Sparse Depth Super Resolution

Jiajun Lu
University of Illinois at Urbana Champaign
jlu23@illinois.edu

David Forsyth
University of Illinois at Urbana Champaign
daf@illinois.edu

Abstract

We describe a method to produce detailed high resolution depth maps from aggressively subsampled depth measurements. Our method fully uses the relationship between image segmentation boundaries and depth boundaries. It uses an image combined with a low resolution depth map. 1) The image is segmented with the guidance of sparse depth samples. 2) Each segment has its depth field reconstructed independently using a novel smoothing method. 3) For videos, time-stamped samples from near frames are incorporated. The paper shows reconstruction results of super resolution from x4 to x100, while previous methods mainly work on x2 to x16. The method is tested on four different datasets and six video sequences, covering quite different regimes, and it outperforms recent state of the art methods quantitatively and qualitatively. We also demonstrate that depth maps produced by our method can be used by applications such as hand trackers, while depth maps from other methods have problems.

1. Introduction

Recent work in HCI has demonstrated a variety of potential applications for depth sensors on mobile devices (gesture interfaces [15]; deictic references in augmented reality [30]), and several such sensors are in production. However, the relatively high power consumption of current depth sensors even at relatively low resolutions presents major difficulties in making these applications practical. Besides that, further improving the resolution of current depth sensors has high cost, while high resolution RGB images are comparatively cheap.

Therefore, there are two compelling reasons to reconstruct high resolution depth from low resolution samples. First, it could allow current sensors to be operated at lower power consumption, and thus make it practical to use them on mobile devices. Second, it could also provide resolution that current sensor systems cannot. It is cheap in money and in power to obtain high resolution image frames registered to the depth sensor, so it is natural to combine upsampling with image or video information.

This paper describes methods to reconstruct high-resolution depth measurements accurately from sparse scattered samples, see Figure 1. We propose to exploit image (resp. video) data obtained at the same time as the depth samples to produce a spatial model that governs our smoothing of depth samples. Our method segments an image with the guidance of sparse depth samples, then uses novel smoothing methods to reconstruct depth within each segment. For video, we use space-time segments and optic flow methods to move time-stamped samples in space.

We believe that experimental work on depth super resolution has not, to date, explored very aggressive subsampling regimes because authors have focused on improving quite good sensors. Typically, previous works explores $\times 2$ to $\times 8$ times super-resolution, with $\times 2$ super resolution being most common for Kinect depth. In contrast, we greatly push forward the upscale ratio and demonstrate high accuracy in ranges from $\times 12$ to $\times 64$ times super resolution.

Our reconstruction outperforms state of the art methods because it respects important structural cues: 1) within object boundaries (or an area of an object), surfaces are reasonably smooth; 2) image boundaries are a fair approximation to object (object area) boundaries , meaning that one should not smooth over image segment boundaries (over-segmentation is not likely to create problems because obtained depth will "smooth" over boundaries); 3) space-time segmentation of video produces boundaries that behave like object boundaries and object area boundaries, and therefore methods for reconstruction from static images can be extended to reconstruct motion sequences with adjustments.

Contributions: 1) We demonstrate methods that can produce near ground truth depth from aggressive subsampling (for example, one depth sample per 4096 pixels) on both images and videos, outperforming recent strong methods and pushing forward upscale ratio. 2) Our experimental work is conducted on three different widely used publicly available datasets, and one novel dataset that we collected, and previous methods only work on limited data. 3) Our depths with big upscale ratio are successfully used by applications such as hand trackers, while other methods

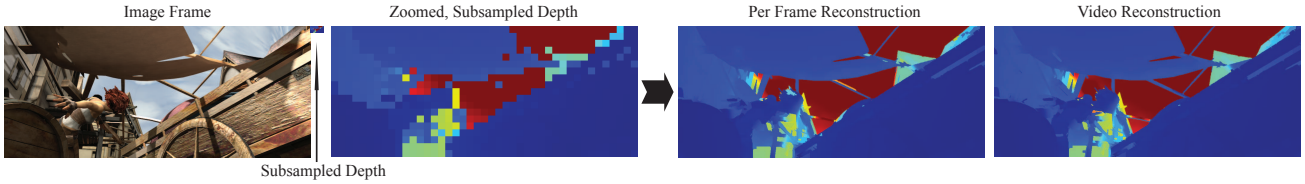

Figure 1. We describe a method to reconstruct high resolution depth maps from aggressively subsampled data, using a smoothing method that exploits image segment information to preserve depth boundaries. Our method is evaluated on four different datasets, and produces state of art results. This example shows a case where there is one depth sample per 24 × 24 block of image pixels (the tiny inset shows the depth map on the same scale as the image). Our method can exploit optic flow and space-time segmentation to produce improved reconstructions for video data.

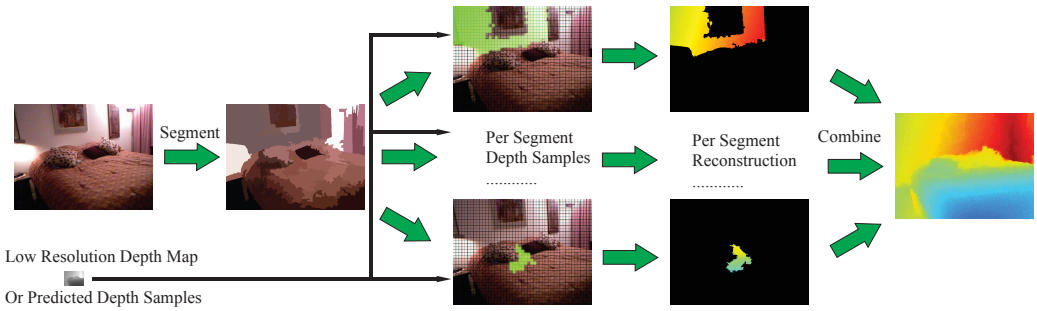

Figure 2. Our method first segments the high resolution image, then reconstructs a high resolution depth map for each segment independently using smoothing methods; finally, these reconstructions are composed.

have problems. 4) Even though our methods are primarily aimed at improving either resolution or power use of active depth sensors by upsampling sensed depth maps, our spatial model is powerful that it can be used to improve the results of existing depth-from-image methods.

2. Related Work

There is a body of work on depth super-resolution methods, exploiting a variety of different approaches.

Markov random field methods can be used to infer high resolution depth from low resolution depth and high resolution intensity images, because the intensity images offer cues to the location of depth discontinuities. Depth map refinement based on MRF was first explored in [3], extended in [21] with a depth specific data term, and combined with depth from passive stereo in [34]. Park et al. [26]add a non-local means term to their MRF formulation to preserve local structure better and to remove outlliers. Aodha et al. [22] treat depth super-resolution as an MRF labeling problem.

Multiple depth maps can be combined to produce a higher resolution depth map. The Lidarboost approach of Schuon et al. [29] combines depth maps acquired from slightly different viewpoints. The Kinect fusion approach of Izadi et al. [14] produces outstanding results by fusing a sequence of depth maps generated by a tracked Kinect camera into a single 3D representation in real-time. Gudmundsson et al . [9] presented a method for stereo and Time of Flight (ToF) depth map fusion in a dynamic programming approach.

Dictionary methods exploit the dependency between sparse representations of intensity and depth signals over appropriate dictionaries. Mahmoudi et al. [23] first learn a depth dictionary from noisy samples, then refine and denoise these samples and finally learn an additional dictionary from the denoised samples to inpaint, denoise, and super-resolve projected depth maps from 3D models. [7] and [31] independently learn dictionaries of depth and intensity samples, and model a coupling of the two signal types during the reconstruction phase. In [18], a joint intensity and depth map model is built to recover the co-structure of image and depth.

RGBD image depth refinement methods exploit image shading to improve raw Kinect depth [33, 10]. Such methods estimate the lighting first, and then use shading information to refine the depth map from Kinect output. Complex spatial albedo maps and complex surface material properties present some difficulties for these methods.

Some other methods include [20][12][6] investigates the relationship between images and depth, and no training data is needed. Space does not allow a reasonable review of **image segmentation** or of **optic flow**. We use the method of [5] because it is stable, easy to implement and runs fast. For video data, we use the video segmentation method of [8], which produces space-time segments. For optic flow estimation, we use the method of [1], which is effective and accurate. We expect other methods would apply as well in each case.

Several recent papers have explored regressing depth

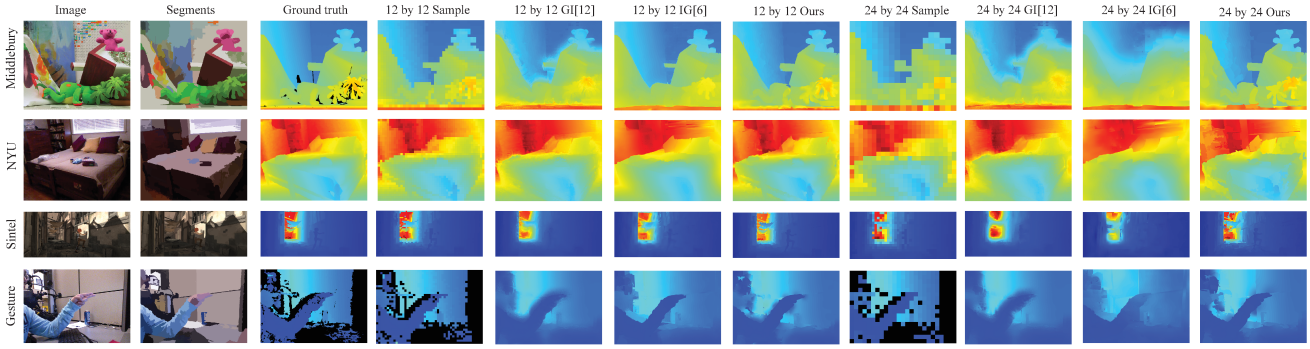

Figure 3. Examples of images from each dataset. Our method works better than [6] [12] on both 12 and 24 times. The results of [6] [12] become bad quickly when the upscale ratio increases, while our method is more stable with the upscale ratio.

against a single image (see review in [16]). The best performance to date has been displayed by a deep network [4].

3. Method

Assume we have an image and scattered depth samples. Our method builds the image segmentation tree, using the scattered depth samples to merge the segments in the tree, then uses smoothing methods to reconstruct depth within each segment independently. For video, we use space-time segments and optic flow methods to move time-stamped samples in space, but otherwise proceed as for static images. Pipeline is in Figure 2.

3.1. Image Segmentation

The segmenter of [5] is a form of agglomerative cluster, and so produces a tree of region merges (known as a dendrogram in the clustering literature). Each image segmentation is a choice of the region merges in the segmentation tree. Previous methods use pixel colors and cross-validation to do this. In our application, we have depth samples, which is useful in identifying segmentations. So we use depth samples to guide the process of region merging.

For each neighboring region pair, we compute a score measuring the consistency between the depth pairs on the segmentation boundary, where a higher score means better segmentation. Write $\mathbf{c}_i$ for the i'th sample location, $d_f(\mathbf{c}_i, \mathbf{c}_j)$ for the difference in depth at the i'th and j'th sample points, and δ and μ for parameters. Write $\gamma(\mathbf{c}_i, \mathbf{c}_j; \delta)$ for

$$\frac{\min(d_f(\mathbf{c}_i, \mathbf{c}_j), \delta)}{\delta}$$

We score the consistency $C_s(\mathbf{c}_i, \mathbf{c}_j)$ of each neighboring sample depth pair using

$$C_s(\mathbf{c}_i, \mathbf{c}_j) = \begin{cases} \mu(e - e^{\gamma(\mathbf{c}_i, \mathbf{c}_j; \delta)}) & \mathbf{c}_i, \mathbf{c}_j \text{ share a segment} \\ e^{\gamma(\mathbf{c}_i, \mathbf{c}_j; \delta)} & \text{otherwise} \end{cases}.$$

We use $\mu = 1$ and fixed δ for each dataset. Notice that larger (resp. smaller) μ values would tend to produce fewer (resp. more) segments. For each image, we first build the segmentation tree with fixed number of levels and minimum area sizes for each dataset. Then, we start with the level that contains the largest segments, and recursively for each segment use the consistency score to decide whether go down to the next level of the segmentation tree. The algorithm is robust to segmentation results. This is mainly because the depth samples can fix segmentation problems, especially over-segmentation.

3.2. Depth Smoothing

We now have a set of image segments, and a collection of depth samples. For each segment, we will smooth the samples inside the segment to form a dense depth map. By using only the samples inside the segment, we can obtain sharp depth boundaries at image segment boundaries. Once each segment has a depth map, we compute an overall depth map by copying the depths from each segment to the image plane.

We describe two methods of smoothing. Our **simple depth smoothing** algorithm is a form of scattered data interpolation. It helps to understand the smoothing system, is faster than the advanced smoothing, and could easily be implemented in parallel. Our **advanced depth smoothing** algorithm allows some large derivatives of depth. Experimental results show the advanced method is better at dealing with complex depth images, poor segmentation, and noise in depth samples. Preliminary, experiments demonstrated the advanced method have an RMSE approximately 5% better than the sample method and all results reported are for the advanced method.

3.2.1 Simple Depth Smoothing

Write $\mathbf{c}_i$ for the location of the i'th sample, $z_s(\mathbf{c}_i)$ for the value at that sample point, $\mathbf{x}$ for a variable point in the image and $z(\mathbf{x}; \mathbf{a})$ for the reconstructed depth function, $\mathbf{a}$ for the parameter vector. We start with a set of radial basis functions centered at each sample point. Write

$\phi(\mathbf{x}; \mathbf{c}_i) = f(\|\mathbf{x} - \mathbf{c}_i\|)$ for the radial basis function centered at $\mathbf{c}_i$. Write $d_{\max}$ as max influence range, we have

$$f(u) = \max(1 - \frac{u}{d_{\max}}, 0)^2$$

We wish to blend depth estimates from nearby samples at a given point $\mathbf{x}$. So it is natural to work with a set of basis functions that add to one at every point. Define

$$\beta(\mathbf{x}; \mathbf{c}_i) = \frac{\phi(\mathbf{x}; \mathbf{c}_i)}{\sum_j \phi(\mathbf{x}; \mathbf{c}_j)}$$

(assuming that $d_{\max}$ is chosen so that at least one ϕ has a non-zero value for all $\mathbf{x}$ in the image).

We assume that there is an initial prior model of depth $z_\pi(\mathbf{x})$. In our experiments, this is usually a zero offset plane, but when the samples are super sparse, using depth maps reconstructed from other methods as frontal plane is useful. However, when images are known to have a particular structure — for example, to be images of rooms as in [13] — it might be useful to have some more complex model. Our depth interpolation is

$$z(\mathbf{x}; \mathbf{a}) = \sum_i a_i \beta(\mathbf{x}; \mathbf{c}_i) + z_\pi(\mathbf{x})$$

where $\mathbf{a}$ are determined by solving the linear system

$$z(\mathbf{c}_j; \mathbf{a}) = \sum_i a_i \beta(\mathbf{c}_j; \mathbf{c}_i) + z_\pi(\mathbf{c}_j).$$

With reasonable choices of $d_{\max}$, this system has full rank, and so the solution yield an interpolation. In our implementation, $d_{\max}$ is twice the average spacing between samples. This means that at any point relatively few ϕ have non-zero values, meaning that evaluating the blending weights is fast.

3.2.2 Advanced Depth Smoothing

There are some difficulties with the simple depth smoothing model. First, it yields an interpolation, which means that noisy depth measurements can seriously disrupt the reconstruction. Second, the basis functions do not encode large depth derivative particularly well. We introduce further basis functions to remedy these problems. At each sample point i we place a unit vector $\mathbf{u}_i$. Then

$$\psi(\mathbf{x}; \mathbf{c}_i, \mathbf{u}_i) = \phi(\mathbf{x}; \mathbf{c}_i)\,(\mathbf{u} \cdot (\mathbf{x} - \mathbf{c}_i))$$

is a basis function with value 0 at $\mathbf{x} = \mathbf{c}_i$. The gradient on position $\mathbf{c}_i$ is large, and it can be steered by choice of $\mathbf{u}_i$. Our smoothed depth model becomes

$$z(\mathbf{x}; \mathbf{a}, \mathbf{b}, \mathbf{u}) = \left(\sum_i a_i \beta(\mathbf{x}; \mathbf{c}_i)\right) + \left(\sum_i b_i \psi(\mathbf{x}; \mathbf{c}_i, \mathbf{u}_i)\right) + z_\pi(\mathbf{x}).$$

We must now choose values of $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{u}$ to produce a reconstruction. We do so by minimizing an objective function that is a sum of three terms, each capturing a natural requirement of the problem. First, we expect that depth samples have relatively low noise, so we expect E_1 to be small.

$$E_1(\mathbf{a}, \mathbf{b}, \mathbf{u}) = \sum_j \left(z(\mathbf{c}_j; \mathbf{a}, \mathbf{b}, \mathbf{u}) - z_s(\mathbf{c}_j)\right)^2$$

Second, we expect that there are relatively few sharp depth gradients, so that we expect E_2 to be small.

$$E_2(\mathbf{a}, \mathbf{b}, \mathbf{u}) = \|\mathbf{b}\|_1$$

The L_1 norm here encourages most of the b_i to be zero.

Finally, at each grid sampling's grid box center $\mathbf{x}_k$, the depths predicted by any two distinct nearby sample points need to be consistent with one another. Write $\mathcal{N}(\mathbf{x}_k)$ for a neighborhood around $\mathbf{x}_k$. We expect that $E_3(\mathbf{a}, \mathbf{b}, \mathbf{u})$ to be small, which is

$$\sum_k \sum_{c_i, c_j \in \mathcal{N}(x_k)} \begin{pmatrix} a_i \phi(\mathbf{x}_k; \mathbf{c}_i) + b_i \psi(\mathbf{x}_k; \mathbf{c}_i, \mathbf{u}_i) \\ -a_j \phi(\mathbf{x}_k; \mathbf{c}_j) - b_j \psi(\mathbf{x}_k; \mathbf{c}_j, \mathbf{u}_j) \end{pmatrix}^2,$$

We choose parameters by optimization, and the weights are chosen by cross-validation.

$$\underset{\mathbf{a}, \mathbf{b}, \mathbf{u}}{\text{argmin}} \; \lambda_1 E_1 + \lambda_2 E_2 + \lambda_3 E_3$$

3.3. Video Data

Our method also applies to video data with some modifications. We use space time segments, because we expect depth to be fairly smooth within a space time segment, but change on its boundaries (for example, an object moving behind an obstacle). We reconstruct from depth samples that are time-stamped (obtained at a certain time, but have effects at a time period). First, consider points nearby in space. We expect the depth at these points to be similar. This justifies using a smoothing where the influence of a sample declines as points get further away. But temporal smoothing is somewhat different. At each depth sample, we can compute optic flow, which is used to predicts the location of the sample forward and backward in time, by moving the depth sample along the flow dirction. For some time interval, we can trust these flow-based predictions, so we transport depth samples along the flow direction before interpolation. We allow the sample to have influence for times up to δt in the future and $-\delta t$ in the past, and the weighting of a sample declines as the inter-frame time interval increases. We use

$$\omega(\Delta t; \delta t, C) = \frac{\min(-\log(|\Delta t|)/\delta t, C)}{C}$$

[log10, RMSE], smaller is better.

Dataset	Middlebury Dataset								NYU Dataset							
Ratio	24×24 times (41,55)		16×16 times (69,82)		12×12 times (92,109)		8×8 times (138,163)		24×24 times (18,24)		16×16 times (27,36)		12×12 times (36,47)		8×8 times (54,71)	
Near	0.0121	0.0366	0.0082	0.0291	0.0056	0.0240	0.0036	0.0189	0.0134	0.1656	0.0111	0.1444	0.0065	0.0977	0.0061	0.0929
Bicubic	0.0139	0.0313	0.0096	0.0246	0.0070	0.0199	0.0047	0.0156	0.0115	0.1333	0.0099	0.1223	0.0055	0.0755	0.0057	0.0780
[12]	n/a	0.0340	n/a	0.0259	n/a	0.0198	n/a	0.0166	n/a	0.1588	n/a	0.1388	n/a	0.1026	n/a	0.1003
[6]	n/a	0.0406	n/a	0.0269	n/a	0.0190	n/a	0.0138	n/a	0.2157	n/a	0.1393	n/a	0.0989	n/a	0.0649
Our	**0.0055**	**0.0186**	**0.0042**	**0.0155**	**0.0034**	**0.0138**	**0.0027**	**0.0119**	**0.0083**	**0.1119**	**0.0051**	**0.0809**	**0.0039**	**0.0660**	**0.0026**	**0.0496**
Dataset	Sintel Dataset								Gesture Dataset							
Ratio	24×24 times (19,43)		16×16 times (28,64)		12×12 times (37,86)		8×8 times (55,128)		24×24 times (17,25)		16×16 times (25,37)		12×12 times (34,49)		8×8 times (50,73)	
Near	0.0278	1.7290	0.0181	1.4092	0.0157	1.3917	0.0085	0.9923	0.0165	0.0284	0.0114	0.0239	0.0092	0.0198	0.0059	0.0157
Bicubic	0.0339	1.4607	0.0226	1.1764	0.0196	1.1932	0.0114	0.8240	0.0180	0.0249	0.0128	0.0205	0.0106	0.0171	0.0068	0.0131
[12]	n/a	1.4461	n/a	1.1586	n/a	1.0219	n/a	0.7975	n/a	0.0249	n/a	0.0207	n/a	0.0193	n/a	0.0176
[6]	n/a	1.5691	n/a	1.1608	n/a	0.9071	n/a	0.6491	n/a	0.0286	n/a	0.0237	n/a	0.0215	n/a	0.0183
Our	**0.0135**	**0.9454**	**0.0088**	**0.7971**	**0.0066**	**0.6940**	**0.0047**	**0.5908**	**0.0123**	**0.0204**	**0.0094**	**0.0168**	**0.0079**	**0.0147**	**0.0056**	**0.0118**

Table 1. Depth error compared to ground truth on four kinds of datasets using nearest neighbor, bicubic, [12], [6] and our method. In the row Ratio, 12×12 means 12 times super resolution in each direction, and (92,109) means there are 92 by 109 boxes in the sample grid. log10 is the mean absolute error of log10 depth. RMSE is root mean square error of recovered depth with respect to the ground truth data.

Δt is the inter-frame time interval, and the function value decreases as $|\Delta t|$ approaches δt.

Now write $\mathbf{c}_i(t_i, T)$ for the location of i'th sample point, which was obtained at time t_i, and current time is T. Write $\mathbf{c}_i(t_i)$ for $\mathbf{c}_i(t_i, t_i)$. The location can be estimated from the optical flow vector $\mathbf{v}$ by approximate integration as

$$\mathbf{c}_i(t_i, T) = \mathbf{c}_i(t_i, T - \Delta t) + \sum_{t_d=1}^{\Delta t} \mathbf{v}(\mathbf{c}_i(t_i, T - t_d)).$$

We can now extend our simple depth interpolation model to obtain

$$z(\mathbf{x}, T; \mathbf{a}) = \sum_i a_i \left[\beta(\mathbf{x}; \mathbf{c}_i(t_i, T))\omega(T - t_i; \delta t, C)\right] + z_\pi(\mathbf{x}).$$

In principle, $\mathbf{a}$ can be solved by solving a linear system to interpolate the sample points. This linear system is large, and grows with the size of the video. It's hard to solve it directly, so we use a simple and effective approximation. Each depth sample arrives at a frame time, so we can collect the components of $\mathbf{a}$ into groups that apply to particular frames. We solve for each frame independently, then use the resulting $\mathbf{a}$ to reconstruct depth. This approximation is efficient.

Similarly, we extend our advanced depth interpolation model so that $z(\mathbf{x}, T; \mathbf{a}, \mathbf{b}, \mathbf{u})$ is given by

$$\sum_i \left[\begin{pmatrix} a_i \beta(\mathbf{x}; \mathbf{c}_i(t_i, T)) + \\ b_i \psi(\mathbf{x}; \mathbf{c}_i(t_i, T), \mathbf{u}_i) \end{pmatrix} \omega(T - t_i; \delta t, C) \right] + z_\pi(\mathbf{x}).$$

Again, we find it sufficient to solve $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{u}$ frame by frame, and then reconstruct for the whole sequence.

4. Experimental Procedures

Generally, we obtain high resolution depth maps, subsample them, reconstruct using our methods, then compare the reconstruction to the original depth maps. For the Sintel dataset, the high resolution depth maps are ground truth; for others, they are the best available depth maps. We aim to produce reconstructions that are as close as possible to the high resolution depth. In our paper, performance is measured in three ways. First, the recovered depth accuracy, such as RMSE. Second, the complexity of the algorithm, such as run time and parallelization. Third, qualitative and quantitative results in applications such as hand tracking.

4.1. Datasets

We apply our method to four different datasets: the Middlebury stereo dataset [27] [28], the NYU indoor scene dataset [24], the Sintel synthesised dataset [2] and our Gesture Kinect dataset. These datasets cover a wide range of different types of data (near views; distant views; simple depth profiles; complex depth profiles). Because these datasets are very large, we use 30 representative examples in each dataset; details are included in supplementary materials. The Middlebury dataset is the most widely used stereo dataset and is also the dataset most super resolution methods use. This dataset contains high resolution depth with lots of details and the images contain lots of textures, which are relatively challenging for segmentation. The NYU dataset is collected from Kinect with extensive post processing, so the depth is better than the Kinect raw depth. This is a widely used RGBD dataset for data driven RGBD image analysis. The Sintel dataset is a synthesized dataset and contains lots of depth details and high quality images. This dataset uses physical simulation to synthesize complex scenes.

4.2. Subsampling Strategies

We adopt a strategy standard in ray-tracing circles by subdividing the image into grids, then drawing one sample per grid box. Note that we do not smooth depths before sampling, because we do not envisage that future depth sensors will be able to do so. This means that conventional reconstruction techniques applied to the sampled depths alone will alias significantly.

Generally, we describe a particular sampling regime by the size of the box of pixels replaced by a single sample, so that a 64×64 result refers to a case where there are 4096 times as many pixels to reconstruct as there are samples. We explored two protocols for drawing samples: **Fixed Sampling**, where the sample is in the center of the box (Z_{ij}

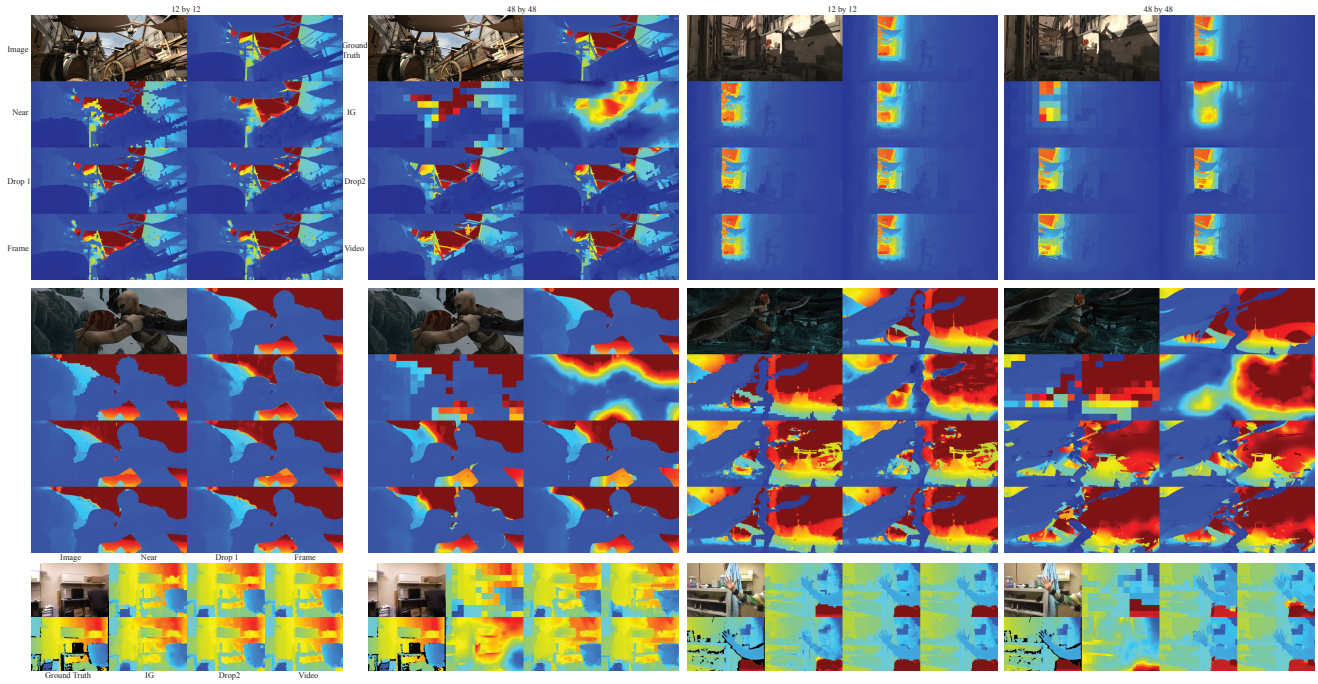

Figure 4. A comparison between [6], per-frame reconstructions, video reconstructions, skip 1 depth frame out of every 2 depth frames and skip 2 depth frames out of every 3 depth frames for examples from six sequences. Note that for drop frames, the frames we show here have no depth values. Our video reconstruction works best and even our drop 2 frames works better than [6], especially when the upscale ratio is big.

for depth at pixel i,j, depth samples are $i = ka$, $j = kb$, $a = 1, 2, 3, b = 1, 2, 3$, k is the integer ratio). Results in the paper are from fixed samples; we have also experimented with **Gaussian Sampling**, where the location of the sample is a draw from a normal distribution with mean the box center and standard deviation $\frac{1}{6}$ box edge length. As one would expect, we find that differences in error statistics are small, slightly favoring Guassian Sampling. Details in supplementary materials.

[bad pixel %, RMSE], smaller is better.

Ratio	Method	Tsukuba		Venus		Teddy		Cones	
	Near	3.53	1.189	0.81	0.408	6.71	1.943	5.44	2.470
	Bicubic	3.84	0.673	0.88	0.290	4.43	2.268	5.98	2.336
4	[32]	2.56	n/a	0.42	n/a	5.95	n/a	4.76	n/a
×	[11]	2.95	**0.450**	0.65	0.179	4.80	1.389	6.54	1.398
4	[18]	**1.73**	0.487	**0.25**	**0.129**	3.54	1.347	5.16	1.383
	[6]	3.03	0.898	0.43	0.201	6.03	0.817	2.98	0.942
	Ours	2.06	0.590	0.29	0.185	**2.03**	**0.593**	**2.56**	**0.938**
	Near	3.56	1.135	1.90	0.546	10.9	2.614	10.4	3.260
	Bicubic	6.67	0.972	2.03	0.427	10.9	2.758	11.9	3.300
8	[32]	6.95	n/a	1.19	n/a	11.50	n/a	11.00	n/a
×	[11]	5.59	**0.713**	1.24	0.249	11.4	1.743	12.3	1.883
8	[18]	3.53	0.753	**0.33**	**0.156**	6.49	1.662	9.22	1.871
	[6]	5.96	1.135	1.98	0.338	12.0	1.376	14.2	1.709
	Ours	**3.52**	0.773	0.47	0.258	**3.82**	**0.863**	**3.65**	**1.041**

Table 2. Comparison with state of art methods on Middlebury dataset. Our method is better than other methods, especially when the image quality is similar to current camera images (eg Teddy; Cones). Images in supplementary material. Bad pixel % is the percentage of bad pixels with respect to all pixels of the ground truth data with error threshold 1.

4.3. Dealing with Noise

The gesture dataset uses raw Kinect depth, so there is noise, mis-alignment and missing values. We deal with depth missing values by using only known depth values. If the segment being smoothed contains only unknown values we mark the segment as having a unique missing value or use the average of the near known neighbors. We control measurement noise by using the advanced smoothing method, which does not interpolate and tends to make the surface smooth in most places. Two approaches help us deal with mis-alignment between the camera and the depth sensor. First, the advanced smoothing method is likely to get rid of a single point that is very different from other points. Second, we identify the largest and smallest ten percent of the depth values within a segment. Any of these points which are also close to the boundary of the segment may represent alignment problems. At problem points, we replace the depth value with a weighted average of the nearest neighbors within the segment, using neighbors that are not themselves problem points.

5. Results

Our method yields state of the art results compared to strong recent methods (Table 1, Table 2, Table 3) in both image depth super resolution and video depth super resolution. Results for images depth super resolution are in Fig-

[log10, RMSE], smaller is better

Ratio	Method	Market		Alley		Ambush		Cave		Office		Gesture	
	Near	0.0531	4.9773	0.0174	2.0154	0.0204	1.0543	0.0319	3.9762	0.0086	0.0365	0.0196	0.0494
12	Bicubic	0.0734	4.2884	0.0182	1.7009	0.0276	0.8958	0.0508	3.3315	0.0092	0.0313	0.0210	0.0436
×	[6]	0.0667	3.7422	0.0215	1.7058	0.0232	0.5401	0.0505	3.4666	0.0131	0.0369	0.0264	0.0356
12	Frame	0.0261	2.5656	0.0131	1.3151	0.0082	0.4580	**0.0188**	**2.4652**	0.0081	0.0290	0.0147	0.0335
	Drop 1	0.0411	2.6793	0.0256	1.5682	0.0216	0.5901	0.0497	4.5012	0.0062	0.0233	0.0133	0.0323
	Drop 2	0.0531	2.8404	0.0310	1.6690	0.0279	0.6516	0.0708	5.7482	0.0066	0.0244	0.0138	0.0329
	Video	**0.0247**	**2.4703**	**0.0101**	**1.2110**	**0.0080**	**0.4511**	0.0200	2.4813	**0.0055**	**0.0208**	**0.0120**	**0.0318**
	Near	0.0899	6.3113	0.0301	2.7225	0.0366	1.3846	0.0626	5.4765	0.0153	0.0508	0.0324	0.0659
24	Bicubic	0.1210	5.4869	0.0294	2.3210	0.0486	1.1637	0.0903	4.6378	0.0157	0.0434	0.0343	0.0549
×	[6]	0.1426	5.1486	0.0437	2.3575	0.0842	1.1855	0.1306	5.4775	0.0255	0.0592	0.0486	0.0671
24	Frame	0.0433	3.2642	0.0227	1.7317	0.0125	0.5612	0.0303	**3.1358**	0.0124	0.0377	0.0219	0.0405
	Drop 1	0.0524	3.0734	0.0307	1.7939	0.0247	0.6407	0.0582	4.8841	0.0093	0.0303	0.0199	0.0369
	Drop 2	0.0660	3.2973	0.0361	1.9139	0.0313	0.7090	0.0793	6.0418	0.0098	0.0318	0.0209	0.0382
	Video	**0.0370**	**2.8828**	**0.0154**	**1.4826**	**0.0116**	**0.5364**	**0.0297**	3.1537	**0.0084**	**0.0281**	**0.0184**	**0.0364**
	Near	0.1396	8.0155	0.0555	3.5887	0.0738	1.9890	0.1197	7.6708	0.0304	0.0781	0.0619	0.0955
48	Bicubic	0.2034	6.8604	0.0547	3.1245	0.0941	1.7141	0.1468	6.6947	0.0293	0.0690	0.0640	0.0846
×	[6]	0.2347	6.2689	0.0781	3.1700	0.1195	1.7024	0.1633	6.8265	0.0356	0.0759	0.0795	0.0944
48	Frame	0.0761	4.3003	0.0322	2.2228	0.0216	0.7437	0.0560	4.2849	0.0165	0.0435	0.0358	0.0534
	Drop 1	0.0753	3.6223	0.0391	2.0803	0.0334	0.8215	0.0816	5.8359	0.0148	0.0411	0.0292	0.0456
	Drop 2	0.0932	4.0598	0.0452	2.1789	0.0405	0.8701	0.1054	7.0715	0.0155	0.0434	0.0309	0.0492
	Video	**0.0582**	**3.3440**	**0.0231**	**1.7372**	**0.0181**	**0.6774**	**0.0480**	**4.1890**	**0.0138**	**0.0387**	**0.0276**	**0.0433**
	Near	0.1588	8.1772	0.0544	3.9099	0.0681	1.9595	0.1040	7.4999	n/a	n/a	n/a	n/a
64	Bicubic	0.2113	7.2739	0.0488	3.2382	0.0965	1.6106	0.1312	6.3807	n/a	n/a	n/a	n/a
×	[6]	0.2206	7.3979	0.0857	3.8713	0.1371	2.1635	0.1791	8.2218	n/a	n/a	n/a	n/a
64	Frame	0.1087	5.1097	0.0418	2.6876	0.0333	0.9675	0.0863	5.6128	n/a	n/a	n/a	n/a
	Drop 1	0.1037	4.4940	0.0466	2.3461	0.0427	0.9905	0.1060	6.9270	n/a	n/a	n/a	n/a
	Drop 2	0.1183	4.7300	0.0543	2.5789	0.0497	1.0212	0.1274	7.9395	n/a	n/a	n/a	n/a
	Video	**0.0847**	**3.9590**	**0.0320**	**2.0638**	**0.0261**	**0.8438**	**0.0721**	**5.2936**	n/a	n/a	n/a	n/a

Table 3. Comparison between nearest neighor, bicubic, [6], our image super resolution, our video super resolution, our skip 1 depth frame out of every 2 depth frames and skip 2 depth frames out of every 3 depth frames results. Our image super resolution performs much better than nearest neighor and [6]; our video super resolution is better still; even our drop depth frames works reasonably well.

ure 3.

5.1. Video Super Resolution

Using the space-time structure of video yields better results than performing super resolution frame by frame. We evaluated our methods on the Sintel dataset (where there is high resolution ground truth depth), and on Kinect sequences. In each case, we used calculated optical flow (rather than ground truth, which Sintel provides). Table 3 shows our per frame super resolution is generally better than all other methods, and that our video super resolution is better than our per frame super resolution. We also works on dropping depth frames, which means at that frame, there are only RGB image and no depth vlaues. Figure 4 shows one of the video frames in different videos.

Looking at the video from Kinect, one can see smoothed noise in reconstructions, resulting from the relatively low sampling rate (depths appear to blink or flash). Note that there is a visible measurement noise in the kinect. Note also that this noise is suppressed, but not removed, by using video rather than per-frame reconstruction.

5.2. Comparisons

Kinect fusion recovers improved accuracy depth maps by fusing multiple depth maps from many different viewpoints [14]. We have no ground truth depths available to do qualitative comparisons, so we show an example comparing our results (with downsampled raw kinect depth as input) with kinect fusion for a view of a keyboard, which contains many fine structures. Figure 6 shows the results and each key on the keyboard occupies about 100 pixels. The input Kinect raw depth is noisy and our results are better than the input raw depth. Our results are also comparable to kinect fusion when the upscale factor is relatively small. When the upscale factor is large, many key segments don't have sample data so our method cannot recover these depths.

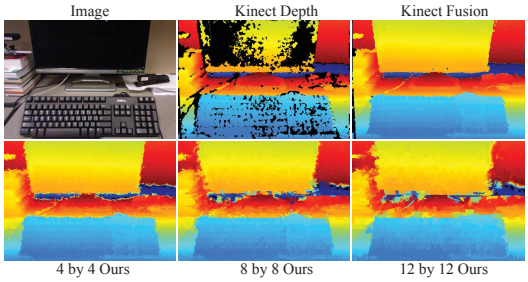

Figure 6. Our depth map is comparable to that obtained by kinect fusion, and rather better than raw depth, especially when the super resolution ratio is small. More information is in the movie.

Super sparse sampling presents particular challenges. Figure 5 demonstrates our method can operate in this regime. Notice that using a z_π obtained from depth transfer methods yields an improved reconstruction in this regime.

5.3. Applications

Depth information is very useful. We demonstrate that our depth super resolution method supports two representative and challenging applications: object insertion and hand tracking from Kinect.

Object insertion methods estimate a depth map for a legacy photograph using depth transfer, then allow users to drag new objects over the estimated scene [17]. Errors in the depth estimate lead to distracting effects in the drag interface. The improvements obtained by our method are sufficient to improve behavior of this interface (see the ex-

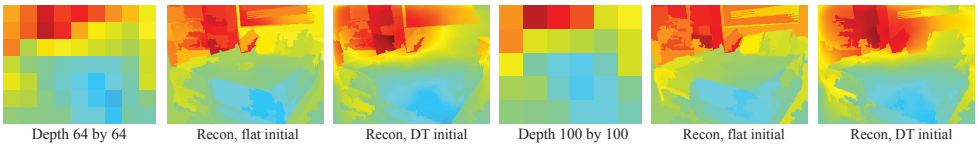

Figure 5. At very sparse depth sampling regimes, the depth map inferred by depth transfer can improve the inferred depth, likely by providing good estimates of low spatial frequency components lost in the sampling. We exploit this information using depth transfer as z_π in our reconstruction method. The figure shows an example from 64×64 and from 100×100 subsampling ("initial" refers to z_π). Note the significant improvements obtained by using depth transfer in this case.

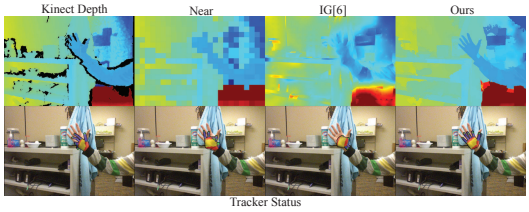

Figure 7. Our reconstruction methods support hand tracking from aggressively subsampled depth maps. The hand tracker works much better on our depth, than Near and [6].

ample in the movie).

Hand tracking is an important application of Kinect which is sensitive to the quality of depth estimates. We evaluated the forth hand tracker [25] [19], (the best currently available depth based hand tracker) on raw kinect depth maps, depth maps subsampled to 24 $\times$24 then reconstructed with nearest neighbors, [6] and our method. Using our depth works as well as using raw kinect depth, while using nearest neighor and [6] depth creates significant difficulties for the tracker (as judged qualitatively by average tracker score in supplementary materials). Figure 7 shows visual results; there are more examples in the movie.

6. Depth from No Samples

Our method works well because the spatial model is effective; depth really does change quickly at segment boundaries, and slowly elsewhere. This suggests applying the method when there are no samples. There is a considerable literature on regressing depth against images (review in [16]). All methods tend to produce very smooth depth maps as a result of the estimation procedures. The best performing method uses deep network features to represent the image, and regresses depth against these features [4]. Because the features are largely invariant to small local shifts of image patches, the regressed depth is smoothed. It is simple to impose a spatial model that is more sensitive to depth boundaries using our method. We subsample the depth map produced by the method of [4], use mean median filter to process depth samples within each image segment for robustness reasons, then upsample using our method. This introduces stronger depth gradients at segment boundaries, and leads to a useful improvement in reconstruction error. Qualitative and quantitative results are in supplementary materials.

7. Discussion and Limitations

Speed: Currently, our advanced method does not run in real time, but the basic method could. Our platform is Windows 8.1 and Matlab 2012b, with 12 GB memory and Intel Core i7. For the biggest images (1300*1100) in dataset, the times are as follows. [6] is about 800s, and [12] is about 30s. For our advanced version, time is about 20s, for our simple version, time is about 5s. The time for each frame in video is similar to single image. The simple version is likely to be real time with good parallel implementation.

Optical flow: Our method extends to upsampling optical flow fields rather well (because flow boundaries tend to appear at image segment boundaries). Detailed results appear in supplementary material. Currently, we have no evidence that a speedup is available from this observation.

Fine details: Our method cannot recover fine or complex surface relief, and will be unhelpful when there is little contrast at object boundaries. Our method tends to work poorly when there are many image segments without a depth sample. Segmentation errors will clearly disrupt our method. We believe that, in general, higher image resolution will have beneficial effects on our results, by producing more detailed segmentations and more accurate optical flow calculations. Finally, errors in sample depth can have serious consequences for our method.

We see a variety of interesting future avenues to explore. A depth camera that makes few depth samples may be able to make those samples more accurately. Finally, our method is adaptive, and it would be interesting to explore procedures that explicitly manage a budget of depth samples to produce the best reconstruction. There is good evidence (the results for skipped frames in Table 3) that one could manage this budget over time (as with a power budget).

Acknowledgements: This material is based upon work supported in part by the National Science Foundation under Grants No. NSF IIS 09-16014 and IIS-1421521; and in part by ONR MURI Award N00014-10-10934.

References

[1] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation, 2011.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *ECCV*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012.

[3] J. Diebel and S. Thrun. An application of markov random fields to range sensing. In *NIPS*, Cambridge, MA, 2005. MIT Press.

[4] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. *arXiv preprint arXiv:1406.2283*, 2014.

[5] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, Sept. 2004.

[6] D. Ferstl, C. Reinbacher, R. Ranftl, M. Ruether, and H. Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *Proceedings International Conference on Computer Vision (ICCV), IEEE*, December 2013.

[7] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Comput. Graph. Appl.*, 22(2):56–65, Mar. 2002.

[8] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph based video segmentation. *IEEE CVPR*, 2010.

[9] S. A. Gudmundsson, H. Aanaes, and R. Larsen. Fusion of stereo vision and time-of-flight imaging for improved 3d estimation. *Int. J. Intell. Syst. Technol. Appl.*, 5(3/4):425–433, Nov. 2008.

[10] Y. Han, J.-Y. Lee, and I. S. Kweon. High quality shape from a single rgb-d image under uncalibrated natural illumination. In *ICCV*, 2013.

[11] S. Hawe, M. Kleinsteuber, and K. Diepold. Analysis operator learning and its application to image reconstruction. *Image Processing, IEEE Transactions on*, 22(6):2138–2150, 2013.

[12] K. He, J. Sun, and X. Tang. Guided image filtering. In *Proceedings of the 11th European Conference on Computer Vision: Part I*, ECCV'10, pages 1–14, Berlin, Heidelberg, 2010. Springer-Verlag.

[13] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. 2009.

[14] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM.

[15] B. Jones, R. Sodhi, D. Forsyth, B. Bailey, and G. Maciocci. Around device interaction for multiscale navigation. In *MobileHCI*, 2012.

[16] K. Karsch, C. Liu, and S. B. Kang. Depthtransfer: Depth extraction from video using non-parametric sampling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2014.

[17] K. Karsch, K. Sunkavalli, S. Hadap, N. Carr, H. Jin, R. Fonte, M. Sittig, and D. Forsyth. Automatic scene inference for 3d object compositing. *ACM Trans. Graph.*, 33(3), June 2014.

[18] M. Kiechle, S. Hawe, and M. Kleinsteuber. A joint intensity and depth co-sparse analysis model for depth map super-resolution. *Computer Vision, IEEE International Conference on*, 0:1545–1552, 2013.

[19] N. Kyriazis, I. Oikonomidis, and A. Argyros. A gpu-powered computational framework for efficient 3d model-based vision. Technical Report TR420, ICS-FORTH, July 2011.

[20] J. Li, Z. Lu, G. Zeng, R. Gan, and H. Zha. Similarity-aware patchwork assembly for depth image super-resolution. June 2014.

[21] J. Lu, D. Min, R. S. Pahwa, and M. N. Do. A revisit to MRF-based depth map super-resolution and enhancement. In *ICASSP'11*, pages 985–988, 2011.

[22] O. Mac Aodha, N. D. Campbell, A. Nair, and G. J. Brostow. Patch based synthesis for single depth image super-resolution. In *ECCV (3)*, pages 71–84, 2012.

[23] M. Mahmoudi and G. Sapiro. Sparse representations for range data restoration. *IEEE Transactions on Image Processing*, 21(5):2909–2915, 2012.

[24] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[25] I. Oikonomidis, N. Kyriazis, and A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC 2011*. BMVA, 2011.

[26] J. Park, H. Kim, Y.-W. Tai, M. S. Brown, and I. Kweon. High quality depth map upsampling for 3d-tof cameras. *Computer Vision, IEEE International Conference on*, 0:1623–1630, 2011.

[27] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47(1-3):7–42, Apr. 2002.

[28] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *CVPR*, CVPR'03, pages 195–202, Washington, DC, USA, 2003. IEEE Computer Society.

[29] S. Schuon, C. Theobalt, J. Davis, and S. Thrun. Lidarboost: Depth superresolution for tof 3d shape scanning. In *CVPR'09*, pages 343–350, 2009.

[30] R. Sodhi, B. Jones, D. Forsyth, B. Bailey, and G. Maciocci. Bethere: 3d mobile collaboration with spatial input. In *SIGCHI*, 2013.

[31] I. Tosic and S. Drewes. Learning joint intensity-depth sparse representations. *IEEE Transactions on Image Processing*, 23(5):2122–2132, 2014.

[32] Q. Yang, R. Yang, J. Davis, and D. Nistér. Spatial-depth super resolution for range images. In *CVPR*, 2007.

[33] L.-F. Yu, S.-K. Yeung, Y.-W. Tai, and S. Lin. Shading-based shape refinement of rgb-d images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[34] J. Zhu, L. Wang, R. Yang, and J. Davis. Fusion of time-of-flight depth and stereo for high accuracy depth maps.