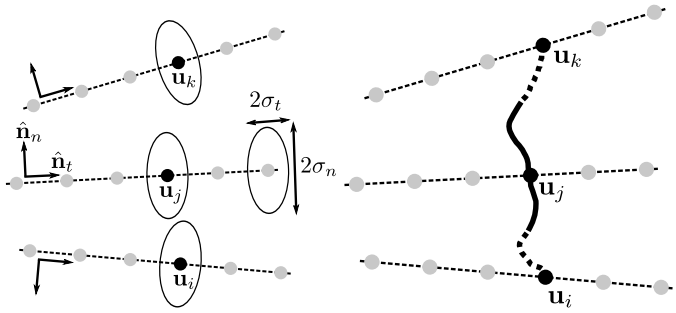

Fig. 4. Components in the pixel footprint function. Left: The optical component, $f_{IFOV}(\mathbf{u})$ is an anisotropic Gaussian function, here illustrated with an elliptic contour line, with size σ_n normal to the pushbroom line, and σ_t tangent to the line. Right: The motion component $f_M(\mathbf{u})$ is an integration over the ground plane trajectory between input sample locations in the output grid. The solid part of the curve corresponds to the integration of the sample at $\mathbf{u}_j$.

The rectified image is produced by interpolating values at the uniform grid points. We denote by $\hat{z}(\mathbf{u})$ the interpolated value at an arbitrary position $\mathbf{u}$.

III. Modeling

Successful data interpolation relies on a certain degree of data correlation that can be exploited to predict data values at arbitrary locations. This section investigates and models the dependence structure of pushbroom data in terms of the spatial autocovariance function

$$\rho(\Delta\mathbf{u}) = \mathrm{cov}(z(\mathbf{u}_i), z(\mathbf{u}_i - \Delta\mathbf{u})) \tag{1}$$

that describes the covariance between a pushbroom sample and a point at a distance $\Delta\mathbf{u}$. It is shown that the covariance function is both non-stationary and anisotropic, a fact that must be considered when interpolating new sample values in the rectified uniform grid. As we are working with georeferenced input coordinates, the unit of $\Delta\mathbf{u}$ is meters. For pushbroom and remote sensing images in general, covariance is induced by two main components:

1) Measurement covariance arises when sensor elements integrate light from the same surface region. A *pixel footprint function* (PFF) describes the region over which a sensor element integrates light.
2) Structural autocovariance of the ground surface reflectance $z_r(\mathbf{u})$ that one ultimately is interested in imaging.

While it is clear that the structural autocovariance generally is both non-stationary and locally anisotropic depending on the ground surface structure, it is shown below that the same also holds for the measurement covariance for pushbroom data. For multi-spectral imaging sensors, there may also be a non-negligible covariance along the spectral dimension that can be exploited to improve the interpolation. If the spectral sensitivity curves, i.e., the spectral footprints, of each band are known, the overlap between these defines a covariance in the measured data across spectral bands. In this work, however, the bands are treated as independent and interpolated separately.

In the sections below, we first derive a model of the pixel footprint function. We then introduce the surface structure model and finally combine these in a measurement model to obtain the total spatial covariance function $\rho(\Delta\mathbf{u})$ in (1).

A. Pixel footprint function

Each detector element in the pushbroom sensor integrates reflected light from a surface region described by a Pixel Footprint Function (PFF) $f(\mathbf{u})$. The PFF describes the relative contribution from each point on the ground surface to the measured sample value and it therefore integrates to 1. Note that the PFF is the reciprocal of the Point Spread Function (PSF), $p(\mathbf{x})$, that describes how light from a point source is imaged onto different sensor elements at positions determined by $\mathbf{x}$ in the image plane. Specifically, the PFF is the reprojection of the PSF onto the ground plane, assuming that the topography is locally flat. It is assumed that the georeferenced coordinate of each input data sample is located in the middle of its PFF. Input samples with overlapping PFFs receive light from the same ground area and will therefore be correlated.

For pushbroom sensors, the PFF is determined by an optical component and a motion component. The optical component is obtained directly from the sensor specification in terms of the so-called Instantaneous Field of View (IFOV), which is the angle subtended by each detector element in the pushbroom sensor. This angle defines a ground-projected pixel footprint at any given time. The IFOV may be different in the along-track and across-track directions of the carrier platform, i.e., pixel footprints are generally anisotropic. For example, the pushbroom sensor used in this work has an along-track IFOV angle that is twice that of the across-track angle. Although the IFOV typically is specified as a rectangle in angular space, it is modified by both the entrance slit and the lens of the sensor optics, see Fig. 1, bottom, which both cause diffraction. The diffraction pattern is described by the so-called Airy function, which is often approximated by a Gaussian function when describing the PSF [19]. The shape of the PFF defined by the IFOV can therefore also be modeled by a Gaussian function:

$$f_{IFOV}(\mathbf{u}) \propto e^{-\frac{1}{2}\mathbf{u}^T\mathbf{F}_{IFOV}^{-1}\mathbf{u}}, \tag{2}$$

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

where $\propto$ means that the scaling factor required for $f_{IFOV}(\mathbf{u})$ to integrate to 1 is omitted. For brevity and without loss of generality, we also assume that the point sample we consider is centered at the origin of the coordinate system. The potentially anisotropic form of the PFF is modeled by defining the metric tensor in (2) as

$$\mathbf{F}_{IFOV} = \sigma_t^2 \hat{\mathbf{n}}_t \hat{\mathbf{n}}_t^T + \sigma_n^2 \hat{\mathbf{n}}_n \hat{\mathbf{n}}_n^T, \tag{3}$$

with different extents σ_t and σ_n in the tangential and normal directions $\hat{\mathbf{n}}_t$ and $\hat{\mathbf{n}}_n$ of the pushbroom line respectively, see Fig. 4, left. The parameters σ_t and σ_n have the unit meters and they can be derived from the IFOV specification of the pushbroom sensor and the distance to the ground.

The second component affecting the pushbroom PFF is motion blur. This is caused by the integration time during which the shutter remains open while the carrier platform moves forward, and can be thought of as a function $f_M(\mathbf{u})$ that is non-zero along the scan trajectory of a particular sensor element, and that integrates to 1, see Fig. 4, right. A zeroth order approximation of $f_M(\mathbf{u})$ is a straight line of constant length l in the normal direction $\hat{\mathbf{n}}_n$ for all samples $\mathbf{u}_i$. To obtain a tractable expression for the total PFF, we further approximate this line with a degenerate Gaussian-shaped function

$$f_M(\mathbf{u}) \propto e^{-\frac{1}{2}\mathbf{u}^T \mathbf{F}_M^{-1} \mathbf{u}} \tag{4}$$

with a metric tensor defined to have no extent in the tangent direction so that it is only non-zero along a line

$$\mathbf{F}_M = 0\, \hat{\mathbf{n}}_t \hat{\mathbf{n}}_t^T + \sigma_l^2 \hat{\mathbf{n}}_n \hat{\mathbf{n}}_n^T. \tag{5}$$

The parameter σ_l is set so that the Gaussian function approximates a box function of length l, e.g., $\sigma_l = l/2$. The motion model can be extended to account for curved trajectories using a higher order approximation to account for sideway motions caused by turbulence. The metric tensor in (5) would then have some extent in the tangent direction too and still be applicable in the continued derivation below.

The total pushbroom PFF is now obtained as the IFOV PFF convolved with the motion PFF, using the fact that the convolution of two Gaussians yields another Gaussian:

$$f(\mathbf{u}) = (f_{IFOV} * f_M)(\mathbf{u}) \propto e^{-\frac{1}{2}\mathbf{u}^T \mathbf{F}^{-1} \mathbf{u}} \tag{6}$$

with

$$\mathbf{F} = \mathbf{F}_{IFOV} + \mathbf{F}_M = \sigma_t^2 \hat{\mathbf{n}}_t \hat{\mathbf{n}}_t^T + \left(\sigma_n^2 + \sigma_l^2\right) \hat{\mathbf{n}}_n \hat{\mathbf{n}}_n^T. \tag{7}$$

Hence, the forward motion of the carrier platform effectively stretches the footprint induced by the optics.

B. Surface structure model

The structure of the surface reflectance can be characterized by its autocovariance function $z_r(\mathbf{u})$. For homogeneous regions the function decays slowly and for rugged surfaces it has a faster decay. Along structures such as roads or roof edges, the covariance decays slowly along the structures and quickly across. Hence, the structural autocovariance in an image is generally both non-stationary and anisotropic. A Gaussian model of an anisotropic surface covariance structure is

$$\rho_s(\Delta\mathbf{u}) = \sigma_s^2\, e^{\Delta\mathbf{u}^T \mathbf{S}^{-1} \Delta\mathbf{u}}. \tag{8}$$

In this model, σ_s^2 represents the general magnitude of the variations on the ground surface. The potentially anisotropic covariance matrix $\mathbf{S}$ controls the smoothness of the surface and the autocovariance decays with spatial distance in a Gaussian-shaped fashion. Adaptive image processing approaches that try to adapt to the local structural autocovariance, for example edge-preserving filtering that filters along edges but not across them, have been suggested, e.g., steerable filters [20] and anisotropic diffusion [21]. These ideas were adapted to irregularly sampled data in [22] using alternating optimization of smoothing parameters and local structure.

In this work, a variant of the surface structure modeling method described in [22] is used. First, a structure tensor is estimated in each output pixel, $\mathbf{u}$, as the weighted outer product of gradients:

$$\mathbf{T} = \sum_{\mathbf{u}_k \in \mathcal{N}(\mathbf{u})} g(\mathbf{u}_k, \sigma) \nabla z(\mathbf{u}_k) \nabla z(\mathbf{u}_k)^T. \tag{9}$$

The neighborhood $\mathcal{N}(\mathbf{u})$ is a set of 7×7 neighbors determined in the input grid, and $g(\mathbf{u}, \sigma)$ is a Gaussian decay. Let $\mathbf{e}_1$, $\mathbf{e}_2$ be the eigenvectors and $\lambda_1 > \lambda_2 > 0$ the eigenvalues of $\mathbf{T}$ respectively. A surface structure covariance matrix is then constructed as

$$\mathbf{S} = \sigma_i^2\, \phi(\lambda_2) \left[\mathbf{I} - \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2} \mathbf{e}_1 \mathbf{e}_1^T\right], \tag{10}$$

where σ_i^2 is a global scaling factor and

$$\phi(\lambda) = \begin{cases} 1 - \lambda/\lambda_{max} & \text{if } \lambda \le \lambda_{max}, \\ 0 & \text{if } \lambda > \lambda_{max}. \end{cases} \tag{11}$$

The function $0 \le \phi(\lambda) \le 1$ is close to 0 when there is much surface structure, as indicated by a large λ_2, leading to a low overall covariance. If λ_2 instead is small there is at least one direction in which the covariance should be large. The parameter λ_{max} indicates the edge strength in the image above which the covariance should be zero. For the data in this work, which is restricted to the range [0,1], $\lambda_{max} = 0.05$ was a good value. The factor $\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}$ is close to 0 for the isotropic surface structure case $\lambda_1 \approx \lambda_2$, leading to an isotropic $\mathbf{S}$, but at edges where $\lambda_1 >> \lambda_2$ this factor is close to 1, indicating a strong anisotropic structure in the $\mathbf{e}_1$ direction along which the covariance should be low. For reference, an isotropic surface structure covariance is also considered in the experiments

$$\mathbf{S} = \sigma_i^2 \mathbf{I}. \tag{12}$$

C. Total pushbroom sample covariance

Using the notation introduced above, i.e., $f(\mathbf{u})$ for the Pixel Footprint Function and $z_r(\mathbf{u})$ for the true surface image, a pushbroom data sample is generated from the light integrated by a sensor element plus a noise term:

$$z = \int_{\mathbb{R}^2} f(\mathbf{u}) z_r(\mathbf{u})\, d\mathbf{u} + \epsilon. \tag{13}$$

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

The goal is to find the autocovariance function $\rho(\Delta\mathbf{u})$ in (1) of such a sample. By inserting (13) into (1) and simplifying the following expression is obtained

$$\rho(\Delta\mathbf{u}) = C\, e^{-\Delta\mathbf{u}^T(\mathbf{F}+\mathbf{S})^{-1}\Delta\mathbf{u}} + \sigma_\epsilon^2\delta(\Delta\mathbf{u}), \qquad (14)$$

where C is a constant that is determined by σ_t, σ_n, σ_l, σ_s and σ_i as introduced in the previous sections, and σ_ϵ^2 denotes the noise variance.

For data interpolation, it is generally sufficient to use the normalized autocorrelation function $r(\Delta\mathbf{u}) = \rho(\Delta\mathbf{u})/\rho(\mathbf{0})$ which becomes

$$r(\Delta\mathbf{u}) = \underbrace{\frac{C}{C+\sigma_\epsilon^2}}_{\text{SNR}} \underbrace{e^{-\Delta\mathbf{u}^T(\mathbf{F}+\mathbf{S})^{-1}\Delta\mathbf{u}}}_{\text{Anisotropic decay}} \text{ for } \Delta\mathbf{u} \neq \mathbf{0}. \qquad (15)$$

The autocorrelation function consists of two parts: a constant factor that is related to the Signal-To-Noise (SNR) level and a factor that describes how the correlation decays with spatial distance. The decay is determined by the metric tensor

$$\mathbf{M} = \mathbf{F} + \mathbf{S}\,, \qquad (16)$$

i.e., a combination of the PFF and the ground surface structure. The metric describes an anisotropic correlation structure that depends on the local image edges and the current motion of the aircraft carrying the pushbroom sensor. This result is used in the next section to perform anisotropic interpolation of the pushbroom data.

IV. Anisotropic scattered data interpolation

In section II it was established that pushbroom image rectification requires an interpolation from the irregular pushbroom samples onto a uniform grid. In section III it was furthermore established that the spatial dependence structure of pushbroom data is inherently anisotropic. In this section, different scattered data interpolation schemes known in the literature are adapted to take this local anisotropy into account. Five different methods are evaluated, one based on forward interpolation and four based on inverse interpolation. The pushbroom interpolation problem is illustrated in Fig. 5, where the value at the point $\mathbf{u}$ in the uniform output grid is predicted using the neighboring input samples $\mathbf{u}_1$, $\mathbf{u}_2$, ….

A. Anisotropic metric

To account for the fact that pushbroom data is differently correlated in different directions, we introduce the Mahalanobis distance metric

$$d_{\mathbf{M}}(\mathbf{u},\mathbf{v}) = \sqrt{(\mathbf{u}-\mathbf{v})^T\mathbf{M}^{-1}(\mathbf{u}-\mathbf{v})}, \qquad (17)$$

where $\mathbf{u}$ and $\mathbf{v}$ are two arbitrary points in space and $\mathbf{M}$ is a metric tensor that defines a possibly anisotropic distance measure. The interpolation methods outlined below are valid for any choice of metric tensor, but for the pushbroom interpolation case we use the tensor defined in (16), which has its main axes oriented relative to a pushbroom line, i.e., along the normal and tangential directions $\hat{\mathbf{n}}_n$ and $\hat{\mathbf{n}}_t$. We introduce a shorthand notation for the distance from a pushbroom input sample point $\mathbf{u}_i$ to any arbitrary point $\mathbf{u}$

$$d_i(\mathbf{u}) = d_{\mathbf{M}_i}(\mathbf{u}_i,\mathbf{u})\,. \qquad (18)$$

The orientation and degree of anisotropy defined by $\mathbf{M}_i$ vary between the pushbroom lines depending on the speed and trajectory of the carrier platform. Hence, the degree of anisotropy varies over the image. Therefore, if $\mathbf{M}_i$ and $\mathbf{M}_j$ are different, there is also an asymmetric distance relationship $d_i(\mathbf{u}_j) \neq d_j(\mathbf{u}_i)$.

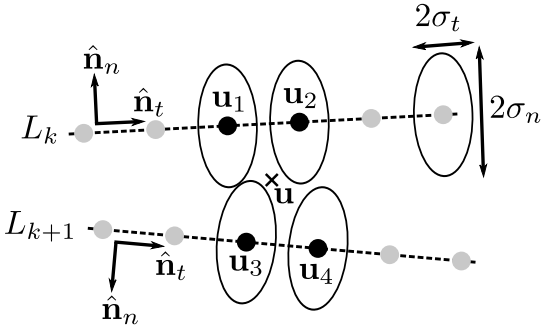

Fig. 5. Samples on two consecutive pushbroom lines denoted by L_k and L_{k+1}. The goal is to predict the value at $\mathbf{u}$ given the measured values on the pushbroom lines $\mathbf{u}_1$, $\mathbf{u}_2$, …. The ellipses illustrate the anisotropic data dependence as a contour curve of a Gaussian function which is aligned with the corresponding pushbroom lines.

B. Forward interpolation

In *forward interpolation* [10], also known as *Splatting* [9], each input pushbroom sample is spread (or splatted) out onto the neighborhood points in the uniform output grid. The contribution from each input sample point $z(\mathbf{u}_i)$, is accumulated iteratively in the output grid points

$$\mathbf{y}(\mathbf{u}) \leftarrow \mathbf{y}(\mathbf{u}) + \begin{bmatrix} w_i(\mathbf{u})z(\mathbf{u}_i) \\ w_i(\mathbf{u}) \end{bmatrix}, \qquad (19)$$

where $w_i(\mathbf{u})$ is a weight that depends on the distance between the irregular sample location $\mathbf{u}_i$ and the uniform grid location $\mathbf{u}$. Here we use the function

$$w_i(\mathbf{u}) = \begin{cases} e^{-d_i^2(\mathbf{u})} & \text{if } \mathbf{u} \in \mathcal{N}_K(\mathbf{u}_i) \\ 0 & \text{otherwise,} \end{cases} \qquad (20)$$

which splats each input sample with a Gaussian anisotropic shape defined by the metric tensor $\mathbf{M}_i$, cf. (18). To limit the computational effort, each input sample is only splatted onto the output points in a square neighborhood $\mathcal{N}_K(\mathbf{u}_i)$ parameterized by the side K, i.e., the Gaussian in (20) is truncated. Here we set K so that at least 95% of the Gaussian is included within the splatting kernel:

$$e^{-\frac{K^2}{2\sigma_{max}^2}} < 0.05 \;\Rightarrow\; K > \sqrt{-2\sigma_{max}^2 \ln 0.05}\,, \qquad (21)$$

where σ_{max} is the largest value of σ_n and σ_t. After the accumulation in (19), the interpolated value $\hat{z}(\mathbf{u})$ is found after normalization with the second element of $\mathbf{y}(\mathbf{u}) = [y_1(\mathbf{u}), y_2(\mathbf{u})]^T$:

$$\hat{z}(\mathbf{u}) = y_1(\mathbf{u})/y_2(\mathbf{u})\,. \qquad (22)$$

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

That is, the predicted value $\hat{z}(\mathbf{u})$ is a weighted average of the contributions of the input samples.

The advantage of the forward interpolation is that it is computationally efficient. A main drawback is that holes with undefined values are created in the output image if the neighborhood $\mathcal{N}_K(\mathbf{u}_i)$ is too small.

C. Inverse interpolation

In an inverse interpolation scheme, an interpolated value is calculated as

$$\hat{z}(\mathbf{u}) = \sum_{\mathbf{u}_i \in \mathcal{N}(\mathbf{u})} w_i \, z(\mathbf{u}_i), \tag{23}$$

where w_i represent weights, $z(\mathbf{u}_i)$ is an input sample and $\mathcal{N}(\mathbf{u})$ denotes a neighborhood around $\mathbf{u}$. In principle, $\mathcal{N}(\mathbf{u})$ can encompass the entire space so that all input samples contribute to the interpolated value, but smaller localized neighborhoods are generally chosen. In this work, the four closest input samples are typically used, see Fig. 5. The weights w_i are determined as a function of distance (17). Without loss of generality, let us assume that we found n input points in the neighborhood $\mathcal{N}(\mathbf{u})$ and denote them $\mathbf{u}_1, \ldots, \mathbf{u}_n$. Different methods have been suggested for choosing the weights $w_1, \ldots, w_n$ in (23). In this work, the *Nearest Neighbor*, *Inverse Distance Weighted*, *Natural Neighbors*, and *Kriging* interpolation schemes are extended to anisotropic interpolation and compared for the pushbroom image rectification.

The inverse interpolation scheme guarantees that each point in the output grid is assigned a predicted value. The main disadvantage of inverse interpolation schemes for irregular inputs is the computational complexity of determining the neighboring input points $\mathbf{u}_i \in \mathcal{N}(\mathbf{u})$. This section is concluded with a trick that exploits the semi-structured nature of pushbroom data to reduce the computational workload by many orders of magnitude. It can also be noted that inverse interpolation schemes are trivially parallelized as the points in the output grid are calculated independently of each other through (23).

1) Nearest Neighbor interpolation: In NN interpolation, only the closest input sample is considered,

$$\min_i d_i(\mathbf{u}), \tag{24}$$

for which the weight is set to 1 in (23). Note that the anisotropic distance from the input samples is used. Another characteristic of the NN interpolation is that the same neighbor will be the closest regardless of the size of the isotropic component of the metric tensor $\mathbf{M}$; only the anisotropic part of $\mathbf{M}$ can yield a different neighbor.

2) Inverse Distance Weighted interpolation: In the IDW interpolation, also known as Shepard's method [11], the weights in (23) are set to

$$w_i = \frac{\gamma}{d_i(\mathbf{u})^2}, \tag{25}$$

where γ is a normalization factor chosen so that $\sum w_i = 1$. Just as for the NN interpolation, only changes to the anisotropic part of $\mathbf{M}$ give different interpolation results. Changes to the isotropic magnitude of $\mathbf{M}$ will not change the weights w_i.

3) Natural Neighbors interpolation: Natural neighbors [14], [7] (NAT) is a technique originally developed for solving partial differential equations on irregular grids [14]. In most inverse interpolation schemes, the weights in (23) are based on distances. NAT instead uses an area-based measure to compute the weights. The first step of the algorithm is to triangulate all input samples using the Delaunay algorithm, from which then the Voronoi tessellation is computed, as shown in Fig. 6 left. Next, the output interpolation point $\mathbf{u}$ is added to the set and

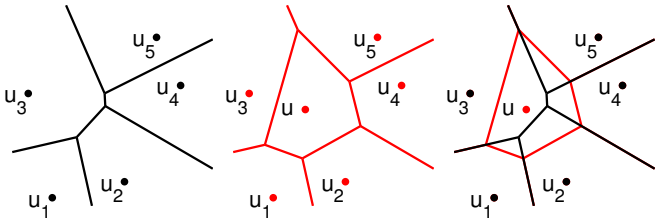

Fig. 6. Illustration of natural neighbors interpolation. Left to right: Voronoi cells for $\{\mathbf{u}_i\}$, Voronoi cells for $\{\mathbf{u}_i\} \cup \mathbf{u}$, intersection of the two.

the Voronoi tessellation is updated, as shown in Fig. 6 middle. Finally, the intersection of the regions in the two tessellations is used to find areas

$$a_i = \texttt{area}(\texttt{region}(\mathbf{u}_i) \cap \texttt{region}(\mathbf{u})), \tag{26}$$

one for each input data point, as shown in Fig. 6 right. Using these, the weights in (23) are computed as:

$$w_i = a_i / \left(\sum_{\mathbf{u}_k \in \mathcal{N}(\mathbf{u})} a_k \right). \tag{27}$$

The use of an area-based weight computation in NAT was introduced to better handle highly irregular sampling densities [7]. For example, having many input samples on one side of the output sample will lead to a bias which area based approaches automatically compensates for. As NAT relies on an initial triangulation with a global metric, there is no straightforward modification of the NAT method to make it take a local anisotropic metric into account.

4) Kriging interpolation: Kriging interpolation in general uses the covariance function $\rho(\mathbf{u}_i, \mathbf{u}_j)$ between sample locations to derive the optimal weights in (23) in a Best Linear Unbiased Estimator (BLUE) sense [23]. The covariance between two points typically decreases with the spatial distance so that the covariance also can be seen as a measure of distance between points. The so-called ordinary Kriging equation is used here for finding the interpolation weights:

$$\begin{bmatrix} w_1 \\ \vdots \\ w_n \\ -\mu \end{bmatrix} = \begin{bmatrix} \rho(\mathbf{u}_1, \mathbf{u}_1) & \cdots & \rho(\mathbf{u}_1, \mathbf{u}_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \rho(\mathbf{u}_n, \mathbf{u}_1) & \cdots & \rho(\mathbf{u}_n, \mathbf{u}_n) & 1 \\ 1 & \cdots & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \rho(\mathbf{u}_1, \mathbf{u}) \\ \vdots \\ \rho(\mathbf{u}_n, \mathbf{u}) \\ 1 \end{bmatrix} \tag{28}$$

The covariances between the input points $\rho(\mathbf{u}_i, \mathbf{u}_j)$ capture the sampling density, and the covariances with the interpolation point $\rho(\mathbf{u}_i, \mathbf{u})$ can be interpreted as distances. The additional

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

parameter μ is a nuisance parameter that is not used but which is required in (28) to ensure that $\sum w_i = 1$. A Gaussian covariance function that uses the anisotropic metric defined in (18) is used here as follows:

$$\rho(\mathbf{u}_i, \mathbf{u}_j) = e^{-d_i^2(\mathbf{u}_j)}. \tag{29}$$

D. Parameter tuning for pushbroom interpolation

To make a fair comparison of the different interpolation methods, the parameters of each method are tuned using a procedure described in this section. The tuned parameters include σ_i which controls the surface structure covariance, as well as σ_n and σ_t which control the pixel footprint size. These parameters are grouped into a parameter vector $\mathbf{p} = (\sigma_i, \sigma_n, \sigma_t)$. Optimal parameter values are determined using a leave-one-out cross-validation scheme: For a certain choice of $\mathbf{p}$, one actual sample $z(\mathbf{u}_i)$ is removed from the pushbroom data set. The predicted value $\hat{z}(\mathbf{u}_i)$ at this position is then interpolated using the remaining samples in the data set. By repeating this for many samples the interpolation accuracy can be measured. In this work, the relative error is used as accuracy measure

$$\varepsilon(\mathbf{p}) = \frac{1}{|\mathcal{E}|} \sum_{i \in \mathcal{E}} \frac{|\hat{z}(\mathbf{u}_i) - z(\mathbf{u}_i)|}{z(\mathbf{u}_i)}, \tag{30}$$

where $|\mathcal{E}|$ is the size of the evaluation set. Using this cross-validation error, one can search for the parameter vector $\mathbf{p}^*$ that gives the best interpolation accuracy. To find values close to the optimal ones, a brute-force grid search in reasonable intervals of each parameter is carried out. The final tuning is then made using a non-linear Nelder-Mead simplex optimization with the parameters found in the coarse search as starting point.

The parameters $\{\sigma_i, \sigma_n, \sigma_t\}$ do not have independent influences on the accuracy measure in (30). Specifically, the surface structure component $\mathbf{S}$ and the pixel footprint component $\mathbf{F}$ will both likely contain an isotropic part. The parameter optimization described above may therefore choose to put the isotropic part in $\mathbf{S}$ by adjusting the magnitude of σ_i, or in $\mathbf{F}$ by adjusting the magnitudes of σ_n and σ_t, i.e., the parameter set $\{\sigma_i, \sigma_n, \sigma_t\}$ results in almost the same structural covariance as $\left\{\sqrt{\sigma_i^2 + a}, \sqrt{\sigma_n^2 - a}, \sqrt{\sigma_t^2 - a}\right\}$, as long as $a < \sigma_n^2$ and $a < \sigma_t^2$. Note that the combined covariance $\mathbf{M} = \mathbf{F} + \mathbf{S}$ in (16), which ultimately is used for the interpolation, remains the same with either choice, so this becomes a problem of parameter interpretability rather than a problem of using the found parameters for the rectification. Nevertheless, one should use sample points at strong surface structures for the cross-validation in the optimization, as these carry more information about $\mathbf{F}$ and $\mathbf{S}$ than samples on isotropic surfaces. In practice, we do this by sorting the pixels in $\mathcal{E}$, according to their gradient strength, i.e., the trace of the structure tensor $\mathrm{tr}(\mathbf{T})$ computed according to (9). We then use only the 10% of pixels with the largest gradient values during parameter tuning. When reporting evaluation scores, however, the entire evaluation set $\mathcal{E}$ is used.

Finally, to compare results of isotropic and anisotropic interpolation, parameters for purely isotropic interpolation that do not account for surface structure of anisotropic footprints are also optimized. For this optimization, σ_n and σ_t are set to zero, and σ_i is used to define $\mathbf{S}$ according to (12).

V. Computational speedup for inverse interpolation of pushbroom data

In regular image resampling on uniform grids, the neighbors in $\mathcal{N}(\mathbf{u})$ are immediately given by the grid structure. When the input is irregularly sampled, a naive implementation must compute the distance from the interpolation point $\mathbf{u}$ to *all* input points $\mathbf{u}_i$ to determine the neighbors in $\mathcal{N}(\mathbf{u})$. If the number of input points is large this becomes computationally very expensive. Below, an approach is presented that exploits the semi-regular structure of pushbroom data to speedup the neighbor-finding procedure. The key idea is to approximate each line of pushbroom samples L_k in a parameterized standard line equation form

$$L_k : A_k x + B_k y + C_k = 0. \tag{31}$$

The parameters A_k, B_k and C_k can be found by a least-squares fit or simply by connecting the first and last points on the pushbroom line. Using this parameterization, we can efficiently

1) Identify the closest pushbroom line to $\mathbf{u}$.
2) Immediately predict the closest input samples on this line to $\mathbf{u}$ by assuming that the samples are equidistant on the line.

The shortest orthogonal distance $d(\mathbf{u}, L_k)$ from a point $\mathbf{u}$ to the line L_k is given by

$$d(\mathbf{u}, L_k) = \frac{|A_k u_x + B_k u_y + C_k|}{\sqrt{A_k^2 + B_k^2}}, \tag{32}$$

and the closest line(s) are thus easily identified. Next, the closest point on the each parameterized line is also easily calculated. This closest point will in general not coincide with an input sample location, but as the input samples are almost evenly distributed on the pushbroom lines, the indexes of the input points that are closest to $\mathbf{u}$ can be predicted.

This trick makes it possible to quickly home in on a small set of candidate neighbor points. Thus, instead of calculating the distance to *all* input points to find the neighbors to $\mathbf{u}$, we need only calculate the distance to a handful of points. Specifically, if we have N_{lines} each consisting of $N_{samples}$ in a pushbroom data set, to interpolate one point, the straightforward naive implementation requires $\mathcal{O}(N_{lines}N_{samples})$ distance computations whereas the method above only requires $\mathcal{O}(N_{lines})$ distance computations. Since $N_{samples}$ typically is larger than 1000 samples, the method above increases the computational efficiency by three orders of magnitude.

It is stressed that once the candidate neighbor points have been found, the actual distances to these points are calculated for the interpolation using their georeferenced coordinates; the line parameterization is just used as an efficient way of identifying a small subset of neighbor candidate points. The parameters $\{A_k, B_k, C_k\}$ are calculated for all lines only once

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

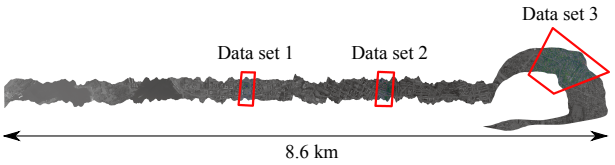

Fig. 7. Three data sets for evaluation are taken from different parts of a long pushbroom swath over the city of Oslo. Each data set consists of the same number of pushbroom lines.

TABLE I
SIZES OF THE INPUT PUSHBROOM DATA AND THE UNIFORM OUTPUT GRID.

	Input size	Output size
Data set 1	600×1600	986×1338
Data set 2	600×1600	1034×1467
Data set 3	600×1600	3181×2098

in a pre-computation step, and the computational effort for this is insignificant. It should be stressed that the above procedure works also when the samples on a pushbroom line deviate slightly from a straight line, e.g., due to lens distortion and small effects caused by the topography in the scene. It will also work if the pushbroom lines cross each other or if the ground prints of lines appear in reversed order relative to the flight direction due to strong motions caused by, e.g., banking or turbulence.

VI. DATA

Data used for evaluation in this work was acquired over the city of Oslo using the hyperspectral HySpex VNIR-1600 pushbroom sensor that is part of the FFI demonstrator system described in [24]. Each line in the image consists of 1600 pixels and 160 spectral bands are measured in the visual to near-infrared wavelengths. Each sensor element has an instantaneous field-of-view of approximately 0.18 mrad across- and 0.36 mrad along-track, yielding an average ground sample distance for this particular data set of about 35 cm. The data was georeferenced using auxiliary INS and DSM data [2]. Three subsets from the 8.6 km long swath were selected for the evaluations experiments, see Fig. 7. All three data sets have the same input size, i.e., 600 pushbroom lines times the 1600 samples of each line. The spatial resolution of the rectified output images $(\Delta x, \Delta y)$ was set to 30×30 cm^2 in all experiments. Table I summarizes the sizes of the input data and the uniform output grids for the three data sets. As the aircraft had different altitudes and velocities in different parts of the swath, the density of the irregular input grid is varying, and data set 3 is more sparse than the others.

VII. RESULTS

The interpolation methods described in Section IV were implemented in C/C++, except for the NAT method for which the *TriScatteredInterp*-function in Matlab was used. Below, pushbroom image rectification using the different interpolation methods is compared in terms of accuracy, parameter stability and computational speed.

Method	Parameter	DS1	DS2	DS3	1% rise bracket
NN	σ_i	0.90	1.20	0.96	$[0.84, 0.98]$
	σ_t	0.65	0.00	0.35	$[0.50, 0.76]$
	σ_n	0.00	0.00	0.00	$[0.00, 0.20]$
IDW	σ_i	1.00*	1.00*	1.00*	-
	σ_t	1.00*	1.00*	1.00*	-
	σ_n	3.61	3.02	0.00	$[2.26, 5.09]$
Kriging	σ_i	0.65	0.72	1.88	$[0.48, 0.76]$
	σ_t	0.12	0.00	0.00	$[0.00, 0.36]$
	σ_n	0.46	0.30	0.00	$[0.30, 0.58]$
Splatting	σ_i	1.12	1.18	2.70	$[0.95, 1.27]$
	σ_t	0.00	0.00	0.00	$[0.00, 0.073]$
	σ_n	0.50	0.33	0.00	$[0.31, 0.65]$

TABLE II
OPTIMIZED VALUES FOR INTERPOLATION METHODS ON THE DIFFERENT DATASETS. 1% RISE BRACKETS ARE FOUND ON DATASET 1. * MEANS THAT THE VALUE WAS SET MANUALLY.

A. Parameter tuning

The parameters obtained with the cross-validation procedure outlined in Section IV-D are presented in Table II. The IDW method is not sensitive to the isotropic part of the metric tensor, as discussed in Section IV-C. For this reason, the optimization is not stable for this method and only σ_n was optimized, keeping σ_i and σ_t manually set to 1.0.

As discussed in section IV-D, there is an inherent ambiguity in the isotropic part of the covariance. In Table II, we can see that this has caused most of the isotropy to end up in σ_i. While this has no negative effect on the resultant interpolation quality, it is still unfortunate, as the found parameters are more difficult to interpret. Still, the larger values of σ_i suggest that the anisotropic surface component is more important to model than the anisotropic pixel footprint function. However, σ_n and σ_t also get significant values, indicating that the model of an anisotropic footprint also contributes to the interpolation accuracy. In most cases we have $\sigma_n > \sigma_t$, which means that the pixel footprint is stretched in the movement direction of the aircraft. Dataset #3 is of a different nature than datasets #1 and #2, with the same number of samples covering a much larger area, see Fig. 7. This makes the optimizer prefer larger covariances for dataset #3, and as can be seen in Table II the larger covariance is produced by a large σ_i.

The cost function $\varepsilon(\mathbf{p})$ for the Splatting method is plotted in Fig. 8. To investigate the stability of the parameters, we characterize the local shape of the cost function at the minimum $\varepsilon(\mathbf{p}^*)$ by checking how far we have to move in each direction for it to rise by 1%. These rise points can be found using the following approximative procedure: Very close to the optimum, the cost function (30) as a function of one of the parameters, may be approximated with a second order Taylor expansion. Thus, we first move a small delta h up and down from the optimum $\mathbf{p}^* = (\sigma_i^*, \sigma_t^*, \sigma_n^*)$, along each parameter dimension. We then fit quadratic polynomials to the function values and arguments along each axis. Using the fitted polynomials, we then find the points where $\varepsilon(\mathbf{p})$ has risen 1% above the minimum. These intervals are reported in the rightmost column in Table II. For example, the parameters found on dataset #2 agree well with the ranges found on dataset #1. Thus, we conclude that the found parameters

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

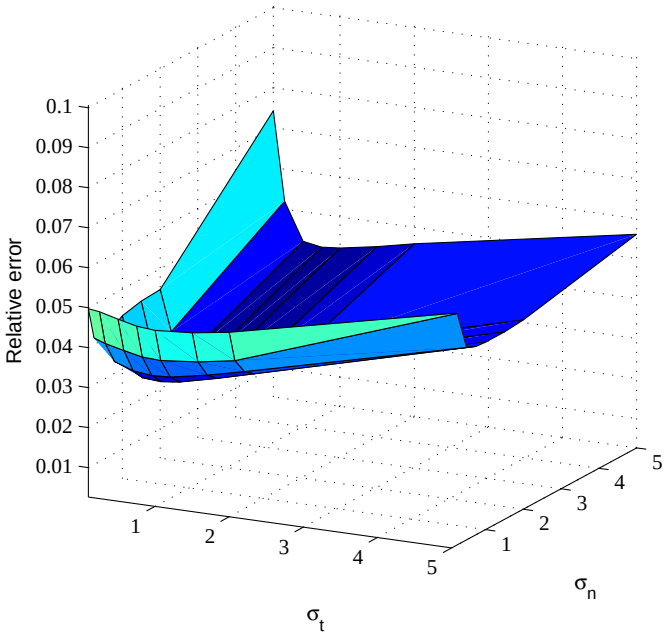

Fig. 8. Example of a grid search. The relative interpolation error is plotted as a function of the σ_n and σ_t parameters (with $\sigma_i = 0$) in the Splatting method.

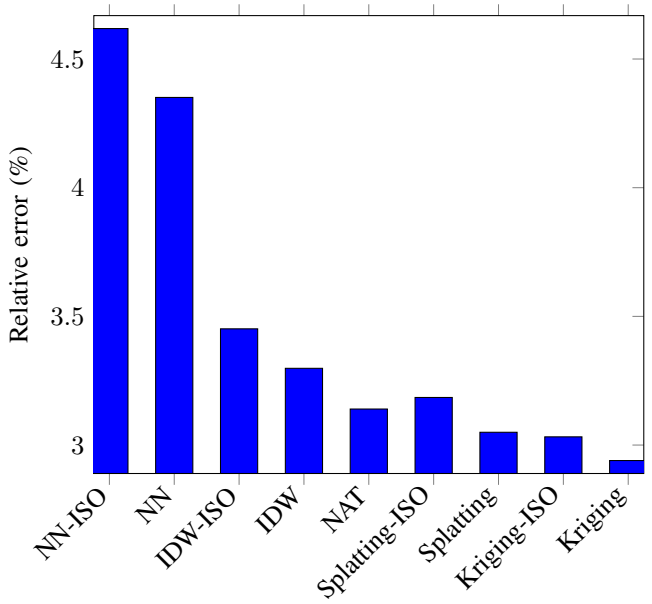

Fig. 9. Accuracy comparison of the interpolation methods with tuned optimal parameters. The relative interpolation error has been averaged across all three datasets. The suffix ISO indicates isotropic interpolation and no suffix means anisotropic interpolation. The NN method has the lowest accuracy and the Kriging method the highest accuracy.

generalize well between these two sets.

B. Interpolation accuracy comparison

The average relative interpolation errors across all three datasets for each interpolation method with optimal parameters are plotted in Fig. 9. It is clearly seen that the NN method has inferior interpolation accuracy. The remaining methods all perform similarly, with only a small accuracy advantage of

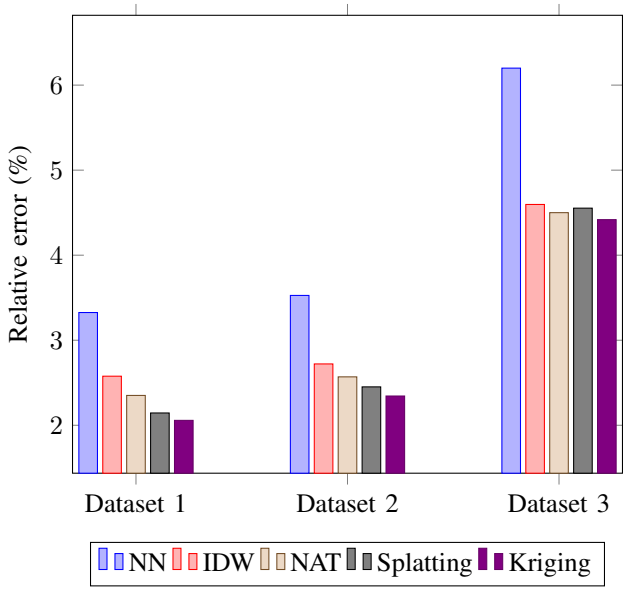

Fig. 10. Accuracy results for each of the three datasets using the different tuned anisotropic interpolation methods.

the Kriging method. It can also be seen that the anisotropic interpolation in all cases perform better than the isotropic interpolation. Fig. 10 shows the interpolation error per dataset. Again the NN method has the highest error while the other methods perform similarly. Data set 3 has somewhat higher interpolation errors due to the more irregular flight path and larger ground sampling distances, cf. Fig. 7.

In Fig. 11, three bands in the red, green and blue wavelengths have been rectified to produce RGB-images for a visual comparison between the methods. The NN method exhibits blocking artifacts that are typical for this method. For the remaining methods, a close examination reveals slight differences in the degree of smoothness, but overall they are similar. This is consistent with the interpolation accuracy results above. In some of the rectified images, the pushbroom line sampling pattern is visible, see for example the third and fourth rows from the top in Fig. 11. The line patterns are most prominent in the NN, Splatting and IDW methods, but for Kriging the effect is almost completely removed. To further visualize differences between the interpolation methods, all spectral bands in the hyperspectral datasets were rectified. In Fig. 12, the spectra in selected pixels in the evaluation set are plotted together with the actual measured spectra. Again the main difference is between the NN and the other methods.

C. Surface structure dependency

In order to further analyze the benefits of the anisotropic interpolation model, we have also separated the evaluation pixels in subsets with large and small amounts of surface structure respectively. All ground truth pixels are sorted according to their gradient strength as reflected in the trace of the structure tensor $\mathrm{tr}(\mathbf{T})$ computed according to (9), and the top 10% are denoted the ANISO subset. Similarly, the lowest 10% are denoted the ISO subset. Errors for these two subsets for dataset 2 are reported in Table III for the IDW and

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

(a) NN (b) Splatting (c) NAT (d) IDW (e) Kriging

Fig. 11. Details of the reconstruction results. Top two rows are from dataset 1, middle two rows are from dataset 2 and bottom two rows are from dataset 3. The images are best viewed in the electronic version of this article.

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

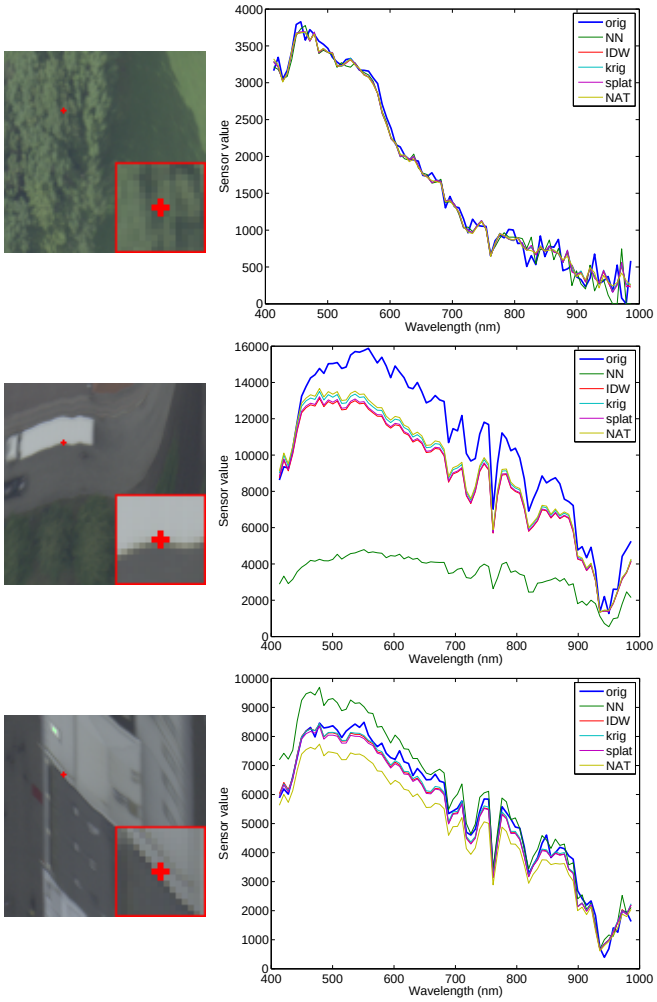

Fig. 12. Interpolation for a few pixels in all spectral bands of the hyperspectral image. Left: the pixel used for interpolation. Right: Spectral plots for the different methods. "orig" is the actually measured spectrum and the other curves are different interpolation results.

IDW	ISO	ANISO
10% ISO	1.29%	1.28%
10% ANISO	7.13%	6.18%

Splatting	ISO	ANISO
10% ISO	1.29%	1.27%
10% ANISO	6.13%	5.11%

TABLE III
COMPARISON OF ERRORS IN REGIONS WITH ISOTROPIC AND ANISOTROPIC GROUND STRUCTURE FOR THE IDW AND SPLATTING METHODS ON DATASET 2.

Splatting methods respectively. The largest difference between the methods using anisotropic or isotropic surface structure model is for the ANISO subset. Similar results are obtained for the other interpolation methods.

D. Speed comparison

Depending on the application, computational speed may be an important factor. The interpolation methods have quite different computational complexities and we have compared the speed between the methods on the different datasets, see Table IV. The numbers in this table were obtained with serial

TABLE IV
METHOD TIMINGS IN SECONDS (SERIAL EXECUTION)

	Dataset 1	Dataset 2	Dataset 3
Splatting	0.4	0.7	4.0
NN	5.4	6.3	26.3
IDW	5.5	6.5	27.5
NAT	21.4	25.8	141.4
Kriging	39.3	43.7	208.3

implementations of the algorithms. The forward interpolation scheme represented by the Splatting method is about an order of magnitude faster than the other methods. Among the inverse interpolation schemes, NN and IDW are the fastest. The computational effort in these methods is mainly spent on finding the closest input samples for the interpolation according to the procedure outlined in Section V. The Kriging interpolation requires additional computations, e.g., the matrix inverse in (28), and it is the slowest method. The NAT method requires a triangulation pre-processing step which is computationally demanding.

In the inverse interpolation schemes the points in the output grid are calculated independently of each other, cf. (23). The implementation of these schemes, in particular NN, IDW and Kriging, is for this reason trivial to parallelize. For example, on a CPU with 4 cores, a speedup of a factor 3.5 was obtained with a parallel implementation compared to the serial implementation. Thus, these methods benefit extensively from parallel processing architectures such as multicore CPUs or GPUs, and may outperform the Splatting method if such resources are available.

In summary, the Splatting method performs the interpolation with the lowest amount of computations, but with a proper implementation, all methods should be able to perform real-time, for example to rectify images on-line in a carrier aircraft.

VIII. DISCUSSION

The image rectification problem addressed in this work is required for visualization of pushbroom data and for fusing results derived from such data with other imaging or geographic information. A key contribution is the modeling of the spatial dependence structure of pushbroom data in terms of the spatial covariance function. The covariance function involves two generally anisotropic and spatially non-stationary components: one that depends on the special properties of the pushbroom data acquisition and one that depends on the imaged surface structure itself. Based on this dependence model, five different interpolation methods for scattered spatial data are compared to interpolate the pushbroom samples at positions in a uniform grid.

In terms of interpolation results, the Nearest Neighbor method is inferior to the other interpolation methods, as can be expected due to its simplicity. The Kriging method consistently performs the best and has fewer visual striping artifacts, but the remaining methods are also viable from a practical view. Furthermore, the anisotropic interpolation schemes consistently yield better results than their isotropic

Dette er en postprint-versjon / This is a postprint version.
DOI til publisert versjon / DOI to published version: 10.1109/TIP.2014.2316377

counterparts. From a practical perspective, it is interesting to look at the computational performance of the different algorithms. In a straightforward implementation on a serial processing unit, the Splatting method has a clear advantage. With the trick to utilize the semi-structured sampling pattern of the pushbroom sensor, inverse interpolation schemes become feasible from a computational view, albeit still not as fast as the Splatting method. The inverse interpolation schemes have the attractive property of guaranteeing that there are no holes with undefined values in the rectified image. Moreover, the inverse interpolation schemes benefit trivially from parallel computational architectures, as each output pixel can be computed independently. With such resources available, inverse schemes could overtake the Splatting method.

A limitation of the current work, and a subject of future work is how to make the ground structure covariance estimation fully automatic. In this work we employ the approach suggested in [22], which uses neighborhoods of constant size, and thus implicitly assumes a constant sample density. In e.g. our dataset 3, where the samples are considerably more sparse, and irregular than in the other two, we had to adjust the neighborhood size manually, i.e. the σ parameter in (9).

A second limitation is that the footprint component of the covariance function is assumed to have constant magnitude (we only change its orientation), the reason being that more investigations are required of how to determine the parameters in the theoretical covariance model in (14) and to let it adapt from pixel to pixel. Through some approximations, e.g., a locally flat terrain around a pushbroom sample and a locally linear flight path, the pixel footprint function is modeled with an anisotropic Gaussian shape in this work. In recent work [25], the pixel footprint for each pushbroom sample was estimated using a Monte Carlo ray tracing method which takes the continuous flight path measured by the on-board INS system and a Digital Surface Model (DSM) into account. Thousands of rays are sent to build up a non-parametric representation of the distribution of ground points that contribute to the pushbroom sample under consideration. Though computationally very demanding, the resulting distribution could be used for interpolation using a forward interpolation Splatting scheme.

IX. Conclusions

Different scattered data interpolation methods for the rectification of pushbroom images have been compared. A model of the pushbroom image acquisition process reveals an inherently anisotropic spatial data dependence structure that should be taken into account in the interpolation, in addition to an anisotropic surface model. This is supported by the experimental results, where consistent gains in accuracy were observed when adding anisotropic modeling, compared to the isotropic case. The anisotropic interpolation models presented here strikes a good balance between efficiency and accuracy, as it results in interpolation methods that can run at interactive speeds. Further gains in accuracy may be obtained with more sophisticated models, but at the expense of more demanding computations. To conclude with a recommendation of method choice, the Inverse Distance Weighted method strikes a good balance between accuracy and practical usability. The Kriging method is superior in terms of quality at the expense of a higher computational complexity, and the Splatting method could be the method of choice if computational resources are scarce.

Acknowledgement

This research has been funded by the Swedish Armed Forces and the CENIIT organization at the Linköping Institute of technology.

References

[1] R. Gupta and R. Hartley, "Linear pushbroom cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 9, pp. 963–975, 1997.

[2] T. Opsahl, T. Haavardsholm, and I. Winjum, "Real-time georeferencing for an airborne hyperspectral imaging system," in *Proceedings of the SPIE Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVII*, 2011.

[3] J. Reguera-Salgado, M. Calvino-Cancela, and J. Martin-Herrero, "GPU geocorrection for airborne pushbroom imagers," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 11, pp. 4409–4419, 2012.

[4] R. J. Aspinall, W. A. Marcus, and J. W. Boardman, "Considerations in collecting, processing, and analysing high spatial resolution hyperspectral data for environmental investigations," *Journal of Geographical Systems*, vol. 4, no. 1, pp. 15–29, 2002.

[5] R. Franke, "Scattered data interpolation: Tests of some methods," *Math. Comput.*, vol. 38, no. 157, pp. 181–200, 1982.

[6] S.-N. Lam, "Spatial interpolation methods: A review," *Amer. Cartogr.*, vol. 10, no. 2, pp. 129–149, 1983.

[7] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: A survey," *J. Electron. Imaging*, vol. 11, no. 2, 2002.

[8] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Verlag, London, 2011.

[9] L. Westover, "Interactive volume rendering," in *CH Volume Visualization Workshop*, 1989, pp. 9–16.

[10] P.-E. Forssén and E. Ringaby, "Rectifying rolling shutter video from hand-held devices," in *IEEE CVPR*, 2010.

[11] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proceedings of the ACM National Conference*, 1968.

[12] D. G. Krige, "A statistical approach to some mine valuations and allied problems at Witwatersrand," Master's thesis, University of Witwatersrand, South Africa, 1951.

[13] G. Matheron, "The theory of regionalized variables and its applications, chapter 3," *Les Cahiers du Centre de Morphologie Mathématique de Fontainebleu*, vol. 5, 1971.

[14] J. Braun and M. Sambridge, "A numerical method for solving partial differential equations on highly irregular evolving grids," *Nature*, vol. 376, no. 24, pp. 655–660, 1995.

[15] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, 1989.

[16] L. Piegl and W. Tiller, *The NURBS Book*. Springer Verlag, 1997.

[17] M. Demirhan, A. Ozpinar, and L. Ozdamar, "Performance evaluation of spatial interpolation methods in the presence of noise," *Int. J. Remote Sens.*, vol. 24, no. 6, pp. 1237–1258, 2003.

[18] A. A. Eldeiry and L. A. Garcia, "Using deterministic and geostatistical techniques to estimate soil salinity at the sub-basin scale and the field scale," in *Proceedings of the AGU Hydrology Days*, 2011.

[19] B. Jähne and H. Haussecker, *Computer Vision and Applications*. Academic Press, 2000.

[20] M. Andersson, "Controllable multi-dimensional filters and models in low-level computer vision," Linköping Studies in Science and Technology. Dissertations No. 282, Linköping University, 1992.

[21] J. Weickert, *Anisotropic Diffusion in Image Processing*. ECMI Series, Teubner-Verlag, 1998.

[22] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 349–366, February 2007.

[23] E. H. Isaaks and R. M. Srivastava, *An Introduction to Applied Geostatistics*. Oxford University Press, 1989.

[24] T. Skauli, T. Haavardsholm, I. Kåsen, G. Arisholm, A. Kavara, T. Olsvik-Opsahl, and A. Skaugen, "An airborne real-time hyperspectral target detection system," in *SPIE 7695, Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVI*, 2010.
[25] T. Opsahl and T. V. Haavardsholm, "Estimating the pixel footprint distribution for image fusion by ray tracing lines of sight in a Monte Carlo scheme," in *SPIE 8743, Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XIX*, 2013.