Accurate Rough Terrain Estimation with Space-Carving Kernels

Raia Hadsell, J. Andrew Bagnell, Daniel Huber, Martial Hebert
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

***Abstract*— Accurate terrain estimation is critical for autonomous offroad navigation. Reconstruction of a 3D surface allows rough and hilly ground to be represented, yielding faster driving and better planning and control. However, data from a 3D sensor samples the terrain unevenly, quickly becoming sparse at longer ranges and containing large voids because of occlusions and inclines. The proposed approach uses online kernel-based learning to estimate a continuous surface over the area of interest while providing upper and lower bounds on that surface. Unlike other approaches, visibility information is exploited to constrain the terrain surface and increase precision, and an efficient gradient-based optimization allows for realtime implementation.**

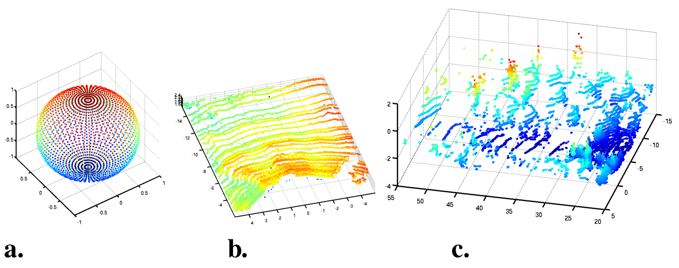

Fig. 1. Evenly sampled 3D objects (left) and laser scans on smooth roadways (center) do not offer the same reconstruction challenges as scans of rough terrain (right), which often have complex structure and very variable resolution that decays rapidly with distance.

I. Introduction

Terrain estimation is a critical component of mobile robot navigation, but accurate reconstruction of rough, hilly, and cluttered terrain is very difficult. The distribution of data points from a 3D sensor on a mobile robot decays rapidly away from the scanner, and there may be large ground regions that return no points at all. This variable resolution is inevitable as it is due to discrete sampling, use of static scanning patterns, and application to terrain whose geometry is unknown *a priori*. Indeed, if the sampling density is dense enough, such as in scanned 3D objects (Figure 1a) or regular enough, such as on smooth roads (Figure 1b), then 3D reconstruction offers less challenge. In rough outdoor terrain, however, complex natural geometry, uneven ground, and inclines all exacerbate the problem and make accurate terrain estimation difficult (Figure 1c).

Variable distributions make terrain estimation very challenging, and many autonomous systems truncate their terrain model to relatively short ranges or make do with a flat, 2D cost map for this reason [8, 7]. Our approach exploits the *visibility* aspect of laser scanning to improve the terrain estimate even in sparse regions. Data points from a ladar sensor must be visible to the sensor; i.e., the rays connecting sensor source to data points must lie *above the terrain surface*. Thus, the elevation function can be constrained by both the ladar points, which must lie *on* the surface, and the ladar rays, which must lie *above* the surface. This can be thought of as a *space carving* approach, since it uses visibility information. The new visibility constraints are incorporated in an reproducing kernel Hilbert space (RKHS) framework rather than a voxel-based approach, yielding a continuous surface estimate with high accuracy that smooths noisy data.

Many approaches simplify the problem considerably by representing the terrain as a flat cost map, but this is insufficient for modeling offroad terrain because hills and rough ground are not accurately represented, forcing the vehicle to drive at very low speeds and make conservative decisions. However, an explicit 3D model of the world, based on points and triangulated meshes, is infeasible: this sort of representation is very expensive, and mesh interpolation over complex terrain is non-trivial. Elevation maps provide a simplification of the full 3D model, but cannot represent overhanging structures and are limited to a fixed resolution by a discretized grid. If the terrain is represented by a continuous elevation *function*, however, then the effect of the ground on the vehicle can be more precisely predicted, allowing for faster autonomous driving and longer range planning. Modeling the terrain as a 3D surface is difficult because of the sparse, uneven distribution of 3D sensor data. Current methods use interpolation to create a continuous mesh surface, but this can be very difficult if the terrain is complex and the data is sparse. In our approach, the 3D surface is modeled as a elevation function over a 2D domain. This surface is estimated using kernel functions, which allow non-linear, complex solutions.

In order to learn the elevation function, we propose a kernel-based approach that models the surface as a hypothesis in a *reproducing kernel Hilbert space* (RKHS). Using a kernel formulation provides a principled means of optimizing a surface function that can produce a highly nonlinear solution. In order to pose the problem as an RKHS regression constrained by visibility information as well as surface points, we incorporate the space-carving constraint into the mathematical framework and give a rule for functional gradient descent optimization, yielding an efficient realtime program. The proposed method is evaluated using LADAR datasets of rough offroad terrain.

II. Related Work

Kernel-based surface estimation has been adopted by the graphics community in recent years for modeling 3D scanned objects [19, 25, 23]. These approaches fit radial basis functions to scanned surface points, yielding an embedding function f. The zero-set $f^{-1}(0)$ implicitly defines the surface. The advantages of using radial basis functions to model the surface of a scanned 3D object are that noise and small holes can be dealt with smoothly, and multi-object interactions can be efficiently computed. However, these approaches cannot be directly applied to terrain data gathered from laser rangefinders mounted on a mobile robot. Such data is dense and precise at close range, but quickly degrades at longer ranges, where the surface is sparsely and unevenly sampled. Given such data, an implicit surface function is ill-constrained and often results in a degenerate solution.

Explicit elevation maps are a standard approach for modeling rough terrain for mobile robotics. There are many strategies for building these maps, from mesh algorithms to interpolation to statistical methods [5, 2, 14].

Burgard et al., following on the research done by Paciorek and Schervish and Higdon et al. [13, 6], have successfully applied Gaussian process regression to the problem of rough terrain modeling, although their approach is computationally expensive and has not been applied to large datasets [16, 15, 12]. Burgard's research adapts Gaussian process regression for the task of mobile robot terrain estimation by considering issues such as computational constraints, iterative adaptation, and accurate modeling of local discontinuity. Our approach uses a kernel-based methodology as well, but we propose an iterative algorithm that exploits both ray and point information to fit basis functions to solve a system of constraints.

Using ray constraints, or visibility information, to improve a 3D surface model has rarely been proposed in mobile robotics. Space carving algorithms, originally suggested by Kutulakos and Seitz [10], use visibility information to produce a voxel model from calibrated images of a scene (a survey of space-carving approach can be found in [21]), but this strategy has not been adopted by the ladar community. Another approach that exploited the visibility constraints was the *locus* method of Kweon et al. [11]. These approaches produced discrete maps, rather than continuous elevation functions, and relied on unwieldy heuristics to ensure that the map had desirable properties such as a watertight surface. In contrast, the approach we propose exploits visibility information while learning a continuous, bounded surface.

In addition, recent publications from several different research groups have advanced the field of 2 and 3D map construction in significant ways. In particular, Yguel et al. proposed the use of sparse wavelets to model a 3D environment from range data [24], and Fournier et al. use an octree representation to efficiently represent a 3D world model [3].

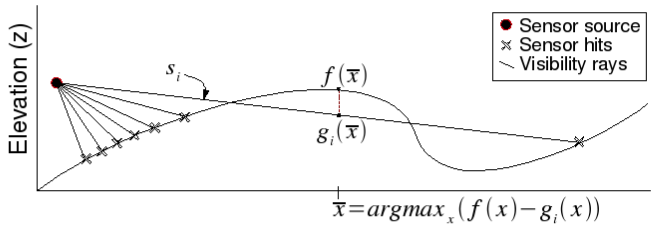

Fig. 2. Visualization in 2D of the point- and ray-based constraints on a terrain elevation function. The estimated surface must intersect the data points ("sensor hits") and lie below the visibility rays from the sensor. If there is a violation of the ray constraint, a support kernel function is added at the most violating point $\bar{x}$.

III. Kernel-Based Regression for Terrain Estimation

Given a set of 3D points from a sensor mounted on a mobile robot, we seek to estimate a continuous elevation function $f(x, y) = z$ that both intersects the data points and does not exceed the height of the rays connecting sensor and data points. A 2D example, in which the elevation map intersects the data points but violates the ray constraint, is shown in Figure 2. The dataset $\mathcal{S}$ consists of n tuples, each with a 3D point (x_i, y_i, z_i) and a corresponding 3D sensor location (sx_i, sy_i, sz_i), which are summarized as a point $\mathbf{x}_i = [x_i \; y_i]$, a height z_i, and a line segment, or ray, connecting source and point, which we denote by $\mathbf{s}_i$. The projection of $\mathbf{s}_i$ on the XY plane is denoted $\hat{\mathbf{s}}_i$, and the function $g_i(\cdot)$ is used to denote the height of $\mathbf{s}_i$ at a given point ($g_i = \infty$ at every location that does not intersect $\hat{\mathbf{s}}_i$). Given this data, we learn a function that meets both a *point*-based constraint (3) and a *ray*-based constraint (4):

$$\begin{aligned} &\text{Given} && \mathcal{S} = \{(\mathbf{x}_1, z_1, \mathbf{s}_1), (\mathbf{x}_2, z_2, \mathbf{s}_2), ..., (\mathbf{x}_n, z_n, \mathbf{s}_n)\} && (1)\\ &\text{find} && f : \mathbb{R}^2 \to \mathbb{R} && (2)\\ &\text{s.t.} && f(\mathbf{x}_i) = z_i \;\; \forall \, \mathbf{x}_i, && (3)\\ & && f(\mathbf{x}) \le g_i(\mathbf{x}) \;\; \forall \, \mathbf{s}_i, \mathbf{x}. && (4) \end{aligned}$$

In order to solve this problem for complex surfaces and datasets, we use a kernel formulation whereby distances between points can be computed in a highly non-linear, high-dimensional feature space without actually computing the coordinates of the data points in that feature space, since any continuous, symmetric, positive semi-definite kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ can be expressed as a dot product in a higher dimensional space [1]. Thus the height function $f(x, y)$ is a hypothesis in RKHS and can be expressed by a kernel expansion:

$$f(x, y) = f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x_i}), \tag{5}$$

where $k(\cdot, \cdot)$ is a radial basis function and α are learned coefficients. For efficiency, the kernel function $k(\cdot, \cdot)$ is a compactly

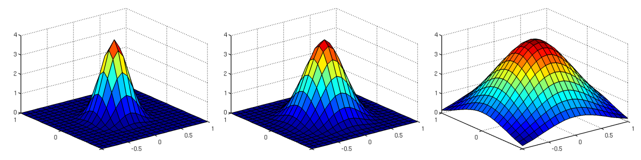
Fig. 3. The Wu $\phi_{2,1}$ function (6), shown with σ of 0.5, 1, and 2.

supported kernel suggested by Wu [18] (see Figure 3):

$$k(\mathbf{x}_i,\mathbf{x}_j) = k(\rho(\mathbf{x}_i,\mathbf{x}_j)) = (1-\rho)_+^4(4+16\rho+12\rho^2+3\rho^3), \tag{6}$$

where

$$\rho(\mathbf{x}_i,\mathbf{x}_j) = \frac{||\mathbf{x}_i-\mathbf{x}_j||}{\sigma}, \tag{7}$$

and σ is the lengthscale of the basis function. This kernel function was chosen over other possible radial basis functions because of the recommendations of Schaback [18] and the empirical success of this kernel for surface estimation in [25].

A. Optimization by Subgradient Descent

To find $f \in \mathcal{H}$ that meets both point-based constraints and ray-based constraints, we form the following convex optimization problem:

$$\mathcal{L}(f,\mathcal{S}) = \lambda||f||_{\mathcal{H}}^2 + \frac{1}{N}\sum_{i=1}^{N} r(\mathbf{x}_i,z_i,g_i,f), \tag{8}$$

where $r(f,g_i,\mathbf{x}_i,z_i)$ penalizes both point and ray constraints. Instead of optimizing an infinite set of linear constraints, i.e., the constraint that each ray must lie above the surface, we optimize a single non-linear constraint and use a max function to find the most offending point in the projection of each ray. The cost function $r(f,g_i,\mathbf{x}_i,z_i)$ is thus:

$$r(\cdot) = \frac{1}{2}\max(0,\max_{\mathbf{x}}(f(\mathbf{x})-g_i(\mathbf{x})))^2+\frac{1}{2}(f(\mathbf{x}_i)-z_i)^2. \tag{9}$$

To solve the optimization problem above, which contains linear and non-linear constraints (because of the max operation), we use a functional gradient descent approach that relies on subgradients to tackle non-differentiable problems. Using a stochastic gradient method allows fast online performance which is critical for real-world application. Online subgradient kernel methods have been successfully applied to several problems [17].

The subgradient method iteratively computes a gradient-like vector which is defined using a tangent to the lower bound at a particular point of the non-differentiable function. There is a continuum of subgradients at each point of non-differentiability, but only one subgradient, which is the gradient, at differentiable points. A detailed examination of the subgradient method can be found in the original work by Shor [20]. Online gradient descent with regret bounds for subgradient methods was developed by Zinkevich [26].

To learn using functional gradient descent, the function is updated with the negative of the (sub)gradient stepped by a learning rate η:

$$f_{t+1} = f_t - \eta_t\frac{\partial\mathcal{L}(f_t,g_t,\mathbf{x}_t,z_t)}{\partial f}. \tag{10}$$

Since f is a hypothesis in RKHS,

$$\frac{\partial\mathcal{L}(f(\mathbf{x}_t),g_t,\mathbf{x}_t,z_t)}{\partial f} = r'(f(\mathbf{x}_t),g_t,\mathbf{x}_t,z_t)k(\mathbf{x}_t,\mathbf{x})+\lambda f. \tag{11}$$

We rely on the kernel expansion of f (Eq. 5) to derive an efficient stochastic update, following the example of [9]. Following this stochastic approach, basis functions are added iteratively, and at the same time the weights of previously added basis functions are decayed. The number and location of the basis functions is not identical to the training points, and the final number of basis functions may be greater or fewer than the sample size. Thus, at time t, a new basis function may be added at location $\mathbf{x}_t$ with coefficient

$$\alpha_t = -\eta_t r'(f(\mathbf{x}_t),g_t,\mathbf{x}_t,z_t), \tag{12}$$

and the existing coefficients are decayed:

$$\alpha_i = (1-\eta_t\lambda)\alpha_i \text{ for } i<t. \tag{13}$$

For our optimization problem, the gradient of r has 2 components, for the different constraints in the loss function, so up to 2 basis functions are added with different coefficients. If $f(\mathbf{x}_i) \neq z_i$, then the added basis function is centered at $\mathbf{x}_i$, with coefficient $\alpha_t = -\eta_t(f(\mathbf{x}_i - z_i)$. The second basis function is added at the most violating location along ray s_i, if one exists. We compute $\overline{x} = \text{argmax}_{\mathbf{x}}(f(\mathbf{x}) - g_i(\mathbf{x}))$ by line search on ray s_i, and if $\overline{\mathbf{x}} > 0$ then a basis function is added at $\overline{\mathbf{x}}$ with a coefficient $\alpha_{t+1} = -\eta_{t+1}(f(\overline{\mathbf{x}}) - g_i(\overline{\mathbf{x}})$. The algorithm is also described in Alg. 1.

The method that has been described in this section uses a fixed lengthscale, σ. A fixed lengthscale with a compactly supported kernel function does not give global support. Many solutions have been proposed for this problem in the graphics community, where radial basis functions are used to estimate 3D surfaces and compactly supported kernels are chosen for efficiency [19, 25, 21]. Most commonly, *backfitting*, a general algorithm that can fit any additive model, is used to decrease the lengthscale incrementally [4]. The training data is partitioned, and each subset is sequentially fit toward the residual of the previous partition using a decaying lengthscale [25]. The same strategy can be employed for our space carving terrain reconstruction algorithm. In fact, there is no need for the training data to be partitioned in our approach: the lengthscale may be decayed on each epoch of gradient-based learning over the full dataset.

B. Uncertainty Bounds

Uncertainty attribution is very valuable for a mobile robot in rough terrain. A terrain estimate coupled with an uncertainty bound is much more powerful than a terrain estimate alone, because the planning and behavior of the robot will be effected. Velocity may be modified, exploration behaviors changed, and safety measures enacted based on the uncertainty

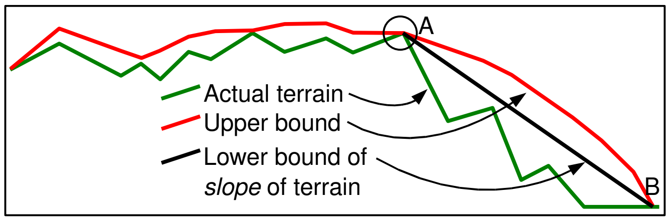

Fig. 4. An upper bound on the terrain gives a lower bound on the slope of the terrain under certain circumstances. For an interior chord to the upper bound ($\overline{AB}$) whose maximum point is coincident with the maximum point on the terrain surface under the chord (at A in the illustration), the slope of the chord is a tight lower bound on the actual slope of the underlying terrain.

attribute. In some circumstances, the upper bound of a region induces a tight lower bound on the slope of the actual terrain, which can be used to identify non-traversable steep slopes, *even if there are no data points in that region*. In other words, the slope of an interior chord of the upper bound is a lower bound on the slope of the actual terrain, *if* the highest elevation of the chord is equal to the local maxima of the terrain under the chord. Thus lethal regions may be located entirely on basis of the upper bound, as long as the upper bound is pinned to the actual surface by at least one data point, an insight described by [8] and provable by application of the mean value theorem. This scenario is depicted in Figure 4: the slope of chord $\overline{AB}$ is a lower bound on the maximum slope of the underlying terrain, because at point A the chord and the terrain are both at local maxima.

Upper and lower bounds can be learned using the kernel-based terrain estimation method that has been proposed. At time 0, the surface estimate is initialized by a positive or negative offset corresponding to the globally maximum or globally minimum expected elevation (in our experiments, +5 meters and -5 meters were the "priors" used to initialize the upper and lower bounds). The subsequent learning is identical to the original algorithm: ray and point constraints are applied to fit the surface by training a kernel function with stochastic gradient-based learning. The outcome is an upper bound surface and a lower bound surface. The three learning processes can be run in parallel on a distributed system.

C. Online Algorithm and Implementation

The algorithm considers each data point on each epoch, although basis functions may not be added if there are no point or ray violations (see Algorithm 1). The learning rate η is reduced at each update step: $\eta_t \propto t^{-\frac{1}{2}}$, and regularization is applied after each learning step. The learning process is halted if the average gradient at time t is below a threshold or if the error rate on a validation set increases, resulting in convergence after a small number of epochs. Since the algorithm is stochastic and iterative in nature, data can be continually added to the training set and the learning will smoothly adapt to the additional information. This is a valuable characteristic for a autonomous navigation system to have, since new data is continuously arriving from sensors.

Another effect of the stochastic, iterative nature of the proposed method is that the learning process can be viewed as an *anytime algorithm* which can be run whenever computational resources and data are available and which does not require the algorithm to terminate in order to access and use the terrain estimate. The upper and lower bounds can be computed in parallel with the surface estimate and thus fit into the *anytime algorithm* formulation. This quality is particularly beneficial for realtime navigation systems that constantly modulate their velocity and trajectory based not only on an estimate of the terrain ahead but also on the uncertainty of that estimate.

```
**Input**: Training set $\mathcal{S}$ (points (Nx3) and sources (Nx3))
**Output**: Surface $f : \mathbb{R}^2 \rightarrow \mathbb{R}$
**while** *Not Converged* **do**
    **foreach** *Point* $\mathbf{x}_i$ **do**
        Calculate $f(\mathbf{x}_i)$;
        **if** $f(\mathbf{x}_i) \neq z_i$ **then**
            Add basis $\mathbf{x}_t = \mathbf{x}_i$;
            With weight $\alpha_t = -\eta_t(f(\mathbf{x}_i - z_i)$;
            incr $t$;
        **end**
        Calculate $\overline{x} = \text{argmax}_{\mathbf{x}}(f(\mathbf{x}) - g_i(\mathbf{x}))$;
        **if** $\overline{\mathbf{x}} > 0$ **then**
            Add basis $\mathbf{x}_t = \overline{\mathbf{x}}$;
            With weight $\alpha_t = -\eta_t(f(\overline{\mathbf{x}}) - g_i(\overline{\mathbf{x}}))$;
            incr $t$;
        **end**
        **foreach** *Basis* $x_j, j < t$ **do**
            Decay weight $\alpha_j = (1 - \lambda_t \eta_t)\alpha_j$;
        **end**
    **end**
**end**
```

Algorithm 1: Online algorithm using subgradients. The first test ($f(\mathbf{x}_i) \neq z_i$) tests for violation of the point constraint, and the second test detects violation of the ray constraint.

One of the benefits of the proposed approach is that it can deliver an accurate, continuous, bounded terrain reconstruction in realtime. The choice of a gradient-based optimization strategy, plus a compact kernel function that allows fast nearest neighbor searching (through use of a kd-tree) to perform the kernel expansion, permit this level of efficiency. The experiments described in this paper were conducted on a 2.2 GHz computer using a non-optimized C++ implementation. The timing results are promising; datasets with 10,000 points converge within a second, and datasets with over 1 million points converge in a few seconds. From this preliminary evaluation, we conclude that deployment on a full realtime navigation system would be feasible.

IV. Evaluation and Analysis

Tests of the proposed method were conducted in order to evaluate the effectiveness of the approach on both artificial and natural datasets, and to demonstrate the convergence properties. Mean squared error is calculated by comparing the elevation of a set of test points with the predicted surface elevation:

$$MSE(\mathcal{S}_{\text{test}}) = E_{\text{test}} = \frac{1}{p}\sum_{i=1..p}(f(\mathbf{x}_i) - z_i)^2, \quad (14)$$

where $\mathcal{S}_{\text{test}}$ is a set of p test points $\mathbf{x}_i$ with known elevations z_i.

A. Evaluation: Sinusoid Data (2D)

A synthetic dataset was used to develop and test the proposed method. Data points were sampled irregularly over a sine function, using an exponentially decaying point density and eliminating points that were not visible from a source point, yielding 400 samples (see Figure 5a). The samples were divided into training and test sets, and testing showed that the algorithm converged and that the space-carving kernels improved the surface estimate. Figure 5b shows the surface and bounds after the first learning epoch and 5c shows the surface after convergence. In addition, a single point was added to the dataset during learning to demonstrate the online capability of this method (Figure 5d).

B. Evaluation: Offroad Natural Terrain Data

The approach described in the previous section has been implemented in Matlab and tested on an outdoor scene using data from the 360° HDL-64 ladar scanner, manufactured by Velodyne and installed on Boss, the Carnegie Mellon entry in the 2008 DARPA Urban Challenge [22]. The left and right images in Figure 6 show a slightly narrowed view of one dataset. Note that the data is very dense within a few meters of the sensor, but quickly degenerates to a sparse, uneven distribution. The terrain in this dataset is extremely rugged, composed of uneven piles of rubble.

The results of reconstructing the terrain are shown in Figure 7, and 2D cross sections of the bounded estimate are shown in Figure 8. The surface was estimated using a training set of 10,000 points (shown as a point cloud in Figure7d), and another set of 5000 points was partitioned for testing. The algorithm was also tested with over one million points to demonstrate computational efficiency. The stochastic algorithm converged after 8 epochs, and a total of 21,392 basis functions were used. To compute the upper and lower bounds, the same training data was learned with an initial surface height of +5 and -5 meters at time 0. The terrain estimate is shown in Figure 7a, the lower bound is shown in 7b, and the upper bound is shown in 7c. The surfaces are similar where data is dense, but the upper and lower bound pull away as the data becomes sparse at the edges of the region shown, and also in the front center of the region, where the robot was occluding the ground.

Looking at the results in 2D sections, as in Figure 8, allows a graphical depiction of the upper and lower bounds, the estimate, and the training points. In this figure, four cross sections of the surfaces shown in Figure 7 are shown, corresponding to the bisections overlaid on the training points in Figure 9. The blue regions in the 2D plots show the uncertainty between the estimated surface and the upper bound on the surface, and the

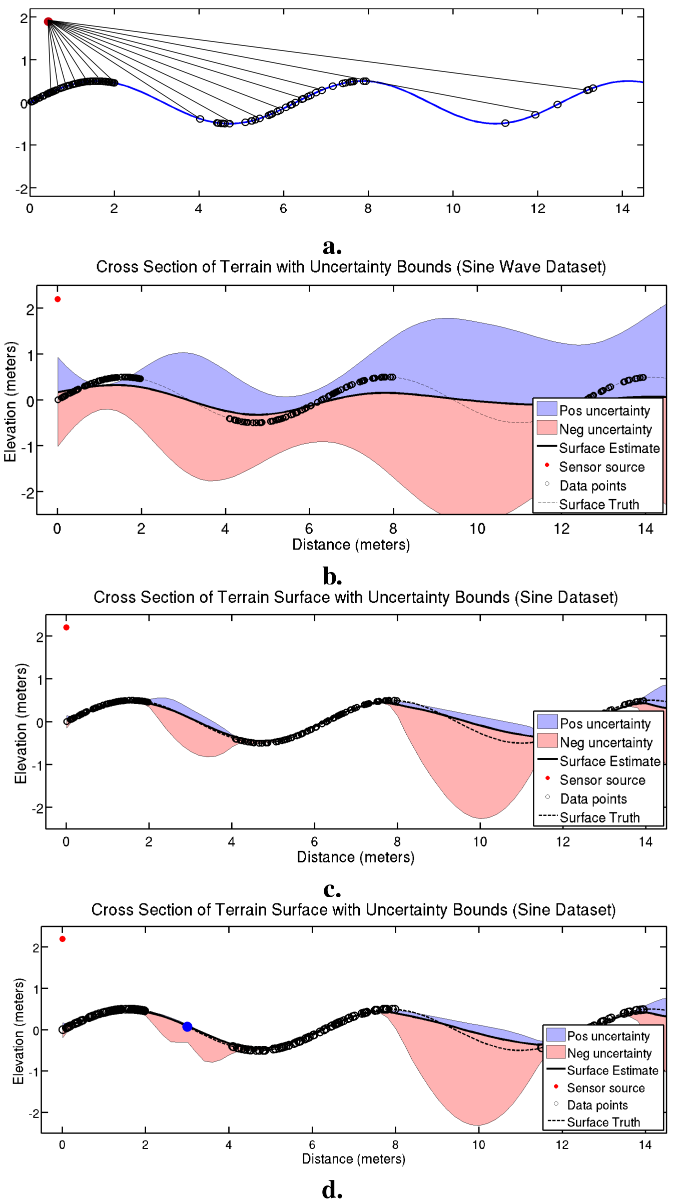

Fig. 5. **a:** An synthetic dataset containing irregularly sampled points on a 2D sine function. The density of points decreases exponentially with distance from the sensor. Points not visible to a sensor were removed. **b:** The estimated surface, plus low and high bounds, after 2 learning epochs. **c:** The estimated surface, plus low and high bounds, after convergence (12 epochs). Note the effect of ray constraints on the upper bound. **d:** A single point (shown in blue) was added interactively to demonstrate the ability of the algorithm to adapt to new data online.

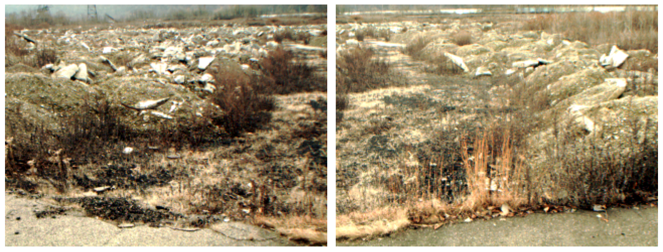

Fig. 6. Left and right views of one of 5 rough terrain datasets used for evaluation of the algorithm.

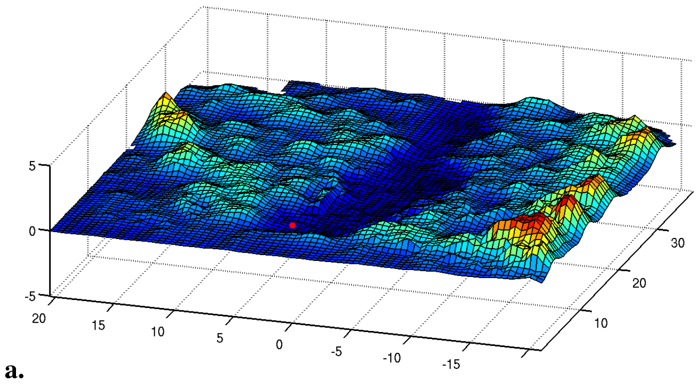
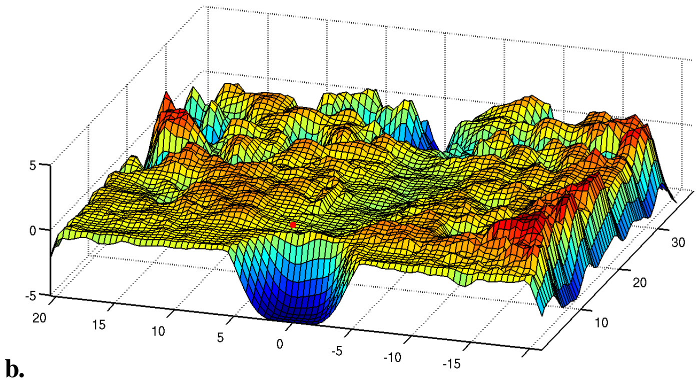
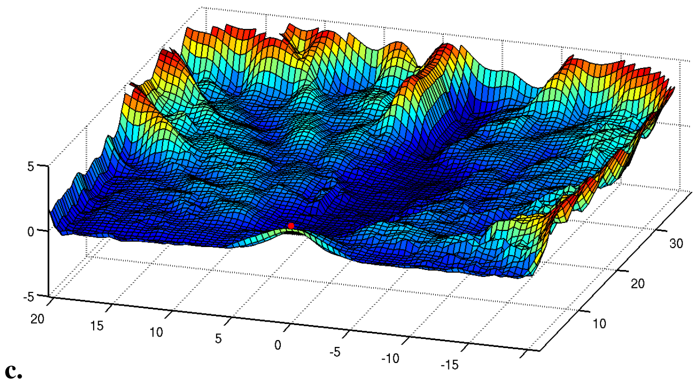
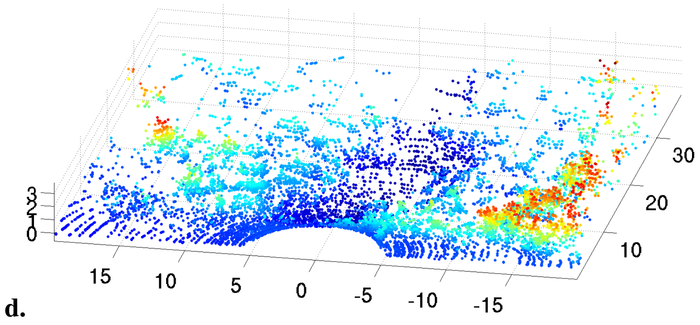

Fig. 7. 3 estimated surfaces, using elevation priors of 0 meters (**a.**), 5 meters (**b.**), and -5 meters (**c.**). Using high and low elevation priors gives uncertainty bounds on the estimated surface. The training set point cloud is shown in **d.**

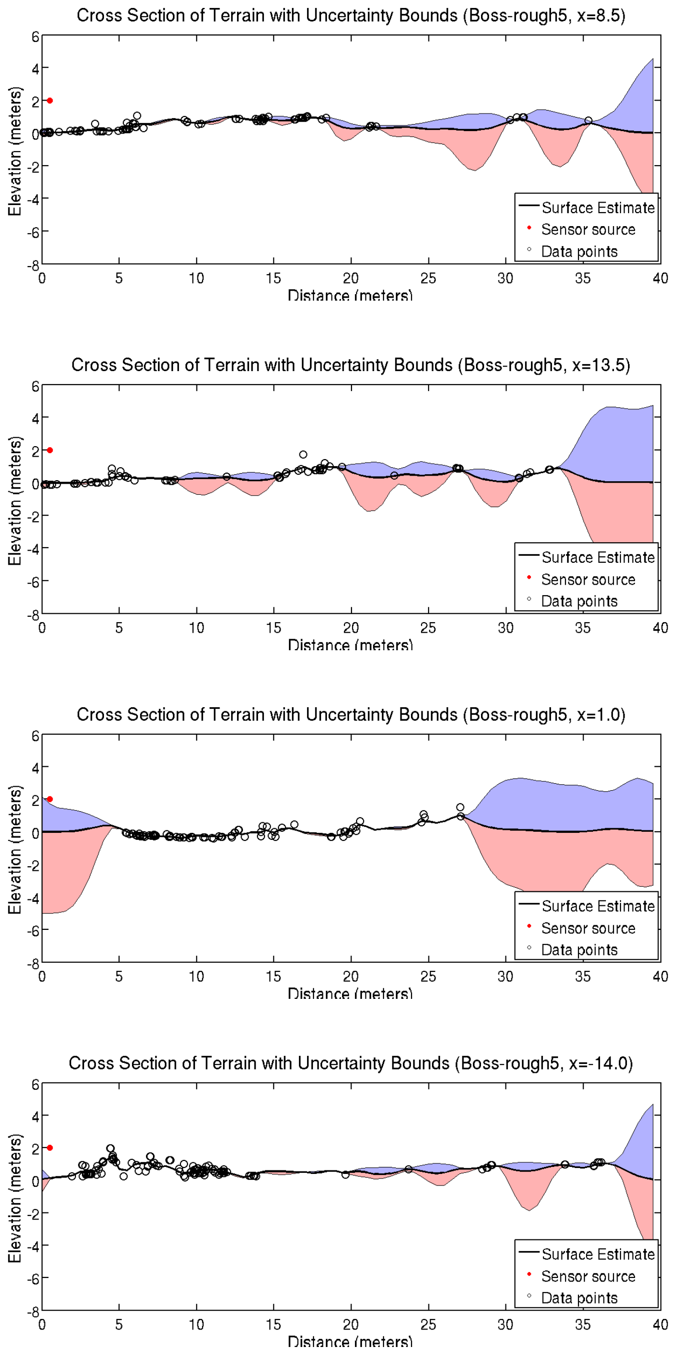

Fig. 8. Various cross sections of the estimated surface from Figure 7 are shown. Upper and lower bounds for the surface are found by combining the surface estimates obtained with different initial elevations. The location of the cross sections in the dataset are shown in Figure 9.

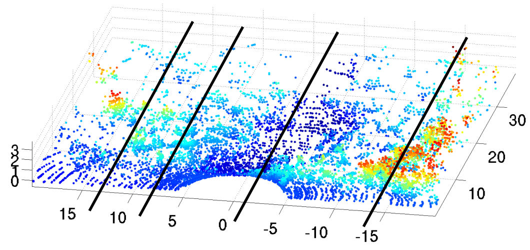

Fig. 9. The locations of the cross sections shown in Figure 8.

TABLE I

ERROR RATES ON FIVE OFFROAD DATASETS.

Dataset	No Visibility Info	With Visibility Info
1	0.059	**0.014**
2	0.045	**0.025**
3	0.120	**0.035**
4	0.098	**0.096**
5	**0.061**	0.065

pink regions show the negative uncertainty. The upper bound is often much tighter than the lower bound because of the ray constraints that give additional constraint on the upper bound, but do not impact the lower bound. In areas where no data at all is given over an area larger than the maximum lengthscale of the kernel (5 meters, in this case), the surface estimate will be at 0 and the upper bound and lower bound will be at 5 and -5 meters respectively, since 0, 5, and -5 are the elevation priors set for the surface and the bounds.

C. Contribution of Visibility Information

To evaluate the contribution of the ray constraints gained from the laser visibility information, five offroad datasets are used. For each dataset, a surface is estimated with and without ray constraints and the mean squared error of a withheld test set is calculated. Ideally, the test set would be uniformly sampled on a grid across the terrain. Instead the test samples are from the same distribution as the training set, which effectively masks the benefit of the ray constraints by not testing in the sparsest areas where the ray constraints have the greatest effect.

Despite this bias, the results given in Table I and Figure 10 confirm the benefit of using visibility information. Using the ray constraints helps substantially on all datasets but one, in which the margin between the two results is very slim. The datasets that are not aided as much by the ray constraints are both flatter terrain, where the potential gain from the visibility information is less because the data is evenly distributed. The convergence of the online algorithms for each data set is plotted in Figure 11.

Inclines are terrain types that are smooth yet problematic for LADAR systems, typically producing very sparse returns because of the high angle of incidence. However, due to that same grazing angle, the ray information is extremely helpful. To elucidate the point, an examination of the surfaces learned for a downhill slope with and without visibility constraints is given. In Figure 12**a.**, the slope is shown (the incline is 30 meters long with a downward slope of roughly 5°). In 12**b.**, the distribution of training data over the test terrain is shown. The data density is very high at the top of the hill, directly in front of the vehicle, and on the embankment at the side of the incline, but almost non-existent toward the bottom of the hill. 12**c.** and 12**d.** show the terrain estimation with and without visibility information. The incline surface is poorly estimated unless the visibility information is used.

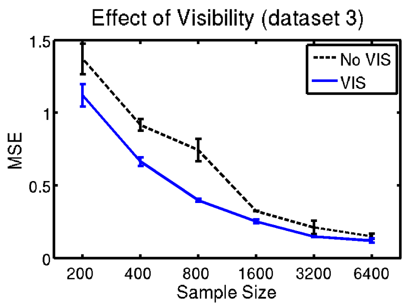

Fig. 10. The performance of the method on a rough terrain dataset. The mean squared error, computed on the test set after convergence of the algorithm, is recorded for increasing sample sizes. The use of ray constraints decreases the error rate at every sample size. Final error rates are given for other datasets (see Table I). Unfortunately, the test data is from the same distribution as the training set, so the benefit of using visibility information is somewhat masked.

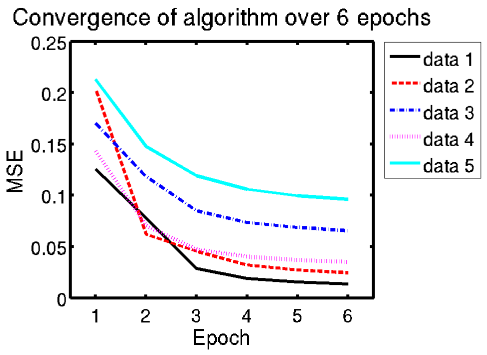

Fig. 11. This graph shows the stable convergence of each natural terrain dataset over 6 epochs.

V. Conclusion

Reconstruction of rough terrain using 3D sensor data is a tough problem because of the highly variable distribution of points in the region of interest. Our proposed approach uses visibility information to carve the surface and produces not only a terrain estimate but also uncertainty bounds. We formulate the learning problem as an RKHS optimization and derive a subgradient-based stochastic solution, which gives computational efficiency, allows data to be added online, and makes the approach an *anytime* algorithm. The evaluation, on both synthetic and natural data, clearly demonstrates the effectiveness of the approach and the utility of the space carving visibility information.

The next phase of this project will incorporate color imagery into the terrain estimation algorithm. The data can be fused

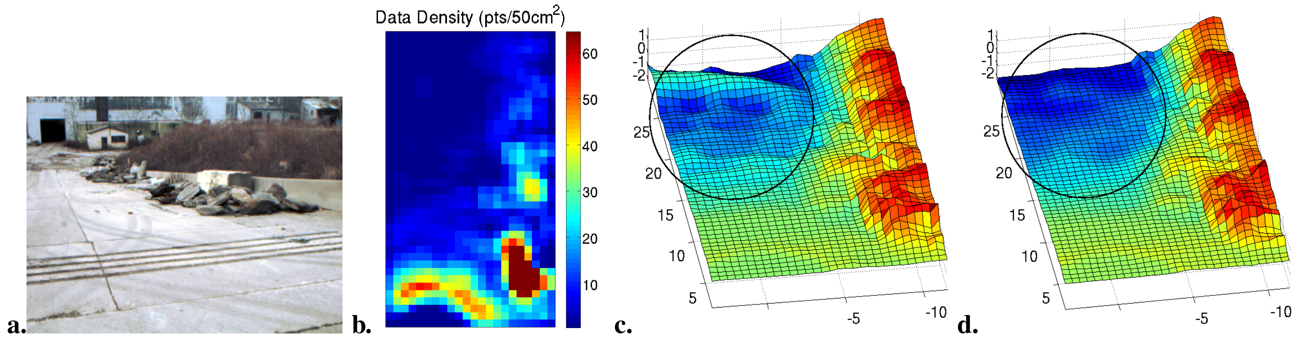

Fig. 12. The laser hits become very sparse if the terrain inclines downward, as for this sloped driveway (**a, b**). If the visibility information is not used, the surface is incorrectly estimated (**c**). When the visibility information is used to constrain the surface, the estimate is far better on the sloping drive (**d**).

using calibration, and the higher resolution, longer range color data can be used to increase both the precision and the range of the terrain model. We also look forward to developing a realtime implementation of the algorithm and expanding the evaluation of the method to include extensive data input over multiple timesteps.

ACKNOWLEDGMENT

This work was supported by the Agency for Defense Development, Republic of Korea.

REFERENCES

[1] M. A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[2] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. R. L. Whittaker. Ambler: An autonomous rover for planetary exploration. *IEEE Computer*, 22(6):18–26, June 1989.

[3] J. Fournier, B. Ricard, and D. Laurendeau. Mapping and exploration of complex environments using persistent 3d model. In *Conference on Computer and Robot Vision*, pages 403–410, 2007.

[4] J. H. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, (76):817–823, 1981.

[5] M. Hebert, C. Caillas, E. Krotkov, I. S. Kweon, and T. Kanade. Terrain mapping for a roving planetary explorer. In *Proc. of International Conference on Robotics and Automation (ICRA)*, volume 2, pages 997–1002, May 1989.

[6] D. Higdon, J. Swall, and J. Kern. Non-stationary spatial modeling. *Bayesian Statistics*, 6, 1998.

[7] L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C. Sullivan. The DARPA LAGR program: Goals, challenges, methodology, and phase I results. *Journal of Field Robotics*, 23(11-12):945–973, 2006.

[8] A. Kelly, A. T. Stentz, O. Amidi, M. W. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski, P. Rander, S. Thayer, N. M. Vallidis, , and R. Warner. *Toward Reliable Off Road Autonomous Vehicles Operating in Challenging Environments*, 25(1):449–483, May 2006.

[9] J. Kivinen, A. J. Smola, and R. C. Williamson. Learning with kernels. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2002.

[10] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000.

[11] I. S. Kweon and T. Kanade. High-resolution terrain map from multiple sensor data. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(2):278–292, 1992.

[12] T. Lang, C. Plagemann, and W. Burgard. Adaptive non-stationary kernel regression for terrain modeling. In *Proc. of Robotics: Science and Systems (RSS)*. MIT Press, 2007.

[13] C. J. Paciorek and M. J. Schervish. Nonstationary covariance functions for Gaussian process regression. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2004.

[14] P. Pfaff, R. Triebel, and W. Burgard. An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *Int. Journal of Robotics Research*, 26(2):217–230, 2007.

[15] C. Plagemann, K. Kersting, and W. Burgard. Nonstationary Gaussian process regression using point estimates of local smoothness. In *Proc. of the European Conference on Machine Learning (ECML)*, Antwerp, Belgium, 2008.

[16] C. Plagemann, S. Mischke, S. Prentice, K. Kersting, N. Roy, and W. Burgard. Learning predictive terrain models for legged robot locomotion. In *Proc. of International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2008.

[17] N. Ratliff, J. D. Bagnell, and M. Zinkevich. (online) subgradient methods for structured prediction. In *Proc. of Conference on Artificial Intelligence and Statistics (AI-STATS)*, 2007.

[18] R. Schaback. Creating surfaces from scattered data using radial basis functions. *Mathematical methods for curves and surfaces*, page 477496, 1995.

[19] B. Scholkopf, J. Giesen, and S. Spalinger. Kernel methods for implicit surface modeling. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2005.

[20] N. Z. Shor, K. C. Kiwiel, and A. Ruszcayǹski. *Minimization methods for non-differentiable functions*. Springer-Verlag New York, Inc., New York, NY, USA, 1985.

[21] G. Slabaugh, R. Schafer, and M. Hans. Multi-resolution space carving using level set methods. In *Proc. of International Conference on Image Processing*, 2002.

[22] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. R. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson. Autonomous driving in urban environments: Boss and the Urban Challenge. *J. Field Robotics*, 25(8):425–466, 2008.

[23] C. Walder, B. Scholkopf, and O. Chapelle. Implicit surface modelling with a globally regularised basis of compact support. *Eurographics*, 25(3), 2006.

[24] M. Yguel, C. T. M. Keat, C. Braillon, C. Laugier, and O. Aycard. Dense mapping for range sensors: Efficient algorithms and sparse representations. In *Robotics: Science and Systems*, 2007.

[25] J. Zhu, S. Hoi, and M. Lyu. A multi-scale Tikhonov regularization scheme for implicit surface modelling. In *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2007.

[26] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, pages 928–936, 2003.