An Observer for Infinite Dimensional 3D Surface Reconstruction that Converges in Finite Time

Sean G. P. O'Brien [*] **Katrina Ashton** [*] **Jochen Trumpf** [*]

[*] *College of Engineering and Computer Science, Australian National University, ACT, 0200, Australia. (email: {sean.obrien,katrina.ashton,jochen.trumpf}@anu.edu.au)*

Abstract: This paper proposes a method of reconstructing the dense structure of scenes from visual or depth sensors that provably converges in finite time. We represent the scene as a superlevel set of a function that resides within some potentially infinite-dimensional function space. The observer state is determined by the parameters of the function that represents the scene. Preliminary experiments show that the observer exhibits convergence behaviour on a variety of different function spaces both in simulation and with real light-field camera data.

Keywords: Nonlinear observers, Infinite-dimensional systems, Computer vision.

1. INTRODUCTION

Scene reconstruction is the task of estimating the dense 3D structure of a sensor's surroundings using measurements obtained from that sensor. Reconstructions have a wide variety of practical applications, including robotic mapping, 3D scanning and inspection of objects, medical imaging, and augmented reality.

The vast majority of the literature on the subject of estimating scenes and objects comes from the computer vision community. In that community, the goal is typically to produce dense reconstructions of objects from their sparse point-cloud representations, and the topic is known as 3D reconstruction or surface reconstruction. Typically, the point clouds that these methods use must be oriented, so that they specify positions in space and corresponding directions normal to the surface to be estimated. The points are assumed to lie on the boundary of the set representing the object and each direction specifies the direction of the gradient of the characteristic function of the set at the corresponding point. The approach introduced by Kazhdan et al. (2006), known as Poisson surface reconstruction, then solves Poisson's equation using this gradient information in order to estimate the characteristic function. Other techniques exploit this information in order to simplify estimates of coefficients of Fourier series, as in Kazhdan (2005), or wavelet representations as in (Manson et al., 2008). Recently, in Mescheder et al. (2018), neural networks have been trained on large datasets in order to produce a measurement-dependent characteristic function that takes as input a point and a measurement and returns a likelihood that the point is occupied given that measurement. All of the mentioned techniques take hours of computation time on dedicated hardware.

The robotics community has also developed techniques for solving this problem over the last few decades. A standard technique, known as occupancy grid mapping was introduced by Thrun and Bü (1996). This technique studies the assignment of function values to discrete voxels. Typically these techniques have a Bayesian flavour, coming from the perspective of machine learning, and the functions being estimated are conditional probability distributions. While the majority of these methods estimate functions defined on a regular voxel grid, recent progress has been made on continuous occupancy mapping techniques (see Ramos and Ott (2016), Senanayake and Ramos (2018)). Again, the literature on this topic comes from a probabilistic perspective. Although the experimental results of these methods are promising, theoretical guarantees of the correctness of 3D reconstruction or occupancy mapping techniques are not provided in these papers.

To the authors' knowledge, it has not before been recognised that occupancy grid mapping techniques are observers, albeit observers with trivial state dynamics. The ramifications of this observation include potentially adding internal models to these techniques in order to produce online dense 4D reconstructions of evolving environments. 4D reconstruction is yet another developing topic within computer vision that concerns estimation of the dense geometry of a scene together with its time evolution. Most published methods on 4D scene reconstruction are performed offline in post-processing (Mustafa et al. (2016)). As with the occupancy mapping approaches, there is good experimental evidence that these methods produce accurate results, but theoretical proofs of convergence are not supplied.

In this paper, we derive an observer that estimates characteristic functions of scenes, in a way that does not depend on the function class of which the characteristic function is assumed to be a member. We prove that the derived observer exhibits point-wise finite-time convergence from

[⋆] This research was supported by the Australian Research Council through the ARC Discovery Project DP160100783 "Sensing a complex world: Infinite dimensional observer theory for robots."

dense measurements of the scene, such as those that may be obtained from a light-field camera or laser range finder, under certain assumptions. We further show that any update function with the right properties will result in a converging scene estimate. Finally, we demonstrate that the derived observer works in simulation and on real light-field camera data.

The remainder of this paper is structured as follows: in Section 2 we develop a theoretical framework for the observer, starting with implicit representations of scenes using extended characteristic functions, parametrisations of the space of functions used to represent the scenes, errors of scene estimates, and measurements of scenes. In Section 3 we develop a general observer for scene reconstruction that applies regardless of the scene representation chosen, and derive different instances of the observer for several chosen function classes: voxels, wavelets, and neural networks. In Section 4 we demonstrate that the observer exhibits convergence behaviour both in simulation and with real light-field camera data. In Section 5 we prove that this observer can estimate points on the scene in finite time, even if the function class chosen to represent scenes itself is infinite dimensional.

2. SCENES AND REPRESENTATIONS

Scene reconstruction is the task of estimating the environment that a sensor resides within. An environment may have many different properties, such as location of objects, lighting conditions, texture and colours of objects, and so on. The aspects of the environment that are estimated will depend on what information can be obtained from the sensor being used. In this paper, we will only estimate the geometry of the environment within a region of space, and the texture of the objects within that region.

2.1 Scenes and Parametrisations

The part of the environment that we wish to estimate is called a *scene*. A scene is typically treated as a subset of $\mathbb{R}^3$. The set of all subsets of $\mathbb{R}^3$ is too large and does not possess enough structure to be practical, so it is necessary in implementation to restrict estimates of scenes to a chosen subclass $\mathbf{X}$ of the powerset of $\mathbb{R}^3$. The choice of *scene class* $\mathbf{X}$ is usually the result of a choice of some constraint or model that we assume the scene must satisfy. However, it is also often the case that even with such a constraint, a scene in $\mathbf{X}$ still requires an infinite number of parameters to specify.

In order to estimate a particular scene $X \in \mathbf{X}$ in an optimisation framework, some parametrisation of the scene class $\mathbf{X}$ needs to be chosen. In this paper, we represent scenes implicitly using a class of parametrised functions. There are two ways in which a set X can be represented by a function. The first way is to take some known set A and take the image of this set under some function $\chi_\theta : A \to P$ whose parameters θ we may vary. Such a representation is known as an explicit representation of the set. An alternative method is to let $\chi_\theta : P \to A$ and let X be the preimage (not to be confused with the image produced by a camera) of some set $B \subset A$, so that $\chi_\theta(X) = B$. Such a representation is called an implicit representation.

In this paper, we represent a scene X implicitly as the zero superlevel set of some function $\chi : P \to \mathbb{R}$, and the scene surface ∂X by the zero level set of the same function. By zero superlevel set, we mean the set of all points $p \in P$ such that $\chi(p) \geq 0$. We call the function χ an extended characteristic function. However, as a result of this representation method, there are likely to be many functions χ that represent the same scene because they have the same zero superlevel set and the same zero level set. Thus, there is an equivalence relation $\chi_1 \sim \chi_2$ if $\chi_1^{-1}(\mathbb{R}^+) = \chi_2^{-1}(\mathbb{R}^+)$ and $\chi_1^{-1}(0) = \chi_2^{-1}(0)$. Every element in the equivalence class $[\chi]$ represents the same scene $X = \chi^{-1}(\mathbb{R}^+)$.

2.2 Errors of Scene Estimates

The fact that a scene is a set poses some challenges to the notion of convergence of estimates. One standard way of defining the distance between two sets is with the Hausdorff distance, which is the largest distance between a pair of points taken from each set. However, designing an observer that uses Hausdorff distance to derive an innovation term is challenging because it will not be resistant to outliers in the measurement data.

In this paper, we exploit our representation of the scene as a function, and define the distance between a scene estimate $\hat{X} \subset P$ and the true scene $X \subset P$ with extended characteristic functions $\hat{\chi}$ and χ, respectively as:

$$E([\hat{\chi}], [\chi]) := \int_P |\mathrm{sgn}(\hat{\chi}(p)) - \mathrm{sgn}(\chi(p))|^2 \, dp. \qquad (1)$$

Note that this distance is well-defined as a distance on equivalence classes of extended characteristic functions.

2.3 Measurements of Scenes

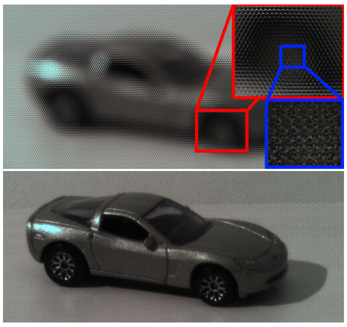


Fig. 1. (Top) A raw light-field measurement μ of a scene. Zoomed portions of this raw data are highlighted to show the image consisting of thousands of densely-packed lenslet images each consisting of hundreds of pixels. (Bottom) An image extracted from the raw data μ.

The observer implemented in Section 4 uses measurement data produced by a light-field camera. A brief overview of these sensors is given in this subsection, but is not essential to understanding the obsever design methodology, as the proposed observer does not operate directly on the raw data produced by these cameras but on depth maps extracted from these cameras. While we prove in Section 5 that depth maps can be correctly extracted from such data, depth maps obtained from other sensors may be used instead. Light-field cameras are sensors that densely sample light passing through a region of space. A light-field camera is modelled by a set $\mathcal{L} \subset \mathbb{R}^2$ called the lenslet plane, and a set $\mathcal{P} \subset \mathbb{R}^2$ called the pixel plane, together with the pose $\xi \in SE(3)$ of the camera, and a tuple $\Phi = (K_1, K_2, f^x, f^y, c^x, c^y, k_1, k_2)$ of real numbers called the *intrinsic parameters* of the camera (note that an additional radial distortion parameter is used in this paper), see O'Brien et al. (2018). A measurement produced by the light-field camera is a function $\mu : \mathcal{L} \times \mathcal{P} \to [0,1]^3$ that implicitly depends on the pose ξ of the camera and the intrinsics Φ and whose values represent the colours of recorded pixels (Figure 1).

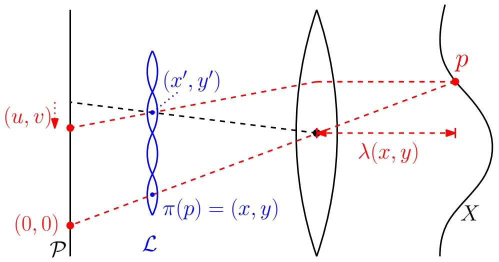


Fig. 2. Light-field geometry: a point p is imaged by lenslets (x, y) and (x', y'). Since the ray that passes through (x', y') and p passes through the optical centre of the focus lens, it has offset $(0, 0)$ and we set $\pi(p) = (x', y')$. The ray that passes through the lenslet (x, y) is refracted by the focal lens and appears in the subimage produced by the lenslet (x, y) at pixel (u, v). The depth $\lambda(x', y')$ assigned to lenslet (x', y') is the depth of the point q on the scene surface ∂X. Observe that ${}^{\mathbf{C}}p^z < \lambda(\pi(p))$ as the point p is in front of the scene.

Let $(x, y) \in \mathcal{L}$ and $(u, v) \in \mathcal{P}$. The coordinates $(x, y, u, v) \in \mathcal{L} \times \mathcal{P}$ are called a *lenslet-pixel pair* and specify a line in space that passes through the lenslet with coordinates (x, y) and the pixel with coordinates (u, v), see Figure 2. If $p \in \mathbb{R}^3$ is a point in front of the camera, and if (x, y, u, v) are the lenslet-pixel coordinates of the ray that passes through the point p, then it is well-known in the light-field literature that there is a quantity δ called disparity that depends only on the depth of the point p for which the rays with lenslet-pixel coordinates $(x+\delta\Delta u, y+\delta\Delta v, u+\Delta u, v+\Delta v)$ all pass through the point p for all $(\Delta u, \Delta v) \in \mathbb{R}^2$ (Wanner and Goldluecke (2014)).

Under the standard Lambertian assumption that the colour of a point does not depend on the perspective it is viewed from, the light-field measurement μ is constant on the set $\{(x+\delta\Delta u, y+\delta\Delta v, u+\Delta u, v+\Delta v) : (\Delta u, \Delta v) \in \mathbb{R}^2\}$, which defines a plane in $\mathcal{L} \times \mathcal{P}$. Thus, the gradient of the light-field is normal to this plane. Therefore, by calculating gradients of the light-field measurement, disparity can be obtained, and so too can depth (for the details see Proposition 3 in Section 5).

The depth-map $\lambda : \mathcal{L} \to \mathbb{R}^+$ is a map that takes a lenslet with coordinates (x', y') and returns the depth ${}^{\mathbf{C}}q^z$ expressed in the body-fixed frame $\mathbf{C}$ of the camera of the first point q on the scene surface ∂X that lies along the ray with coordinates $(x', y', 0, 0)$, see Figure 2. It is also useful to define the centre perspective projection π that maps a point p in front of the camera to the location of the lenslet (x', y') for which the ray $(x', y', 0, 0)$ passes through p, see Figure 2. The coordinates (x', y') may or may not correspond to an actual lenslet in $\mathcal{L}$. If they do, i.e. if $\pi(p) \in \mathcal{L}$, then ${}^{\mathbf{C}}q^z = \lambda(\pi(p))$ is the depth of the scene in direction of p.

Although the experimental results presented in this paper use depth measurements obtained from light-field camera data, it is neither essential that this particular sensor is used, nor that depth is explicitly computed. What is important is that as long as the measurement μ can be used to define an update with the properties described in Section 3, the derived observer will produce an estimate of the extended characteristic function χ that asymptotically converges to the equivalence class $[\chi]$.

3. OBSERVER DESIGN

We assume that the true scene X actually resides within our scene class $\mathbf{X}$, so that there are parameters θ such that $\chi_\theta^{-1}(\mathbb{R}^+) = X$. We assume that the scene is stationary in our experiments and analysis. While an assumption of a stationary scene is a standard one in most SLAM and 3D reconstruction algorithms, our observer approach allows the introduction of non-trivial dynamics into the model. However, in this paper we assume that the parameters that represent the true scene are constant, that is

$$\dot{\theta}_t = 0. \tag{2}$$

At time t, we receive a partial measurement of the scene surface in the form of a depth-map λ_t that is computed from the light-field measurement μ_t. At time t, we also have a parameter estimate $\hat{\theta}_t$ that determines a scene estimate $\hat{X}_t = \hat{\chi}_{\hat{\theta}_t}^{-1}(\mathbb{R}^+)$. From the depth measurement λ_t, we may compute the error of the current parameter estimates given the current depth estimates by computing what the sign of the current extended characteristic function estimate is, and what the depth measurement says the sign of the function value should be:

$$\epsilon(\hat{\theta}_t, \lambda_t) := \int_{\pi_t^{-1}(\mathcal{L})} (\operatorname{sgn}(\hat{\chi}_{\hat{\theta}_t}(p)) - \operatorname{sgn}({}^{\mathbf{C}}p^z - \lambda_t(\pi_t(p))))^2 dp. \tag{3}$$

Ideally, the observer dynamics would be written in the standard innovation term form $\dot{\hat{\theta}}_t = -\nabla_1 \epsilon(\hat{\theta}_t, \lambda_t)$, where the internal model term is zero according to Equation (2). The gradient of the integrand may not be well-defined due to the presence of the 'sgn' function, however, we still pursue the idea of updating $\hat{\theta}_t$ in the direction that minimises the error $\epsilon(\hat{\theta}_t, \lambda_t)$.

To do this, we initialise the extended characteristic estimate so that

$$\chi_{\hat{\theta}_0}(p) = 0 \text{ for all } p \in P. \tag{4}$$

and calculate $\dot{\hat{\theta}}_t$, so that the following is satisfied:

$$\operatorname{sgn}(\dot{\chi}_{\hat{\theta}_t}(p)) := \begin{cases} \operatorname{sgn}({}^{\mathbf{C}}p^z - \lambda_t(\pi_t(p))) & \pi_t(p) \in \mathcal{L} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Under certain mild conditions (see Section 5), this observer will converge in finite-time (see Prop. 2).

The way in which the derivative of the parameters is computed depends on the choice of representation used.

In this paper, we test this method using voxel, wavelet, and neural network representations and show that there are practical ways of implementing this method for each of these choices. Sometimes, in the case where the relationship between the parameters and function outputs is trivial as in voxels, or linear as in wavelets, there are techniques for simplifying the calculation of new parameters from the old parameters. However, even when the relationship between the parameters and function outputs is nonlinear, as in neural networks, there are still techniques for easily computing the update of the parameters.

3.1 Voxel Representation

In this section, we demonstrate the estimation of an extended characteristic function using a voxel representation. In this case, the set of functions is given by $\{\chi : P \to \mathbb{R}\}$, and P is a discrete grid of 3D points $p = (i, j, k) \in \mathbb{Z}^3$ where $i_{\min} \leq i \leq i_{\max}, j_{\min} \leq j \leq j_{\max}, k_{\min} \leq k \leq k_{\max}$. A function $\chi \in \mathcal{F}$ is determined by the parameters $\theta_p := \chi(p)$ where $p \in P$. In this case, the extended characteristic function is updated directly with

$$\dot{\hat{\theta}}_p := \begin{cases} \operatorname{sgn}({}^{\mathbf{C}}p^z - \lambda_t(\pi_t(p))) & \pi_t(p) \in \mathcal{L} \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

3.2 Curvelet Representation

Curvelets were constructed with the goal of finding sparse representations of functions that have discontinuities along C^2-curves, as is common in image processing and computer vision tasks. Whereas classical wavelets are functions that, under translation and scaling, form a basis of $L^2(\mathbb{R})$, curvelets are functions that under translation, parabolic scaling, and rotation form a *Parseval frame* of $L^2(\mathbb{R}^2)$. A Parseval frame for $L^2(\mathbb{R}^2)$ is a family of functions $\{\phi_i\}_{i=1}^{\infty}$ that satisfy Parseval's identity, namely that for all $\psi \in L^2(\mathbb{R}^2)$ we have that $\sum_{i=1}^{\infty} |\langle \psi, \phi_i \rangle|^2 = ||\psi||^2$. In the curvelet literature, curvelets are often said to form a tight frame, however this is not to be confused with other notions of a tight frame which only require Parseval's relation to hold up to a constant scale (see Christensen (2016)).

We will not give a complete description of how a curvelet is constructed in this paper, a more comprehensive overview of curvelets can be found in Candès et al. (2006). Curvelets are typically constructed by taking a mother curvelet φ, and defining the curvelet family $\varphi_{j,k,l}$, that depends on the parameters $j \in \mathbb{N}$, $l \leq 2^j \in \mathbb{N}$, and $k \in \mathbb{Z}^2$. The family of functions $\varphi_{j,k,l}(x) := 2^{3j/2}\varphi(D_j R_{j,l} x - k_\delta)$, where D_j is a parabolic scaling matrix, $R_{j,l}$ is a rotation matrix, and k_δ is a translation depending on a predetermined fixed parameter δ form a Parseval frame of $L^2(\mathbb{R}^2)$. This notion may be extended to construct a Parseval frame of $L^2(\mathbb{R}^3)$, for more details see Ying et al. (2005).

To use a curvelet representation, consider an extended characteristic function $\chi \in L^2(P)$, where P is some rectangular prism in $\mathbb{R}^3$. Since $\chi \in L^2(P)$, it has a curvelet expansion

$$\chi(p) = \sum_{j,k,l} \theta_{j,k,l} \varphi_{j,k,l}(p). \tag{7}$$

The parameters of the extended characteristic function χ in this representation are the coefficients

$$\theta_{j,k,l} = \int_P \chi(p) \varphi_{j,k,l}(p) dp. \tag{8}$$

In order to update these coefficients given a depth measurement λ_t, we approximate the coefficients $\Delta\hat{\theta}_t$ of $\dot{\chi}_{\hat{\theta}_t}$ by computing the curvelet coefficients of the 'update' function

$$v_t(p) := \begin{cases} \operatorname{sgn}({}^{\mathbf{C}}p^z - \lambda_t(\pi_t(p))) & \pi_t(p) \in \mathcal{L} \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

We exploit the sparsity of the coefficients for real scenes, by keeping only the N most significant coefficients in the state after applying the update, and setting the rest to 0, which helps eliminate noise that may be present in the measurements (see Section 4.1).

3.3 Neural Network Representation

A feed-forward neural network is a function $\chi : \mathbb{R}^m \to \mathbb{R}^n$ that is a finite iterated composition of functions of the form

$$p \mapsto \sigma_l(A_l p + b_l),$$

where $\{\sigma_l\}_{l=1}^{L}$ are nonlinear functions known as activation functions, A_l is a matrix and b_l is a vector. Let

$$f_l(p) := \sigma_l(A_l p + b_l),$$

then the neural network function is given by

$$\chi(p) := (f_L \circ \cdots \circ f_1)(p).$$

The number of functions in the composition is L and is known as the depth of the network. Given that the activation functions σ_l are chosen beforehand, the parameters of the neural network function are given by the sequence of matrices and vectors: $\theta := (A_l, b_l)_{l=1}^{L}$.

The parameters of this representation are updated by taking a random sample S of points in $\pi_t^{-1}(\mathcal{L})$, pairing each $p \in S$ with an ideal value $y(p)$, and performing backpropogation on the training pairs $\{(p, y(p))\}_{p \in S}$ for a small number of training steps using the error

$$\tilde{\epsilon}(\hat{\theta}_t, \lambda_t) := \sum_{p \in S} \left| \hat{\chi}_{\hat{\theta}_t}(p) - y(p) \right|^2,$$

where $\hat{\chi}_{\hat{\theta}_t}$ denotes the function that is computed by a neural network with parameters $\hat{\theta}_t$.

In effect, this process will approximate $-\nabla_1 \tilde{\epsilon}(\hat{\theta}_t, \lambda_t)$ and update the parameters in the direction of this gradient. The ideal value $y(p)$ assigned to point p will depend on the choice of activation functions used. For example, if the activation function on the final layer is $\sigma_L(h) = 2\tilde{\sigma}(h) - 1$, where $\tilde{\sigma}$ is a sigmoid function, then the range of the neural network function is $(-1, 1)$, in which case letting

$$y_t(p) = \begin{cases} \operatorname{sgn}({}^{\mathbf{C}}p^z - \lambda_t(\pi_t(p))) & \pi_t(p) \in \mathcal{L} \\ \hat{\chi}_{\theta_t}(p) & \text{otherwise} \end{cases}$$

will result in an update that approximates (5).

4. EXPERIMENTS

In this section, we provide both simulated and experimental evidence for the correctness of our approach. The simulated scene is a 3D model of a bas-relief obtained from

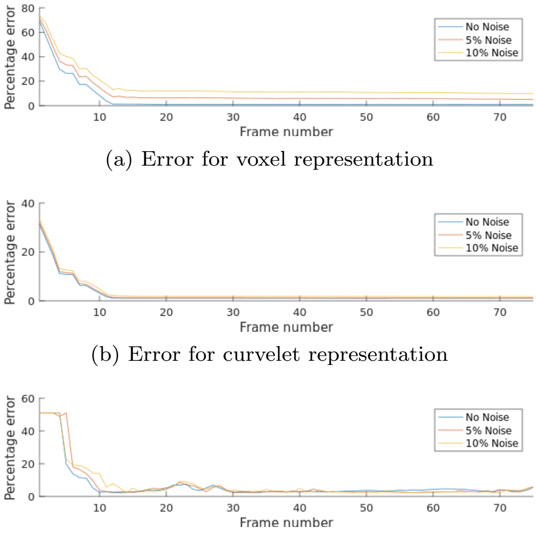
Maier et al. (2017). This dataset is chosen in our simulation because it best reflects the theoretical assumptions on the scene given in Section 5. Depth maps were computed from triangle meshes extracted from this data and used as scene measurements. The domain P is chosen so that the triangle mesh divides the domain into two halves, and from this the true characteristic function can be computed.

We also test the observer on a real scene using data produced by a Lytro Illum camera. However, since ground truth is not available, actual error trajectories cannot be computed for this data. The final 3D reconstructions of the observer are provided instead for visual inspection. A sample central sub-aperture image of the scene is shown in Fig 5 for comparison. The scene consists of an object to be reconstructed and a checkerboard. The checkerboard is used to calibrate the camera, providing both estimates of the camera intrinsic parameters Φ and of the pose of the camera ξ for each frame. A total of 101 frames are used in this experiment.

For each simulated frame, an error comparing the estimated characteristic function with the true characteristic function based on the known scene geometry is reported. This error is defined on the output of the function, not on the parameters of the function. Given a regularly sampled voxel grid G on the input space P, we compute at each time step the approximate error

$$\tilde{E}([\hat{\chi}_t],[\chi]) := \frac{1}{2}\sum_{p\in G} |\mathrm{sgn}(\hat{\chi}_t(p)) - \mathrm{sgn}(\chi(p))|^2 .$$

Graphs of these errors for each of the representation methods are shown in Fig 3. Final reconstructions of the simulated scene are shown in Fig 4.

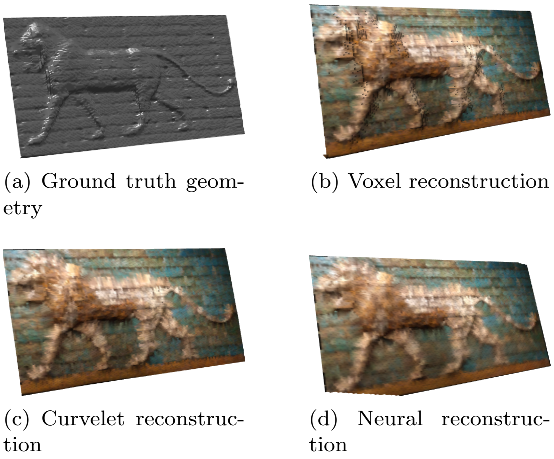


(a) Error for voxel representation

(b) Error for curvelet representation

(c) Error for neural representation

Fig. 3. Error graphs for each representation of the simulated scene.



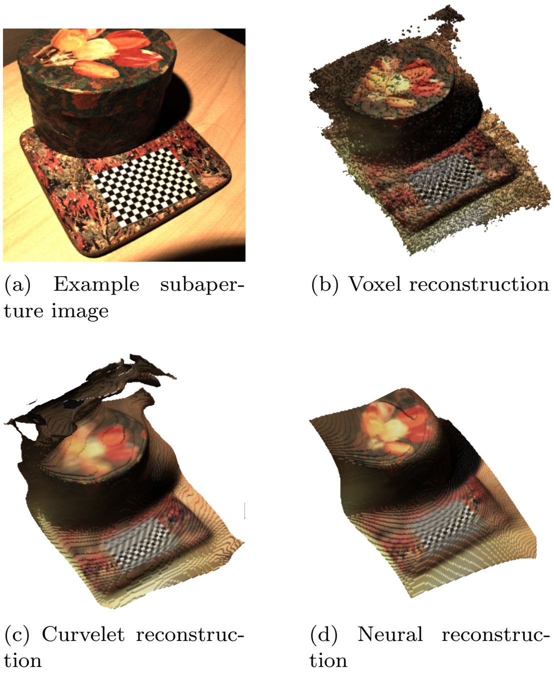
(a) Ground truth geometry

(b) Voxel reconstruction

(c) Curvelet reconstruction

(d) Neural reconstruction

Fig. 4. Comparison of final reconstructions of a simulated scene shown with ground truth.



(a) Example subaperture image

(b) Voxel reconstruction

(c) Curvelet reconstruction

(d) Neural reconstruction

Fig. 5. Final reconstructions using real light-field camera data from a Lytro Illum camera.

We represent the extended characteristic functions in several different ways in order to demonstrate that our approach is not limited to a specific function class or representation. For the voxel representation, the resolution used is 128×128×128. For the curvelet representation, we utilize the Curvelab toolbox to implement a discrete curvelet transform (Candes et al. (2015)). At each timestep, we progressively increase the number of parameters used. The maximum number of parameters used in the curvelet representation is 10000, that is 209 times lower than what is required for the voxel representation. Because the curvelet transform used is not norm-preserving, the scaling factors associated with each curvelet have been robustly estimated as a preprocessing step. At each timestep, the largest curvelet coefficients, taking into account the scaling, are extracted from the current scene estimate. In our experiments, we do not use curvelets of scaling depth greater than 8. The neural network used in the simulations is a fully-connected neural network with 4 hidden layers, each layer consisting of 100, 50, 20 and 10 neurons, respectively. For the neural network representation, the number of training update steps at each time step is 20, and the activation function used at each layer is a sigmoid function.

This update is fast enough to be performed online (on average 0.7 seconds per frame) for a small neural network of 6693 parameters. The parameters for this representation are not initialised to zero, as with the other representations used, but assigned randomly.

In the simulations, the camera trajectory consists of 75 frames and is chosen so that every point on the scene is within the field of view of the camera at least once. By frame 20, each portion of the scene has been seen. The trajectory consists of a fast scan of the entire scene with minimal overlapping between frames followed by a slower scan of the scene for the remaining frames.

The trajectory for the real data is taken by hand, and is constrained more by keeping the checkerboard in view and in focus in order to achieve good calibration results for the data. These calibration results are used to extract pose estimates for the camera. The focal plane for the light-field camera is set to roughly 30 cm.

4.1 Discussion of results

In Fig. 3 we show the error trajectories for the simulated data. It can be seen that the neural network method exhibits a steeper initial descent than the other methods, but the voxel and curvelet trajectories converge much faster than the neural network representation and have a lower final noise floor, as well as less variability at this noise floor. It can be seen that for the voxel and curvelet methods, the estimate approaches the noise floor by the time the fast pass over ends at frame number 12. For the neural network method, there is a brief period where the error does not decrease. This is likely due to some time being required before the parameters of the neural network represent a significantly different state to the inital state. Additionally, the neural network method exhibits oversmoothing and much of the finer details of the bas-relief are lost. The difference between the voxel and curvelet representation graphs is striking, and demonstrates that the parameter thresholding in the curvelet representation results in significant noise reduction.

Fig. 4 presents the final reconstructions of the simulated scene for each of the methods together with the ground truth data. The final results for the voxel and curvelet methods are similar despite the latter using 200 times fewer parameters. The neural network method exhibits oversmoothing, but still produces a good approximation of the scene. It is likely that different activation functions and more sophisticated network architectures would produce superior results in this approach.

Fig. 5 presents the final reconstructions of the real data together with a subaperture image of the scene. The curvelet method seems to reduce noise when compared to the voxel method, but can result in artifacts towards the edges of the bounding box. This is a known phenomenon in the curvelet literature (Candès et al. (2006)). As with the simulated data, the neural network method seems to exhibit excessive smoothing.

5. THEORETICAL ANALYSIS

In this section, we prove that the observer converges pointwise in finite time, despite the fact that the state is infinite-dimensional, if we can update the output values of the characteristic function directly. To do this, we make several mild assumptions, however these are not necessarily the weakest assumptions under which the observer converges. Further analysis of the behaviour of the observer when only the parameters may be updated is the subject of future work. We also show that the observer can be implemented using light-field measurement data.

Notation In this section, we use the following notation for line segments. For two distinct points $p_1, p_2 \in \mathbb{R}^3$, let $[p_1, p_2]$ denote the line segment starting at p_1 and ending at p_2, that is

$$[p_1, p_2] := \{p' \in \mathbb{R}^3 : p' = p_1 + \alpha(p_2 - p_1) : \alpha \in [0, 1]\}.$$

We denote the set of positions assumed by the lenslets of the camera by K, so that if $\mathcal{L}_t \subset \mathbb{R}^3$ denotes the embedded lenslet plane at time t, then $K = \bigcup_{t \in \mathbb{R}^+} \mathcal{L}_t$.

Assumptions on the scene There are several assumptions that are necessary in order to prove convergence of the scene estimate to the true scene. The first assumption we need is that we are estimating the portion X of some larger star-shaped scene X' that is contained within a rectangular prism P.[1]

Assumption 1. The portion X of the scene that is to be estimated is given by $X = X' \cap P$, where X' is star-shaped and P is a rectangular prism.

The next assumption is necessary in order for several of the maps used in the proof to be differentiable, as well as to guarantee boundedness of the depth map.

Assumption 2. The total scene surface $\partial X'$ is a manifold that is diffeomorphic to the sphere S^2.

Since $\partial X'$ is diffeomorphic to the sphere, the Jordan-Brouwer separation theorem says that $\mathbb{R}^3 \setminus \partial X'$ is equal to the disjoint union of two separated sets called the interior I which is bounded, and the exterior E.

Assumptions on the camera trajectory The following assumption is a persistency of excitation condition, and is a constraint on the camera trajectory.

Assumption 3. For all $p \in P$ there exists a $t > 0$ and a positive number $\delta > 0$ such that $\pi_s(p) \in \mathcal{L}$ for all $s \in (t, t + \delta)$.

That is: for each point that point is updated at least once, continuously for some interval of time. We also assume that every point in P is always in front of the camera.

Assumption 4. The depth of every point $p \in P$ satisfies ${}^{\mathbf{C}}p^z > 0$ for all times $t \geq 0$.

Finally, we assume that every point of the total scene surface $\partial X'$ is visible from the position of the camera at every time.

Assumption 5. K is contained in the kernel of the interior I of $\partial X'$.

Proof of convergence The proof of convergence uses the following approach. Firstly, we show that any point $p \in P$

[1] Note that because X' is not being estimated, this assumption is far stronger than neccessary. It is only necessary for it to be *possible* that X is contained in a star-shaped set X'.

can be unambiguously said to be in front of, behind, or on the scene in a way that does not depend on a particular choice of perspective $k \in K$ (Proposition 1). Then we note that for points in front of the scene the characteristic value can only decrease, for points behind the scene the value can only increase, and for points on the scene the value is always zero. This leads to our point-wise finite-time convergence result (Proposition 2).

Proposition 1. Use Assumptions 1, 2, and 5, and let

(1) P^- be the set of points $p \in P$ such that for all $k \in K$ the line segment $[k, p]$ does not intersect $\partial X'$ (the *visible set*), and
(2) P^+ be the set of points $p \in P$ such that for all $k \in K$ the line segment $[k, p]$ does intersect $\partial X'$ (the *occluded set*).

Then $P \setminus \partial X = P^- \cup P^+$.

Proof. By Assumptions 1, 2 and 5, the total scene $\partial X'$ is star-shaped and K is within the kernel of the interior I. Let P^- and P^+ be the sets defined in the statement of the proposition.

Let $p \in (P \setminus \partial X) \cap E$ and let $k \in K$. Since $k \in I$ and $p \in E$, and I and E are separated, then because $[k, p]$ is connected, it contains a point that is in neither set. Since $\partial X' = (I \cup E)^c$, we have that there is some $x \in [k, p]$ such that $x \in \partial X'$. Therefore, $[k, p] \cap \partial X' \neq \emptyset$ for all $k \in K$ and all $p \in (P \setminus \partial X) \cap E$. Therefore $(P \setminus \partial X) \cap E \subset P^+$.

Let $p \in (P \setminus \partial X) \cap I$ and let $k \in K$. Assume, to arrive at a contradiction, that $[k, p] \cap \partial X' \neq \emptyset$. Since $k, p \in I$, the line segment $[k, p]$ must cross through the boundary $\partial X'$ at least twice, but this is not possible because I is a star-shaped set and k is in the kernel of I. It follows that $[k, p] \cap \partial X' = \emptyset$ for all $k \in K$ and all $p \in (P \setminus \partial X) \cap I$. Therefore $(P \setminus \partial X) \cap I \subset P^-$.

Note that the sets $(P \setminus \partial X) \cap I$ and $(P \setminus \partial X) \cap E$ partition $P \setminus \partial X$ because $I \cup E = \mathbb{R}^3 \setminus \partial X'$ and $\partial X = P \cap \partial X'$. Also note that $p \in P^+$ implies $p \notin P^-$ and vice versa simply by the definitions of these sets. Therefore, $(P \setminus \partial X) \cap E = P^+$ and $(P \setminus \partial X) \cap I = P^-$. This shows that $P \setminus \partial X = P^- \cup P^+$.

□

Proposition 2. Let $\dot{\chi}_{\hat{\theta}_t}(p)$ be an integrable function satisfying Eqns. (4) and (5). Use Assumptions 1, 2, and 5, and let P^- and P^+ be the sets defined in Proposition 1. Then, under Assumptions 3 and 4 all of the following hold:

(1) For all $p \in P^-$ there exists a time $T \geq 0$ such that $\chi_{\hat{\theta}_\tau}(p) < 0$ for all $\tau > T$,
(2) For all $p \in P^+$ there exists a time $T \geq 0$ such that $\chi_{\hat{\theta}_\tau}(p) > 0$ for all $\tau > T$,
(3) For all $p \in \partial X$, we have that $\chi_{\hat{\theta}_t}(p) = 0$ for all $t \geq 0$.

Proof. The third statement follows immediately from $\dot{\chi}_{\hat{\theta}_t}(p) = 0$ for all $t \geq 0$ and all $p \in \partial X$, and the fact that $\chi_{\hat{\theta}_0}(p) = 0$.

To show the first statement, let $p \in P^-$. Then there exists a time $t > 0$ and a $\delta > 0$ such that $\pi_s(p) \in \mathcal{L}$ for all $s \in (t, t + \delta)$ (Assumption 3).

For a given $s \in (t, t + \delta)$ let $k \in \mathcal{L}_s \subset K$ denote the embedded location of $\pi_s(p) \in \mathcal{L}$ and let $q \in \partial X'$ be the point that lies on the half-line starting at k and passing through p (Assumptions 1 and 5). Since $p \in P^-$ the line segment $[k, p]$ does not intersect $\partial X'$ hence the distance of p from k is less than the distance of q from k. Now, due to Assumption 4, we have that the depth ${}^{\mathbf{C}}p^z$ of point p is also less than the depth ${}^{\mathbf{C}}q^z = \lambda_s(\pi_s(p))$ of point q.

Therefore, the value of ${}^{\mathbf{C}}p^z - \lambda_s(\pi_s(p))$ is negative on the interval $(t, t + \delta)$. Now, since $\pi_s(p) \in \mathcal{L}$ for $s \in (t, t + \delta)$, the derivative $\dot{\chi}_{\hat{\theta}_s}(p)$ of $\chi_{\hat{\theta}_s}(p)$ is negative on this interval. Therefore, we have that $\chi_{\hat{\theta}_{t+\delta}}(p) = \chi_{\hat{\theta}_t}(p) + \int_t^{t+\delta} \dot{\chi}_{\hat{\theta}_s}(p) ds < \chi_{\hat{\theta}_t}(p)$ because the integral is negative.

But since the derivative of $\chi_{\hat{\theta}_t}$ at p is always either negative or zero, and $\chi_{\hat{\theta}_0}(p) = 0$, we have that $\chi_{\hat{\theta}_t}(p) \leq 0$ to begin with. Therefore, $\chi_{\hat{\theta}_{t+\delta}}(p) < 0$. Let $T := t + \delta$ and note that for all future times $\tau > T$, we have that $\dot{\chi}_{\hat{\theta}_\tau}(p) \leq 0$ and hence $\chi_{\hat{\theta}_\tau} < 0$.

The statement for $p \in P^+$ follows along the same lines.

□

A careful inspection of the previous proof shows that if Assumption 3 is strengthened to require the existence of a finite time $T_{\max} > 0$ such that for all $p \in P$ the corresponding $t + \delta < T_{\max}$ then the entire scene estimate converges in finite time $T_{\max}$. More generally, any portion of the scene is reconstructed as soon as it has been seen continuously for some time.

Proof of implementability from light-field measurements In this section, we show that depth measurements may in principle be perfectly estimated using light-field data. To show this, we require an assumption on the colour distribution of our light-field measurement.

Assumption 6. The light-field measurement $\mu : \mathcal{L} \times \mathcal{P} \to [0, 1]^3$ is differentiable and satisfies $||\nabla \mu_{xy}(x, y, 0, 0)|| > 0$ for all $(x, y) \in \mathcal{L}$.

This assumption is fulfilled for Lambertian scenes whose colouring has non-zero gradient everywhere. The existence of such a colouring for any given (smooth) scene surface is guaranteed by a theorem of Hirsch (1961).

The following proposition proves correctness of depth estimates from light-field measurement data given this assumption.

Proposition 3. Let $\Theta = (K_1, K_2, f^x, f^y, c^x, c^y)$ be the intrinsic parameters of a light-field camera without lens distortion. Use Assumption 6 and define the function $\delta : \mathcal{L} \to \mathbb{R}^+$ as

$$\delta(x, y) := -\frac{\nabla_{xy}\mu(x, y, 0, 0) \cdot \nabla_{uv}\mu(x, y, 0, 0)}{||\nabla_{xy}\mu(x, y, 0, 0)||^2}. \tag{10}$$

Then, the function $\lambda : \mathcal{L} \to \mathbb{R}$ defined as

$$\lambda(x, y) := -\frac{K_2}{K_1 + \delta(x, y)}$$

is equal to the depth of the first point on the scene surface that lies along the ray with coordinates $(x, y, 0, 0)$.

Proof. In O'Brien et al. (2018), it was shown that the depth of a point p with $\pi(p) \in \mathcal{L}$ expressed in body-fixed coordinates $\mathbf{C}$ of the camera is given by

$$\mathbf{c}_{p^z} = -\frac{K_2}{K_1 + \frac{R(p)}{r}}, \tag{11}$$

where $R(p)$ is the plenoptic disc radius of p, and r is the subimage radius. In the same paper it was shown that (noting that in that paper, pixel coordinates are defined absolutely on the raw image rather than relative to a lenslet coordinate)

$$\begin{pmatrix} u_1 - u_2 \\ v_1 - v_2 \end{pmatrix} = \frac{r}{R(p)} \begin{pmatrix} x_1 - x_2 \\ y_1 - y_2 \end{pmatrix}$$

holds for all pairs of lenslet-pixel coordinates (x_1, y_1, u_1, v_1) and (x_2, y_2, u_1, v_2) imaging the same point p. Comparing this to the defining equation

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \delta(p) \begin{pmatrix} \Delta u \\ \Delta v \end{pmatrix}$$

for the disparity $\delta(p)$ of p, it follows that

$$\delta(p) = \frac{R(p)}{r}$$

and hence Equation (11) can be rewritten as

$$\mathbf{c}_{p^z} = -\frac{K_2}{K_1 + \delta(p)}. \tag{12}$$

The light-field measurement $\mu : \mathcal{L} \times \mathcal{P} \to [0,1]^3$ is constant on the level set

$$\{(x + \delta(q)\Delta u, y + \delta(q)\Delta v, \Delta u, \Delta v) : (\Delta u, \Delta v) \in \mathbb{R}^2\},$$

where q is the first point on the scene surface ∂X that lies along the ray with coordinates $(x, y, 0, 0)$. The gradient of μ is nonzero by Assumption 6 and orthogonal to this level set at $(x, y, 0, 0)$. By expressing $(\Delta u, \Delta v) = \rho\omega$ for some $\omega \in S^2$, $\rho > 0$ we therefore obtain that

$$\delta(q)\omega \cdot \nabla_{xy}\mu + \omega \cdot \nabla_{uv}\mu = 0 \tag{13}$$

which, for any $\omega \in S^2$, has the solution

$$\delta(q) = -\frac{\omega \cdot \nabla_{uv}\mu}{\omega \cdot \nabla_{xy}\mu}. \tag{14}$$

Letting $\omega = \nabla_{xy}\mu(x, y, 0, 0)$, we obtain (10), and the result follows by substituting into Equation (12).

□

6. CONCLUSION

In this paper, we represent a scene as the superlevel set of an extended characteristic function. We then use dense measurements of the scene that are known to correlate with depth, such as those obtained from a light-field camera or laser range finder, in order to update the parameters of the extended characteristic function. We prove that using ideal light-field data, we may in principle perfectly reconstruct a scene under certain mild assumptions using this observer. Regardless of the dimensionality of the representation, the observer estimate converges to the true scene in finite time under the same assumptions. Future work will involve rigorous comparisons with existing benchmarks on wider datasets.

ACKNOWLEDGEMENTS

The authors would like to thank Viorela Ila for directing us towards existing literature on this topic, Rob Mahony for his comments on parts of the theory, and Donald Dansereau for his advice on obtaining light-field data.

REFERENCES

Candès, E., Demanet, L., Donoho, D., and Ying, L. (2006). Fast discrete curvelet transforms. *Multiscale Modeling & Simulation*, 5(3), 861–899.

Candes, E., Demanet, L., Donoho, D., and Ying, L. (2015). Curvelab. http://www.curvelet.org/. Accessed: 2019-11-01.

Christensen, O. (2016). *An Introduction to Frames and Riesz Bases*. Birkhauser Verlag GmbH, Springer.

Hirsch, M. (1961). On imbedding differentiable manifolds in Euclidean space. *Annals of Mathematics*, 73(3), 566–571.

Kazhdan, M. (2005). Reconstruction of solid models from oriented point sets. In *Proceedings of the Third Eurographics Symposium on Geometry Processing*, 73–82.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP, 61–70. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland.

Maier, R., Kim, K., Cremers, D., Kautz, J., and Nießner, M. (2017). Intrinsic3d: High-quality 3d reconstruction by joint appearance and geometry optimization with spatially-varying lighting. In *IEEE International Conference on Computer Vision (ICCV)*, 3133–3141.

Manson, J., Petrova, G., and Schaefer, S. (2008). Streaming surface reconstruction using wavelets. *Computer Graphics Forum*, 27(5), 1411–1420.

Mescheder, L.M., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. (2018). Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*.

Mustafa, A., Kim, H., Guillemaut, J., and Hilton, A. (2016). Temporally coherent 4d reconstruction of complex dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4660–4669.

O'Brien, S.G.P., Trumpf, J., Ila, V., and Mahony, R.E. (2018). Calibrating light-field cameras using plenoptic disc features. *International Conference on 3D Vision (3DV)*, 286–294.

Ramos, F. and Ott, L. (2016). Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent. *The International Journal of Robotics Research*, 35, 1717–1730.

Senanayake, R. and Ramos, F. (2018). Building continuous occupancy maps with moving robots. In *32nd AAAI Conference on Artificial Intelligence (AAAI)*.

Starck, J.L., Candes, E.J., and Donoho, D.L. (2002). The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, 11(6), 670–684.

Thrun, S. and Bü, A. (1996). Integrating grid-based and topological maps for mobile robot navigation. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, AAAI'96, 944–950. AAAI Press.

Wanner, S. and Goldluecke, B. (2014). Variational light field analysis for disparity estimation and super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3), 606–619.

Ying, L., Demanet, L., and Candes, E. (2005). 3d discrete curvelet transform. 351 – 361. International Society for Optics and Photonics, SPIE.