A Hierarchical Wavelet Decomposition for Continuous-Time SLAM

Sean Anderson[1], Frank Dellaert[2], and Timothy D. Barfoot[1]

***Abstract*— This paper proposes using hierarchical wavelets as a basis in parametric continuous-time batch estimation. The need for a continuous-time robot pose in the simultaneous localization and mapping (SLAM) problem has arisen as state-of-the-art batch SLAM algorithms attempt to handle more challenging hardware; specifically, the continuous-time framework is particularly beneficial when using high-rate sensors, multiple unsynchronized sensors, or scanning sensors, such as lidar and rolling-shutter cameras, during motion. Although the traditional discrete-time SLAM formulation can be adapted by using temporal pose interpolation, approaches using the continuous-time framework are able to generate smooth robot trajectories with less state variables. In this paper, we focus on the parametric approach using temporal basis functions to develop a finite-element representation of the continuous-time robot trajectory. While the majority of current implementations have utilized a uniformly spaced B-spline basis, we note that trajectory richness is often quite variable; in this paper, we show how a hierarchical system of wavelet basis functions can be used to increase the resolution of the solution only in the temporally local regions of the trajectory that require additional detail. We validate our approach by contrasting uniform B-splines and wavelets in a six-dimensional pose-graph SLAM experiment, using both simulated and real data.**

I. Introduction

Batch nonlinear optimization techniques for simultaneous localization and mapping (SLAM) are an important aspect of autonomous robotics and are a staple for generating accurate pose estimates over long distances. In particular, the use of stereo cameras for visual odometry remains a leading paradigm that encourages the formulation of a full nonlinear least-squares optimization problem, because it is both flexible and efficient [1]. However, due to the nature of the traditional discrete-time SLAM formulation, it is not well-suited to handle many challenging sensor outputs; specifically, the use of high-rate, motion-distorted, or unsynchronized sensors all require special treatment. For example, motion-distorted sensor outputs are often ignored by reducing platform velocity, increasing capture rate, or limiting the robot to a stop-and-go motion tactic, and extremely high-rate measurements are often marginalized to occur at a more convienent rate.

A recent strain of estimators focus on addressing the shortcoming of discrete-time batch SLAM to handle high-rate measurements by leveraging a continuous-time model to represent the pose of the robot. A motivating example for the robotics community is the desire to use rolling-shutter type cameras interchangeable with global-shutter type cameras in a bundle adjustment style estimator; rolling shutter cameras often use Complementary Metal-Oxide-Semiconductor (CMOS) technology, which is known to be far cheaper than global shutter cameras that employ a Charge-Coupled Device (CCD). Continuous-time batch estimation has been performed both parametrically, using a weighted sum of temporal basis functions [2], and non-parameterically, using a Gaussian-Process [3]. A benefit of the continuous-time approach is that it provides an elegant means to include asynchronous and high-rate measurements, such as those from an inertial measurement unit (IMU) or even individually timestamped features extracted from a motion-distorted image, into a typical batch nonlinear optimization framework.

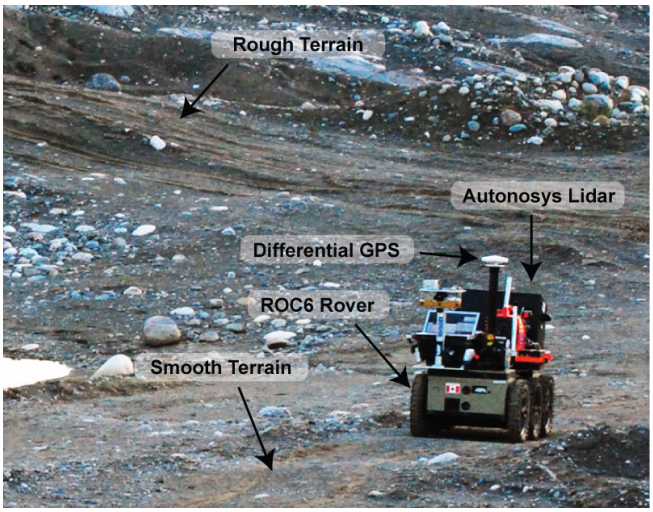

Fig. 1. This figure shows the ROC6 mobile rover, equipped with an Autonosys LVC0702 lidar and a Thales DG-16 Differential GPS unit, traversing the Ethier Sand and Gravel pit in Sudbury, Ontario, Canada. Data collected from this experiment is used to validate our approach in Section IV. Using hierarchical wavelets, our algorithm is able to automatically add detail to segments of the solution that experience rough terrain, while maintaining a smaller state size in smooth parts of the trajectory.

While several continuous-time, state estimation implementations have used uniformly spaced B-splines, in reality the richness of a trajectory can be quite variable. Consider a traversal where a ground robot is twisting and turning over natural, unstructured, three-dimensional terrain and then encounters a flat, open space where it travels in a straight line for several meters (see Figure 1); in order to sufficiently model this trajectory, it is expected that each segment will require a very different temporal density of basis functions.

We propose using a hierarchical system of wavelet basis functions to parameterize the continuous-time robot trajectory. Wavelets and the theory of multiresolution analysis are

[1]University of Toronto Institute for Aerospace Studies, 4925 Dufferin Street, Toronto, Ontario, Canada, sean.anderson@mail.utoronto.ca, tim.barfoot@utoronto.ca. [2]School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA, frank@cc.gatech.edu.

a parametric approach used in a variety of fields, ranging from signal analysis to computer graphics [4]. Using wavelet functions allows us to decompose our continuous-time trajectory into a coarse overall shape and a set of refining detail functions that can be adaptively added to the solution where they are required.

In Section II, we begin with a review of related literature. In Section III, we provide some background on multiresolution analysis techniques and formulate our adaptive wavelet parameterization for continuous-time state estimation. In Section IV, we set up an example state estimation problem and provide the results of our algorithm using both simulated and real data. Final comments and a discussion of future work are available in Section V.

II. Related Work

In its most fundamental form, continuous-time estimation can be performed by applying pose-interpolation techniques to the traditional, discrete-time SLAM formulation. Similarly to discrete-time SLAM, notable works using pose interpolation place *keyframes* at every frame acquisition time. Although it is possible to interpolate between non-sequential frames, the representational power of these formulations is somewhat limited. Using a continuously spinning lidar, Bosse and Zlot [5] present a technique for performing SLAM while scanning and moving; the technique is akin to the iterative closest point (ICP) algorithm and interpolates poses in order to compensate a sequence of temporally discretized lidar scans. Continuations of this work have been successful in mapping a large underground mine [6] and estimating the irregular motion of a 2D lidar attached to the end of a spring [7]. Dong and Barfoot [8] perform frame-to-frame motion-compensated visual odometry using sparse, appearance-based features extracted from the intensity data of a two-axis scanning lidar. With a high-rate, rolling-shutter camera, Hedborg et al. [9] demonstrate the application of interpolation to the full nonlinear bundle adjustment problem.

Modelling the trajectory of a robot as a continuous-time function is an active research area in the field of robotics. Furgale et al. [2] formally derive the general continuous-time SLAM problem and demonstrate the use of uniformly spaced temporal basis functions to model the state in a typical camera-IMU calibration problem. The works by Bibby and Reid [10] and Fleps et al. [11] also exhibit the use of uniformly spaced B-splines to parameterize the robot trajectory in a global frame. Anderson and Barfoot [12] also leverage this parametric approach to continuous-time estimation, but derive the relative formulation of the problem, similar to the discrete-time work of Sibley et al. [13].

In an effort to improve representational power, Tong et al. [3] use a Gaussian process to model the continuous-time trajectory of a robot. Although using a Gaussian process in essence provides an infinite resolution solution, we note that the often *weak* prior term, common to many of the continuous-time SLAM formulations including the one presented in this paper, plays a much more important role without a finite basis to help prevent overfitting.

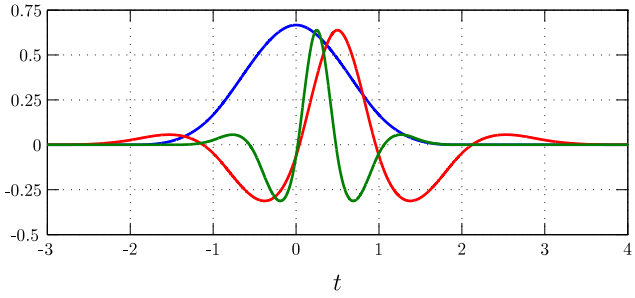

Fig. 2. This figure depicts the shape and temporal spacing of our chosen scaling and wavelet functions over a uniformly spaced integer knot sequence. The blue line depicts a cubic B-spline scaling function at level ℓ, spanning four time segments. The red and green lines depict our choice of wavelet function, at level ℓ and $\ell + 1$, spanning 7 and 3.5 time segments, respectively. Note that at each increase of resolution level, wavelets are simply defined over a knot sequence that uses half-size time intervals.

Recent work by Oth et al. [14] demonstrates the use of parametric, continuous-time batch estimation to calibrate the line delay of a rolling-shutter camera. Due to the highly nonlinear nature of this problem, an integral component of their technique is an adaptive scheme that chooses a non-uniform basis spacing to prevent overfitting the data. To the authors' knowledge, this is the only other work that demonstrates an adaptive scheme for refining the parametric representation of a continuous-time trajectory.

We draw our wavelet motivation from the variational modelling technique developed by Gortler and Cohen [15] in the field of computer graphics. However, we note a fundamental difference in the problems being solved. The goal of their variational modelling approach is to find a continuous curve that passes exactly through a few known constraint points, while satisfying a minimum energy cost function. In contrast, a typical robotics problem has many noisy measurements and in parametric, continuous-time estimation, we are concerned with the problem of overfitting. This leads our formulation to adopt a different metric for determining the addition and placement of high-resolution wavelets.

III. Methodology

In this section, we derive our adaptive, continuous-time estimation scheme using a system of hierarchical wavelet basis functions. Using a wavelet basis allows us to decompose our solution into a rough overall shape and a hierarchy of refining details. Furthermore, it allows detail to be adaptively added to local segments of the continuous-time trajectory that require it. To gain a deeper understanding of these concepts we recommend reading [4]; motivation for the specific approach we take to variational modelling can be found in [15]. The necessary background information required to develop a hierarchical wavelet basis for continuous-time state estimation is summarized here.

A. Hierarchical Wavelets

We begin by defining our 6D continuous-time trajectory,

$$\mathbf{x}(t) := \begin{bmatrix} \boldsymbol{\rho}(t) \\ \boldsymbol{\varphi}(t) \end{bmatrix} \tag{1}$$

where $\boldsymbol{\rho}(t)$ are the translational parameters, and $\boldsymbol{\varphi}(t)$ are the rotational parameters. Using a wavelet basis we define our trajectory as the sum of a coarse shape function, $\mathbf{s}(t)$, and detail function, $\mathbf{d}(t)$,

$$\mathbf{x}(t) = \mathbf{s}(t) + \mathbf{d}(t), \tag{2}$$

where each is in turn a weighted sum of temporal basis functions.

In many robotic applications, the richness of motion in a trajectory can be quite variable; while some trajectory segments may be sufficiently smooth, others may include heavy maneuvering that require additional detail in the continuous-time parameterization to be properly modelled. In order to refine our finite-element solution, we define a hierarchical detail function,

$$\mathbf{d}(t) := \mathbf{d}_0(t) + \mathbf{d}_1(t) + \ldots + \mathbf{d}_L(t) \tag{3}$$

where $\ell = 0 \ldots L$ denotes the resolution *level*, and with each increase in level, $\ell + 1$, basis functions are simply defined over a knot sequence that uses half-size time intervals.

In the context of parametric continuous-time state estimation, there are a few important properties that we want our basis to have. The first is that only a temporally local subset of the basis functions is non-zero at any sample time; this is often referred to as *local support*. This property is important in order to enable local batch optimizations, such as a *sliding window filter*. The second important property is that the basis functions have simple analytical derivatives and integrals, which can be important when developing error terms for some sensors (such as an IMU).

The basis of our shape function, $\mathbf{s}(t)$, is constructed using temporally translated copies of a chosen *scaling function*, denoted $\phi(t)$; in this paper, we choose to use cubic B-spline basis functions. The cubic B-spline is a popular choice because it has C^2 continuity, which is important when evaluating the state for acceleration. Furthermore, it is simple to include B-splines in our formulation using the matrix representations developed by [16].

The basis of the detail functions, $\mathbf{d}_\ell(t)$, are constructed using both scaled and temporally translated copies of a chosen *wavelet function*, denoted $\psi(t)$. In this paper, we adopt the wavelet basis recommended for cubic B-splines in [15], depicted in Figure 2.

Our hierarchical system uses a single shape function, $\mathbf{s}(t)$, at only the coarsest resolution level, $\ell = 0$. However, we define it more generally to assist in our discussion of *multiresolution analysis*, in Section III-C, where we show the construction method of the chosen wavelet basis function. Using matrix notation, we write a shape function as

$$\mathbf{s}_\ell(t) := \boldsymbol{\Phi}_\ell(t)\mathbf{c}_{\phi_\ell}, \tag{4}$$

$$\boldsymbol{\Phi}_\ell(t) := \begin{bmatrix}\phi_{\ell,0}(t) \ldots \phi_{\ell,v_\ell-1}(t)\end{bmatrix}, \tag{5}$$

$$\phi_{\ell,i}(t) := \phi(2^\ell t - u_i), \tag{6}$$

where the scaling basis, $\boldsymbol{\Phi}_\ell(t)$, spans the vector space, V_ℓ (of dimension v_ℓ), $\phi_{\ell,i}(t)$ is a 6×1 column vector, u_i is the translation of basis function i, and $\mathbf{c}_{\phi_\ell}$ is a $v_\ell \times 1$ column vector of weights that allow us to vary the output. Similarly, we write the detail functions as

$$\mathbf{d}_\ell(t) := \boldsymbol{\Psi}_\ell(t)\mathbf{c}_{\psi_\ell}, \tag{7}$$

$$\boldsymbol{\Psi}_\ell(t) := \begin{bmatrix}\psi_{\ell,0}(t) \ldots \psi_{\ell,w_\ell-1}(t)\end{bmatrix}, \tag{8}$$

$$\psi_{\ell,i}(t) := \psi(2^\ell t - u_i), \tag{9}$$

where the wavelet basis, $\boldsymbol{\Psi}_\ell(t)$, spans the vector space, W_ℓ (of dimension w_ℓ), $\psi_{\ell,i}(t)$ is a 6×1 column vector, u_i is the translation of basis function i, and $\mathbf{c}_{\psi_\ell}$ is a $w_\ell \times 1$ column vector of weights that allow us to vary the output. The full hierarchical framework can then be written as

$$\mathbf{x}(t) := \boldsymbol{\Omega}(t)\mathbf{c}, \tag{10}$$

$$\boldsymbol{\Omega}(t) := \begin{bmatrix}\boldsymbol{\Phi}_0(t) & \boldsymbol{\Psi}_0(t) & \boldsymbol{\Psi}_1(t) & \cdots & \boldsymbol{\Psi}_L(t)\end{bmatrix}, \tag{11}$$

$$\mathbf{c} := \begin{bmatrix}\mathbf{c}_{\phi_0}^T & \mathbf{c}_{\psi_0}^T & \mathbf{c}_{\psi_1}^T & \cdots & \mathbf{c}_{\psi_L}^T\end{bmatrix}^T, \tag{12}$$

where L is the finest resolution level used by the hierarchical parameterization.

In this paper, we perform only full batch optimizations and make use of the extreme case, where the coarsest resolution level contains only a single time segment. However, we note that it is trivial to perform local batch optimizations by including more time segments at the coarsest level; this would becomes important when implementing a *sliding window filter* for SLAM.

B. Adaptability

A key benefit of using a wavelet basis for the continuous-time parameterization of a robot trajectory is that it provides a means to adaptively add resolution only where it is needed in the solution. The contribution of a wavelet to the continuous-time trajectory depends on its resolution and the detail required in the time segments that it locally supports.

An important intuition is that as ℓ is increased, and we add finer and finer resolution wavelets to our parameterization, the wavelet coefficient weights, $\mathbf{c}_{\psi_\ell}$, should tend to zero as we capture the shape of the curve. Therefore, a hierarchical wavelet basis has the ability to exclude all wavelet basis functions with a negligable coefficient weight, while making only a small approximation to the continuous-time function.

The development of an adaptive algorithm, to decide if and where the basis requires refinement, plays a critical role in using the hierarchical wavelet system to its full potential. The variational modelling approach from computer graphics uses an *oracle* algorithm that simply looks at the magnitude of the coefficient weights. However, as noted previously, the variational modelling problem often only needs to satisfy a few noise-free constraints. Closely analyzing the weights of wavelet coefficients in a typical robotics problem will inevitably lead to overfitting our solution.

In order to safely predict where we need more detail, without overfitting, we adopt the metric proposed by Oth et al. [14] in their adaptive knot subdivision scheme. The proposed metric helps avoid overfitting by comparing the objective function cost, J, against its expected value, $E[J]$. In our hierarchical wavelet system, we begin by activating

only the coarsest level of basis functions, $L = 0$. For each time segment, s, of each active level, $\ell = 0 \dots L$, we then check if $J_{\ell,s} > E[J_{\ell,s}]$; if true, it implies the trajectory requires additional detail in that time segment, and we enable the wavelet basis functions that have local support over that temporal area. Iteratively, we then solve for the coefficients of active wavelets, increment our maximum active resolution level, $L = L+1$, and re-check if, and where, additional resolution is required. We note that the success of this scheme is dependent on having properly modelled the covariance of the measurements.

By tactically placing detail coefficients, we expect a reduced state size, which is computationally important to performing batch estimation problems. Additionally, the hierarchical approach we take to iteratively adding detail allows for a convienent initialization method. At each iteration, the previously solved state vector, using $\ell = 0 \dots L-1$, is still a valid solution to the expanded state vector, using $\ell = 0 \dots L$. The new components, $\mathbf{c}_{\psi_L}$, can simply be initialized to zero. Assuming the solution does not change drastically, initializing at the previously optimal solution allows the nonlinear batch estimator to converge in less iterations.

C. Multiresolution Analysis

In this subsection, we will define the relationship that exists between basis functions at different resolution levels, show how this relationship is used to construct our chosen wavelet basis function, and derive an interesting parallel that exists between our hierarchical parameterization and a traditional uniform B-spline parameterization. This parallel will be useful in quantifying the compression of our parameterization. We begin by noting that the scaling-function vector spaces, V_ℓ, are nested:

$$V_0 \subset V_1 \subset \ldots \subset V_\ell \subset \ldots \tag{13}$$

Since the basis of $V_{\ell-1}$ is a subspace of the basis of V_ℓ, we know that a double-width scaling function, belonging to the level $\ell - 1$, can be expressed as a linear combination of single-width scaling functions, belonging to the level ℓ. In matrix notation, this is written as,

$$\mathbf{\Phi}_{\ell-1}(t) = \mathbf{\Phi}_\ell(t)\mathbf{P}_\ell, \tag{14}$$

where $\mathbf{P}_\ell$ is a tall, band matrix of dimension $v_\ell \times v_{\ell-1}$ and each column is a translated copy of the vector $\mathbf{p}$, which is independent of level; the well-known weighting sequence for cubic B-spline basis functions is

$$\mathbf{p} = \begin{bmatrix} \frac{1}{8} & \frac{1}{2} & \frac{3}{4} & \frac{1}{2} & \frac{1}{8} \end{bmatrix}^T. \tag{15}$$

The effect of projecting a curve from a basis in V_ℓ to a double-width basis in $V_{\ell-1}$ is that the resulting curve is smoothed. The goal in choosing a wavelet basis is to be able to capture the detail that is lost between the adjacent scaling-function vector spaces; that is, we wish to choose a wavelet basis such that $W_{\ell-1}$ is the complement of $V_{\ell-1}$ in the space V_ℓ. Although there is some freedom in choosing the *mother wavelet*, $\psi(t)$, we must be able to express any function in V_ℓ as a combination of a function in $V_{\ell-1}$ and a function in $W_{\ell-1}$. Note that the wavelet space, $W_{\ell-1}$, is also a subspace of V_ℓ and therefore a double-width wavelet function, belonging to the level $\ell - 1$, can also be expressed as a linear combination of single-width *scaling functions*, belonging to level ℓ. In matrix notation, this is written as,

$$\mathbf{\Psi}_{\ell-1}(t) = \mathbf{\Phi}_\ell(t)\mathbf{Q}_\ell \tag{16}$$

where $\mathbf{Q}_\ell$ is a tall, band matrix of dimension $v_\ell \times w_{\ell-1}$ and each column is a translated copy of the vector $\mathbf{q}$. By using these hierarchical relationships, all that is left in order to construct a wavelet function is choosing a valid weighting vector, $\mathbf{q}$. In this paper, we adopt the sequence for cubic B-splines suggested by [17]:

$$\mathbf{q} = \frac{1}{256}\begin{bmatrix} 5 & 20 & 1 & -96 & -70 & 280 & -70 \\ & & & & & -96 & 1 & 20 & 5 \end{bmatrix}^T, \tag{17}$$

which results in the wavelet basis function shape illustrated in Figure 2.

Using the invertible *isomorphic* relationship, $\mathbf{W}_\ell$, note that a finite-element solution can be expressed equivalently in different vector spaces:

$$\begin{bmatrix} \mathbf{c}_{\phi_{\ell-1}} \\ \mathbf{c}_{\psi_{\ell-1}} \end{bmatrix} = \mathbf{W}_\ell \mathbf{c}_{\phi_\ell}, \quad \mathbf{W}_\ell := \begin{bmatrix} \mathbf{P}_\ell & \mathbf{Q}_\ell \end{bmatrix}^{-1}. \tag{18}$$

By applying this relationship recursively,

$$\mathbf{W}_\Omega := \begin{bmatrix} \mathbf{W}_0 & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \mathbf{W}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \cdots \begin{bmatrix} \mathbf{W}_L & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{W}_{L+1}, \tag{19}$$

we note a parallel between the hierarchical wavelet parameterization and the more traditional uniform cubic B-spline parameterization of continuous-time SLAM,

$$\mathbf{x}(t) = \mathbf{\Omega}(t)\mathbf{c} \equiv \mathbf{\Phi}_{L+1}(t)\mathbf{c}_{\phi_{L+1}}, \tag{20}$$

where $\mathbf{\Omega}(t)\mathbf{W}_\Omega \equiv \mathbf{\Phi}_{L+1}(t)$ and ${\mathbf{W}_\Omega}^{-1}\mathbf{c} \equiv \mathbf{c}_{\phi_{L+1}}$.

Using this parallel, we will later validate that our adaptive algorithm does not smooth any important details that could be captured in the *available* vector space, V_{L+1}, by verifying that our wavelet solution has similar, or less, error than a uniform B-spline solution using the basis $\mathbf{\Phi}_{L+1}(t)$.

IV. Experimental Validation

In this section, we validate our proposed hierarchical trajectory parameterization by applying it to a 6D pose-graph SLAM problem. We begin by formulating the problem in continuous time and then test our approach using both simulated and real data. We note that previous work [2][3][12] has shown the formulation of continuous-time bundle-adjustment-type problems and the associated state size reduction benefits over the discrete-time formulation; although the hierarchical formulation presented in this paper would also benefit bundle-adjustment-type problems, we simplify our problem formulation and focus on the contribution by only considering relative pose change measurements.

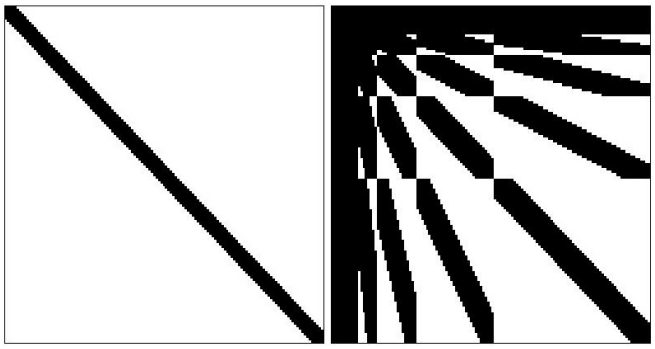

Fig. 3. In this figure, we contrast the sparsity patterns of the matrix $(\mathbf{A}_m + \mathbf{A}_u)$ using uniform B-splines (shown on the left) and using a hierarchical wavelet scheme (shown on the right). Each basis spans an identical vector space and the bases are correlated to the state vectors $\mathbf{c}_{\phi_{L+1}}$ and $\mathbf{c}$, respectively. Although hierarchical systems exhibit a denser sparsity pattern, we are able to maintain a relatively sparse block-banded structure by carefully ordering our state vector. At a high level, it is important that the state, $\mathbf{c}$, be sorted by level, as in (12); this also makes it trivial to add higher level wavelet coefficients on to an existing solution. The banded structure is obtained when the chosen wavelet basis has local support and coefficients of the sub-states, $\mathbf{c}_{\phi_0}, \mathbf{c}_{\psi_0}, \ldots, \mathbf{c}_{\psi_L}$, are sorted temporally. The hierarchical system shown here uses a single time segment at the coarsest level, 0, and has a maximum level of $L = 6$. In this extreme case, the entries corresponding to scaling functions and wavelets in the levels $\ell = 0 \ldots 2$ are fully dense because they have local support over the entire trajectory; using more time segments at the coarsest level will result in a sparser upper and left hand side.

A. Continuous-Time Pose-Graph SLAM

We begin by formulating the pose-graph SLAM problem and provide an overview of the typical continuous-time SLAM setup. A full derivation of the continuous-time, state-estimation framework can be found in [2].

In pose-graph SLAM we are given a set of relative pose change measurements, $\tilde{\mathbf{T}}_m$, between times $t_{m,1}$ and $t_{m,2}$, where each measurement, $\tilde{\mathbf{T}}_m$, is a 4×4 transformation matrix belonging to $SE(3)$, and has an associated 6×6 uncertainty matrix, $\mathbf{U}_m$. Our goal is to minimize a cost function by finding the optimal solution of the continuous-time robot trajectory, $\mathbf{x}(t)$. Using our hierarchical parameterization, we note that the solution of $\mathbf{x}(t)$ can be varied using only the finite-element vector $\mathbf{c}$.

We begin by defining our measurement error term, $\mathbf{e}_m$, to be the difference between our measured and estimated relative transform,

$$\mathbf{e}_m := \ln(\tilde{\mathbf{T}}_m \mathbf{T}(t_{m,1}) \mathbf{T}(t_{m,2})^{-1})^\vee = \ln(\tilde{\mathbf{T}}_m \mathbf{T}_m^{-1})^\vee, \quad (21)$$

where $\mathbf{T}(t)$ is a transformation matrix constructed from our state vector, $\mathbf{c}$, using the exponential map,

$$\mathbf{T}(t) := \exp(\mathbf{x}(t)^\wedge) = \exp((\boldsymbol{\Omega}(t)\mathbf{c})^\wedge), \quad (22)$$

$$\mathbf{x}^\wedge = \begin{bmatrix}\boldsymbol{\rho} \\ \boldsymbol{\varphi}\end{bmatrix}^\wedge := \begin{bmatrix}\boldsymbol{\varphi}^\wedge & \boldsymbol{\rho} \\ \mathbf{0}^T & 0\end{bmatrix}, \quad (23)$$

$$\boldsymbol{\varphi}^\wedge = \begin{bmatrix}\varphi_1 \\ \varphi_2 \\ \varphi_3\end{bmatrix}^\wedge := \begin{bmatrix}0 & -\varphi_3 & \varphi_2 \\ \varphi_3 & 0 & -\varphi_1 \\ -\varphi_2 & \varphi_1 & 0\end{bmatrix}, \quad (24)$$

and $\vee$ is the inverse operation of $\wedge$. The exponential map can be computed in closed form [18] using

$$\mathbf{T} := \exp(\mathbf{x}^\wedge) = \begin{bmatrix}\mathbf{C} & \mathbf{J}\boldsymbol{\rho} \\ \mathbf{0}^T & 1\end{bmatrix}, \quad (25)$$

where the rotation matrix $\mathbf{C} \in SO(3)$ is

$$\mathbf{C} := \exp(\boldsymbol{\varphi}^\wedge) = \cos\varphi \mathbf{1} + (1 - \cos\varphi)\mathbf{a}\mathbf{a}^T + \sin\varphi \mathbf{a}^\wedge, \quad (26)$$

the left *Jacobian* of $SO(3)$ is

$$\mathbf{J} := \frac{\sin\varphi}{\varphi}\mathbf{1} + (1 - \frac{\sin\varphi}{\varphi})\mathbf{a}\mathbf{a}^T + \frac{1 - \cos\varphi}{\varphi}\mathbf{a}^\wedge, \quad (27)$$

and our rotational parameters, $\boldsymbol{\varphi}$, use an axis-angle parameterization, $\boldsymbol{\varphi} := \varphi\mathbf{a}$, such that φ is the angle and $\mathbf{a}$ is a unit-length axis vector.

In continuous-time state estimation, we also typically associate a cost with our motion model; this acts as a prior and helps to smooth the solution. A typical choice is to model acceleration as a zero-mean, white-noise Gaussian process,

$$\ddot{\mathbf{x}}(t) \sim \mathcal{GP}(\mathbf{0}, \mathbf{Q}\delta(t - t')), \quad (28)$$

where the covariance function is $\mathbf{Q}\delta(t-t')$ and $\delta(\cdot)$ is Dirac's delta function. The error term is simply,

$$\mathbf{e}_u(t) := \ddot{\mathbf{x}}(t). \quad (29)$$

The objective function that we wish to minimize is

$$J(\mathbf{x}(t)) := J_m + J_u \quad (30)$$

$$J_m := \frac{1}{2}\sum_m \mathbf{e}_m^T \mathbf{U}_m^{-1} \mathbf{e}_m \quad (31)$$

$$J_u := \frac{1}{2}\int_{t=0}^{T} \mathbf{e}_u(\tau)^T \mathbf{Q}^{-1} \mathbf{e}_u(\tau) d\tau. \quad (32)$$

The full derivation, and details concerning (32), can be found in [2]. In order to minimize our objective function, J, with respect to the state vector, $\mathbf{c}$, we take the Gauss-Newton approach; we first linearize the error terms with respect to a perturbation in the state, $\delta\mathbf{c}$, initialize the nominal solution, $\bar{\mathbf{c}}$, and iteratively solve for the optimal update step, $\delta\mathbf{c}^*$, by solving $\frac{\partial^T J}{\partial \delta \mathbf{c}} = \mathbf{0}$ for $\delta\mathbf{c}$.

From the basis function definition, $\mathbf{x}(t) = \boldsymbol{\Omega}(t)\mathbf{c}$, it follows that $\bar{\mathbf{x}}(t) + \delta\mathbf{x}(t) = \boldsymbol{\Omega}(t)(\bar{\mathbf{c}} + \delta\mathbf{c})$ and therefore the nominal and perturbed solutions are simply $\bar{\mathbf{x}}(t) = \boldsymbol{\Omega}(t)\bar{\mathbf{c}}$ and $\delta\mathbf{x}(t) = \boldsymbol{\Omega}(t)\delta\mathbf{c}$. At each iteration, the nominal error is evaluated as

$$\bar{\mathbf{e}}_m = \ln(\tilde{\mathbf{T}}_m \exp(\bar{\mathbf{x}}(t_{m,1})^\wedge) \exp(-\bar{\mathbf{x}}(t_{m,2})^\wedge))^\vee. \quad (33)$$

Linearizing our measurement term, we find that

$$\mathbf{e}_m(\bar{\mathbf{c}} + \delta\mathbf{c}) \approx \bar{\mathbf{e}}_m - \mathbf{H}_m \delta\mathbf{c}, \quad (34)$$

where,

$$\mathbf{H}_m = \left[\boldsymbol{\mathcal{J}}(t_{m,2})\boldsymbol{\Omega}(t_{m,2}) - \mathrm{Ad}(\mathbf{T}_m)\boldsymbol{\mathcal{J}}(t_{m,1})\boldsymbol{\Omega}(t_{m,1})\right], \quad (35)$$

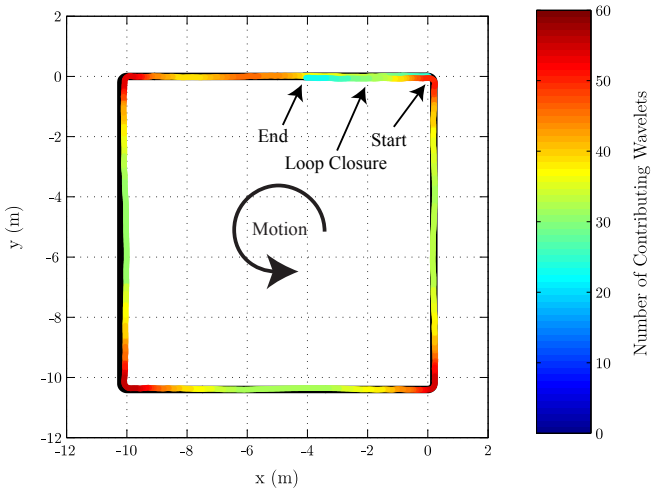

Fig. 4. This figure shows the estimated (coloured) and ground truth (black) trajectory used for the simulation results. The estimated trajectory is coloured based on the number of actively contributing wavelet coefficients at each sample time. The wavelet density is distinguishably higher near the corners of the solution, and relaxes in the straight segments.

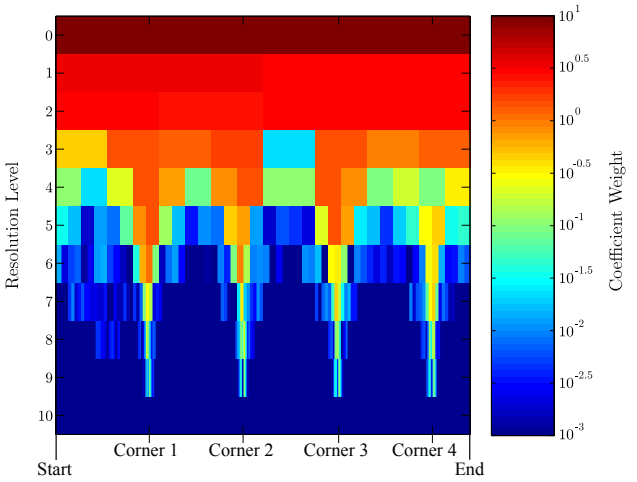

Fig. 5. This figure shows the wavelet coefficient magnitudes, $\mathbf{c}_{\psi_\ell}$, for the continuous-time function $\varphi_3(t)$. Each row of the plot corresponds to a different resolution level, ℓ, and contains 2^ℓ wavelet coefficients; non-active wavelet coefficients are set to 10^{-3} for colouring purposes. The distinguishable four spikes in wavelet depth correspond to the four corners traversed in the simulation trajectory.

$\mathrm{Ad}(\cdot)$ is the *adjoint* of $SE(3)$, and $\mathcal{J}$ is referred to as the *Jacobian* of $SE(3)$ (discussed further in [18]). Setting $\frac{\partial^T J}{\partial \delta \mathbf{c}} = \mathbf{0}$, we find the optimal state update equation:

$$(\mathbf{A}_m + \mathbf{A}_u)\delta\mathbf{c}^* = \mathbf{b}_m + \mathbf{b}_u. \tag{36}$$

Therefore, with respect to J_m, $\frac{\partial^T J_m}{\partial \delta \mathbf{c}} = \mathbf{A}_m \delta\mathbf{c} - \mathbf{b}_m$, where,

$$\mathbf{A}_m := \sum_m \mathbf{H}_m^T \mathbf{U}_m^{-1} \mathbf{H}_m, \tag{37}$$

$$\mathbf{b}_m := \sum_m \mathbf{H}_m^T \mathbf{U}_m^{-1} \bar{\mathbf{e}}_m. \tag{38}$$

The derivation of $\mathbf{A}_u$ and $\mathbf{b}_u$ is found in [2]. Following the Gauss-Newton approach, we solve for $\delta\mathbf{c}^*$ and update the nominal solution, $\bar{\mathbf{c}} \leftarrow \bar{\mathbf{c}} + \delta\mathbf{c}^*$; the whole approach is iterated to convergence. The sparsity pattern of $(\mathbf{A}_m + \mathbf{A}_u)$, created by using a hierarchical system of wavelet basis functions is shown in Figure 3. With respect to the adaptive algorithm, this optimization scheme is run at every iteration, $L = L+1$, before evaluating J and adding higher resolution wavelets where they are required.

B. Simulation Results

In order to illustrate the behaviour of the adaptive wavelet algorithm in a controlled environment, we begin with a simulation example. The simulated path is a square with sharply rounded corners to emphasize the need for temporally local detail coefficients, seen in Figure 4. The simulated robot moved at a constant linear velocity for 120 seconds and 6D relative pose estimates were generated at a rate of 10Hz (corrupted with zero-mean Gaussian noise); a loop closure observation was also provided near the end of the run. Intuitively, the expected result is that the adaptive algorithm will add very high-level resolution wavelet functions near the corners of the trajectory and add very low-level resolution wavelets on the straight segments.

In order to verify this intuition, we generated both geometric and weight-space plots that illustratively show the density of active wavelets being used in our adaptive solution. First, in Figure 4 we plot the estimated trajectory against the ground truth and colour it based on the number of actively contributing wavelet coefficients at each sample time. As expected, we see that near the corners there is a very high number of contributing wavelets (roughly 60 wavelets) and furthermore that the wavelet density gradually reduces as we approach the midpoint of a straight segment (roughly 25-30 wavelets). Second, we plot the wavelet coefficient magnitudes, $\mathbf{c}_{\psi_\ell}$, for the continuous-time function $\varphi_3(t)$, seen in Figure 5. Specifically, we are interested in $\varphi_3(t)$ because it roughly corresponds to the rotation of the robot around the z-axis, which is the state component requiring the sharp details in this simulation. Each row of the plot corresponds to a different resolution level, ℓ, and contains 2^ℓ temporally sorted wavelet coefficients. From this plot, the four corners in the simulated trajectory are easily distinguished by the deep level of high-magnitude coefficients. Furthermore, we see a nice gradient drop-off of wavelet magnitude on the sides of each peak.

In order to quantitatively discern if we are gaining any benefit by using a system of hierarchical wavelets, we compared the solution provided by the adaptive wavelet algorithm against the solution found using a set of uniform cubic B-splines. We plot the objective function cost and maximum translation error against the number of finite-element state variables used in each of the solutions, as seen in Figure 6. In contrast to the uniform cubic B-spline solution, where the number of knots used in the solution is gradually increased, the progression of sample points for the adaptive wavelet algorithm corresponds to the maximum wavelet resolution level used in each iteration of the adaptive algorithm, $L = 0 \ldots 10$. The final wavelet sample, near 2000

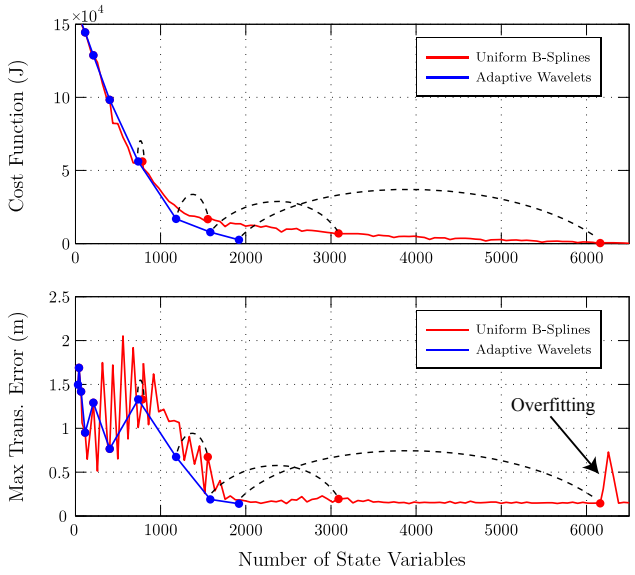

Fig. 6. This figure shows the objective function cost, J, and the maximum Euclidean translation error for solutions to the simulated dataset. The blue line shows the progression of the adaptive wavelet algorithm, where each sample corresponds to an iteration of the adaptive scheme. The red line shows the uniform B-spline solution with a gradually increasing number of knots. We place markers at the uniform B-spline solutions with bases, $\mathbf{\Phi}_{\ell+1}$, that have an equivalent representations (using $\mathbf{W}_\Omega$) in the *available* vector space of an adaptive wavelet solution. By verifying that the maximum error of corresponding solutions is near identical, we can establish that the adaptive algorithm has not smoothed any important details; the difference in state size is the compression benefit gained by using the adaptive algorithm.

state variables, marks the end of the adaptive scheme, as the cost function evaluated to less than the expected value in all time segments.

Although the cost function of the uniform B-spline method reduces quite smoothly, the error term varies wildly depending on how well the control points align with the corners in the solution; this is most visible during the sub-1500 state variable range. Furthermore, we see the effect of overfitting occur in the uniform B-spline solution near the 6200 state variable mark.

In order to show the compressive ability of the adaptive algorithm, we place markers at the uniform B-spline solutions with bases, $\mathbf{\Phi}_{\ell+1}$, that have an equivalent representations (using $\mathbf{W}_\Omega$) in the *available* vector space of an adaptive wavelet solution. By verifying that the maximum error of corresponding solutions is near identical, we can establish that the adaptive algorithm has not smoothed any important details. Furthermore, a corresponding uniform B-spline solution with similar, or possibly higher, error and a lower objective function cost, is likely overfitting segments of the solution that did not require additional detail. The advantage of using a richer *available* vector space is shown by the difference in objective function cost and error between wavelet and uniform B-spline solutions that use a similar number of state variables. Although we see marginal error improvement at the final wavelet solution, we note that without knowledge of the true trajectory it is difficult to know when error has plateaued. Assuming we must rely on

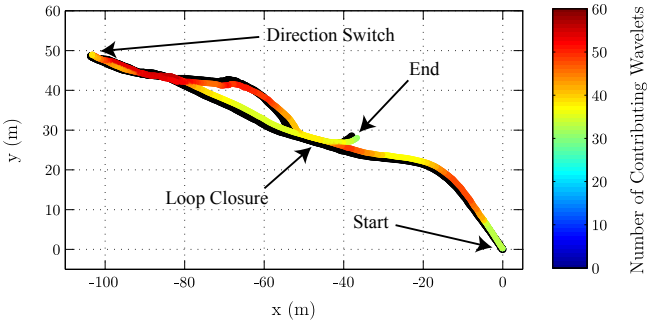

Fig. 7. This figure shows the estimated (coloured) and DGPS (black) trajectory used for the 200m experimental trajectory. The estimated solution is coloured based on the number of actively contributing wavelet coefficients at each sample time. Note that the wavelet density tends to be higher near turns and high frequency motion.

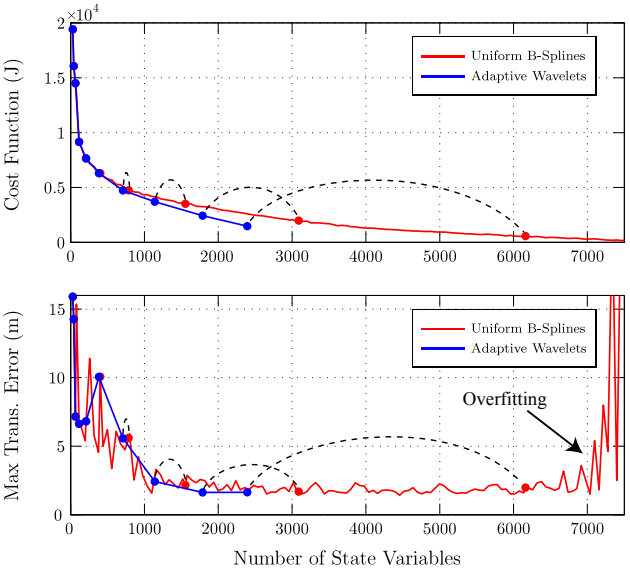

Fig. 8. This figure shows the objective function cost, J, and the maximum Euclidean translation error for solutions to the experimental dataset. In a similar fashion to Figure 6, the blue line shows the progression of the adaptive wavelet algorithm, where each sample corresponds to an iteration of the adaptive scheme and the red line shows the uniform B-spline solution with a gradually increasing number of knots. Again, we place markers at the uniform B-spline solutions with bases, $\mathbf{\Phi}_{\ell+1}$, that have an equivalent representations (using $\mathbf{W}_\Omega$) in the *available* vector space of an adaptive wavelet solution. By verifying that the maximum error of corresponding solutions is near identical, we can establish that the adaptive algorithm has not smoothed any important details Notably, the smaller objective function cost of corresponding uniform B-spline solutions suggest that smooth regions of the solution are overfit.

the objective function cost, J, and its expected value, $E[J]$, we note that the uniform B-spline solution does not reach the same objective function stopping point until it has used roughly 5000 state variables.

C. Real Data Results

In order to verify that our technique works in practice, we applied an identical methodology to a set of real data acquired using a ROC6 mobile rover, as seen Figure 1, at a sand and gravel pit in Sudbury, Ontario, Canada. Specifically, we used a 200m traversal from *The Gravel Pit*

Lidar-Intensity Imagery Dataset [19]. During the experiment, the robot travelled between 0.0 and 0.5 m/s and used an Autonosys LVC0702 lidar to capture intensity and range images, with a resolution of 480×360, at 2Hz. A Differential Global Positioning System (DGPS) was used to collect the ground-truth data used to validate our estimation results. A SURF implementation was used to extract features from the intensity imagery and then 3D pose estimates (and uncertainties) were generated using a motion-compensated RANSAC algorithm [20].

Similar to the simulation analysis, we plotted the estimated trajectory against the ground truth and coloured it based on the number of actively contributing wavelet coefficients at each sample time, as seen in Figure 7. Again, we see that areas involving turns generally required a higher density of wavelet functions. However, the analysis is no longer as straightforward; we note that even during seemingly straight segments there appears to be an influx of active wavelet functions. Although not well demonstrated in the simulation example, changes in attitude from driving over uneven terrain and even sharp variations in robot velocity can cause the solution to require additional detail.

The objective function cost and maximum translation error, seen in Figure 8, exhibit similar findings to that of the simulation example. A key difference is that these results clearly show marked uniform B-spline solutions, with bases $\mathbf{\Phi}_{\ell+1}$, having similar error to the corresponding wavelet solutions, but a lower objective function cost. This suggests that the uniform B-spline solutions are overfit in regions that are sufficiently smooth. Furthermore, this effect is noticeable at the second last pair of corresponding solutions, which occurs before the proposed objective function stopping point. We draw attention to the result that overfitting real data can results in dramatic error spikes (seen past 6500 state variables); this emphasizes the need for an adaptive technique to add detail only where it is required.

V. Conclusion and Future Work

This paper has explored the novel application of a hierarchical wavelet basis to the continuous-time SLAM problem. Using multiresolution analysis, we are able to decompose the continuous-time solution into a coarse shape function and a set of hierarchical detail functions. Furthermore, by selecting a wavelet basis with local support we note the ability to adaptively add detail to the solution only where it is required. This allows us to find a high-quality solution using a smaller state vector than traditional methods. In order to validate the technique, our adaptive wavelet algorithm was applied to both simulated and real data. The next steps in extending this work are to apply it to a full bundle adjustment style problem, and to try using a wavelet basis in the relative, continuous-time SLAM formulation. By performing multiresolution analysis in the velocity domain, it may be possible to gain even larger state compression benefits.

ACKNOWLEDGMENT

We would like to extend our deepest thanks to the NSERC and the Canada Foundation for Innovation, DRDC Suffield, the Canadian Space Agency, and MDA Space Missions for providing us with the financial and in-kind support necessary to conduct this research.

References

[1] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustment - a modern synthesis," *Vision algorithms: theory and practice*, pp. 153–177, 2000.

[2] P. T. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, USA, 14-18 May 2012, pp. 2088–2095.

[3] C. H. Tong, P. T. Furgale, and T. D. Barfoot, "Gaussian process gauss-newton for non-parametric simultaneous localization and mapping," *Intl. Journal of Robotics Research*, vol. 32, no. 5, pp. 507–525, 2013.

[4] E. J. Stollnitz, T. D. Derose, and D. H. Salesin, *Wavelets for computer graphics: theory and applications*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1996.

[5] M. Bosse and R. Zlot, "Continuous 3d scan-matching with a spinning 2d laser," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2009, pp. 4312–4319.

[6] R. Zlot and M. Bosse, "Efficient large-scale 3d mobile mapping and surface reconstruction of an underground mine," in *Proceedings of the International Conference on Field and Service Robotics (FSR)*, Matsushima, Japan, 16-19 July 2012.

[7] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1104–1119, 2012.

[8] H. Dong and T. D. Barfoot, "Lighting-invariant visual odometry using lidar intensity imagery and pose interpolation," in *Proceedings of the International Conference on Field and Service Robotics (FSR)*, Matsushima, Japan, 16-19 July 2012.

[9] J. Hedborg, P. Forssén, M. Felsberg, and E. Ringaby, "Rolling shutter bundle adjustment," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1434–1441.

[10] C. Bibby and I. Reid, "A hybrid slam representation for dynamic marine environments," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 257–264.

[11] M. Fleps, E. Mair, O. Ruepp, M. Suppa, and D. Burschka, "Optimization based imu camera calibration," in *In Proc. of the IEEE Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011, pp. 3297–3304.

[12] S. Anderson and T. D. Barfoot, "Towards relative continuous-time slam," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, 6-10 May 2013.

[13] G. Sibley, C. Mei, I. Reid, and P. Newman, "Adaptive relative bundle adjustment," in *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[14] L. Oth, P. T. Furgale, L. Kneip, and R. Siegwart, "Rolling shutter camera calibration," in *Proc. of The IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, Portland, USA, June 2013.

[15] S. J. Gortler and M. F. Cohen, "Hierarchical and variational geometric modeling with wavelets," in *Proceedings of the 1995 symposium on Interactive 3D graphics*. ACM, 1995, pp. 35–ff.

[16] K. Qin, "General matrix representations for b-splines," *The Visual Computer*, vol. 16, no. 3, pp. 177–186, 2000.

[17] A. Cohen, I. Daubechies, and J.-C. Feauveau, "Biorthogonal bases of compactly supported wavelets," *Communications on pure and applied mathematics*, vol. 45, no. 5, pp. 485–560, 1992.

[18] P. T. Furgale and T. D. Barfoot, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Transactions on Robotics*, 2014. Accepted on December 9, 2013. Manuscript # TRO-13-0419.

[19] S. Anderson, C. McManus, H. Dong, E. Beerepoot, and T. D. Barfoot, "The gravel pit lidar-intensity imagery dataset," Technical Report ASRL-2012-ABL001, UTIAS, 2012.

[20] S. Anderson and T. D. Barfoot, "Ransac for motion-distorted 3d visual sensors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, 3-7 November 2013.