HyperDepth: Learning Depth from Structured Light Without Matching

Sean Ryan Fanello* Christoph Rhemann* Vladimir Tankovich
Adarsh Kowdle Sergio Orts Escolano David Kim Shahram Izadi

Microsoft Research

Abstract

Structured light sensors are popular due to their robustness to untextured scenes and multipath. These systems triangulate depth by solving a correspondence problem between each camera and projector pixel. This is often framed as a local stereo matching task, correlating patches of pixels in the observed and reference image. However, this is computationally intensive, leading to reduced depth accuracy and framerate. We contribute an algorithm for solving this correspondence problem efficiently, without compromising depth accuracy. For the first time, this problem is cast as a classification-regression task, which we solve extremely efficiently using an ensemble of cascaded random forests. Our algorithm scales in number of disparities, and each pixel can be processed independently, and in parallel. No matching or even access to the corresponding reference pattern is required at runtime, and regressed labels are directly mapped to depth. Our GPU-based algorithm runs at a 1KHz for 1.3MP input/output images, with disparity error of 0.1 subpixels. We show a prototype high framerate depth camera running at 375Hz, useful for solving tracking-related problems. We demonstrate our algorithmic performance, creating high resolution real-time depth maps that surpass the quality of current state of the art depth technologies, highlighting quantization-free results with reduced holes, edge fattening and other stereo-based depth artifacts.

1. Introduction

Consumer depth cameras have revolutionized many aspects of computer vision. With over 24 million Microsoft Kinects sold alone, structured light sensors are still the most widespread depth camera technology. This ubiquity is both due to their affordability, and well-behaved noise characteristics, particularly compared with time-of-flight cameras that suffer from multipath errors [17]; or passive stereo techniques which can fail in textureless regions [43, 5].

Structured light systems date back many decades; see [39, 16]. Almost all follow a similar principle: A calibrated camera and projector (typically both near infrared-based) are placed at a fixed, known baseline. The structured light pattern helps establish *correspondence* between observed and projected pixels. Depth is derived for each corresponding pixel through triangulation. The process is akin to two camera stereo [43], but with the projector system replacing the second camera, and aiding the correspondence problem.

Broadly, structured light systems fall into two categories: *spatial* or *temporal*. The former uses a *single* spatially varying pattern, e.g. [14, 45], and algorithms akin to stereo matching to correlate a patch of pixels from the observed image to the reference pattern, given epipolar constraints. Conversely, the latter uses a varying pattern over time to encode a unique temporal signature that can be decoded at each observed pixel, directly establishing correspondence.

Temporal techniques are highly efficient computationally, allowing for a simple, fast lookup to map from observed to projected pixels, and estimate depth. However, they require *complex optical systems* e.g. MEMS based projectors and fast sensors, suffer from *motion artifacts* even with higher framerate imagers, and are *range limited* given the precision of the coding scheme. Therefore many consumer depth cameras are based on spatially varying patterns, typically using a cheap diffractive optical element (DOE) to produce a pseudo-random pattern, such as in Kinect.

However, spatial structured light systems carry a fundamental algorithmic challenge: *high computational cost* associated with matching pixels between camera and projector, analogous to stereo matching. This computational barrier has also motivated many *local* stereo methods; see [43, 5]. Whilst progress has been made on efficient stereo methods, especially so called O(1) or constant time methods [5], these often trade accuracy or precision for performance, and even then very high framerates cannot be achieved.

Just a single disparity hypothesis often requires two local patches (in left and right images) to be compared, with many pixel lookups and operations. Spatial structured light algorithms e.g. in Kinect [14, 26], attempt to reduce these comparisons, but even then ∼20 patch comparisons are re-

*Authors equally contributed to this work.

quired per pixel. These are even higher for dense stereo methods. In addition, there are further sequential operations such as region growing, propagation or filtering steps [5]. This explains the fundamental limit on resolution and framerate we see in depth camera technologies today (typically 30-60Hz VGA output).

In this paper we present HyperDepth, a new algorithm that breaks through this computational barrier without trading depth accuracy or precision. Our approach is based on a *learning-based* technique that frames the correspondence problem into a *classification* and *regression* task, instead of stereo matching. This removes the need for matching entirely or any sequential propagation/filtering operations. For each pixel, our approach requires less compute than a single patch comparison in Kinect or related stereo methods.

The algorithm independently classifies each pixel in the observed image, using a label uniquely corresponding to a subpixel position in the associated projector scanline. This is done by only sparsely sampling a 2D patch around the input pixel, and using a specific recognizer per scanline. Absolutely *no* matching or even access to the corresponding reference pattern is required at runtime. Given a calibrated setup, every pixel with an assigned class label can be directly mapped to a subpixel disparity and hence depth.

To train our algorithm, we capture a variety of geometric scenes, and use a high-quality, offline stereo algorithm [7] for ground truth. This allows our recognizers to learn a mapping for a given patch to a (discrete then continuous) class label that is invariant to scene depth or affine transformations due to scene geometry. Using this approach, we demonstrate extremely compelling and robust results, at a working range of 0.5m to 4m, with complex scene geometry and object reflectivity. We demonstrate how our algorithm *learns* to predict depth that even surpasses the ground truth. Our classifiers learn from *local* information, which is critical for generalization to arbitrary scenes, predicting depth of objects and scenes vastly different from the training data.

Our algorithm allows each pixel to be computed independently, allowing parallel implementations. We demonstrate a GPU algorithm that runs at 1KHz on input images of 1.3MP producing output depth maps of the same resolution, with 2^{17} disparity levels. We demonstrate a prototype 375Hz camera system, which can be used for many tracking problems. We also demonstrate our algorithm running live on Kinect (PrimeSense) hardware. Using this setup we produce depth maps that surpass the quality of Kinect V1 and V2, offline stereo matching, and latest sensors from Intel.

1.1. Related Work

Work on structured light dates back over 40 years [46, 35, 4, 3]. At a high level these systems are categorized as *temporal* or *spatial* [40, 39, 16].

Temporal techniques require multiple captures of the scene with a varying dynamic pattern (also called multishot [16]). This projected pattern encodes a temporal signal that is uniquely decoded at each camera pixel. Examples of patterns include binary [41, 20] gray code, [35], and fringe patterns [19, 53]. These techniques have one clear advantage, they are computationally very efficient, as the correspondence between camera and projector pixel is a biproduct of decoding the signal. Depth estimation simply becomes a decode and lookup operation. However, systems require multiple images, leading to motion artifacts in dynamic scenes. To combat this, fast camera and projector hardware is required, such as demonstrated by the Intel F200 product. However, these components can be costly and fast motions still lead to visible artifacts. Systems are also range limited given temporal encoding precision.

Spatial structured light instead use a single unique (or pseudo unique) 1D [27, 51] or 2D pattern [24, 14, 26, 45] for single shot depth estimation. These techniques appeal as they use simple optical elements, i.e. no dynamic projector and regular framerate cameras, and are more robust to motion artifacts and range limitations. However, they also suffer from a fundamental challenge: the correspondence problem becomes far more challenging. Almost all methods frame this problem as a *local stereo matching* problem. For example, the PrimeSense algorithm inside the Kinect [14, 26], first extracts observed dots, then matches a patch around each dot, with corresponding patches in the reference image using NCC. This leads to $\sim$20 NCC patch matches per pixel, with the minimum NCC score being selected. Then a sequential region growing process creates a dense disparity map. Each of these steps: dot extraction, NCC patch matching, disparity selection, and region growing takes considerable time with many pixel lookup and operations.

There is a large body of work on stereo matching, some of which is relevant for structured light systems. For example, [15] first detects sparse support points in stereo images and performs sparse correspondence search among them before a dense propagation step. Others approximate global optimization methods [18, 12, 30, 6, 47, 48] based on dynamic programming and achieve reasonable frame rates but are restricted to low-resolution images and operate on a strongly quantized depth range (typically at 64 discrete depth values).

Local stereo algorithms are generally faster than their global counterparts, because they identify corresponding pixels only based on the correlation of local image patches. Many correlation functions can be implemented as a filter with a computational complexity independent of the filter size [43]. Recent real-time stereo approaches focus on filters that weight each pixel inside the correlation window based on image edges, e.g. based on bilateral filtering [22, 38, 50, 31] or guided image filtering [37, 10]. These approaches show good computational performance with small disparity levels, but do not scale to high precision estimation.

PatchMatch stereo [7] has been shown to achieve high quality dense depth maps by leveraging slanted support windows and sub-pixel disparities within a PatchMatch framework [2]. This technique has recently been extended to real-time performance by assuming fronto parallel windows and reducing the number of iterations of disparity propagation [36, 54]. [25] bring together concepts from PatchMatch stereo and cost volume filtering within a unified framework, but the reliance on superpixels limits structured light use.

Whilst all this work leads to less computation for stereo matching, these approaches ultimately still require a large number of computations (still fundamentally relying on computing matching costs across a large number disparity levels), with often expensive preprocessing, filtering and propagation steps that can be sequential. This has meant that even GPU, FPGA or ASIC stereo implementations [28, 34, 37, 12, 36, 14] can only operate on limited input/output resolution, at speeds rarely exceeding 30Hz, often with a trade in accuracy.

Other work has looked at combining a pair of cameras with either fixed [33, 23, 54] or dynamic structured light patterns [9, 52]. The former referred to as active stereo, is a technique used in the recent Intel R200 depth camera. The latter is used to extend stereo to the temporal domain (space-time stereo) but suffers from motion artifacts and complex hardware setups, similar to other temporal structured light systems. Both these techniques again rely on local stereo algorithms with computational limitations.

Our work attempts to bring the computational benefits of temporal structured light to more widespread and appealing single-shot spatial structured light systems. To achieve this we take a radical departure from the literature, reformulating this correspondence problem to a classification-regression rather than stereo matching task. As we will show in the remainder of this paper, this learning-based approach brings some extremely compelling computational and accuracy benefits. In essence, our approach allows each pixel in the camera image to be evaluated independently, with a computational effort similar to testing a *single* disparity hypothesis in local stereo methods.

Techniques that employ machine learning for depth estimation have begun to appear. [49] explore deep nets for computing stereo matching costs, but still require multiple disparity hypothesis evaluation using computationally expensive learning architectures. [11, 42] predict depth from a single image, but are expensive and lack the accuracy in general scenes. [13] uses diffuse infrared light to learn a shape from shading mapping from infrared intensity to depth, but this technique fails in general scenes.

2. Learning to Recognize Structured Light

In this section we reformulate the spatial structured light correspondence problem from a machine learning perspective, and show how disparity maps with subpixel accuracy can be predicted extremely efficiently.

2.1. Problem formulation

We use a setup analogous to the Kinect, where an infrared (IR) camera is placed at a fixed baseline to a structured light DOE projector. The IR camera captures images at 1280×1024 resolution. The IR projector generates a pseudo-random dot pattern in the scene that is observed by the IR camera as image I. The pattern projector can be seen as a virtual camera that always observes the same constant image, referred to as the reference pattern R. We assume images I and R to be calibrated and rectified. Therefore, for each pixel $\mathbf{p} = (x, y)$ in I the corresponding pixel $\hat{\mathbf{p}} = (\hat{x}, y)$ in R, that shows the same local dot structure, lies on the same scanline y. The shift along the x coordinate $\hat{x} - x$ is known as disparity d and is inversely proportional to the scene depth Z via $Z = \frac{bf}{d}$, where b is the baseline of the system and f is the focal length.

The local dot structure in a small spatial neighborhood in the reference pattern uniquely identifies each pixel $\hat{\mathbf{p}} = (\hat{x}, \hat{y})$ along a scanline $\hat{y}$. Therefore, we can assign each pixel $\hat{\mathbf{p}}$ along a scanline in R to a unique label $c = \hat{x}$ according to its $\hat{x}$ coordinate. If we are able to recognize the class c for a pixel $\mathbf{p} = (x, y)$ in the observed IR image I, we can simply infer the disparity of $\mathbf{p}$ via the direct mapping $d = c - x$.

Motivated by this observation, we cast the depth estimation problem into a machine learning problem, where given training data, we learn to recognize the class label in the observed image I from the local dot structures and, as a consequence, the depth. Note that in the Kinect reference pattern, the same local structures reappear in different $\hat{x}$ coordinates of different scanlines y. Finding these repetitions is a challenging task itself [29]. To overcome this issue, we simply use a classifier per line, which also provides additional robustness against the distortion of the projected pattern. This allows us to reuse the same class labels $c = \hat{x}$ in different scanlines y. Each of these classifiers has to disambiguate $C = 1280$ classes, equal to the width of the image. The reference pattern is symmetric around the central projector pixel and repeats three times along the X and Y dimension. Using a classifier per scanline circumvents this symmetry problem and in practice, due to distortion of the projector, classes never repeat within a scanline. As future work, we could exploit these repetitions to reduce the total number of class labels and classifiers.

Subpixel Accuracy Class labels need to support subpixel shifts of the pattern to avoid quantization of disparity maps. In order to obtain subpixel accuracy, each class is additionally divided into subclasses, equal to the desired level of subpixel precision. In our case, each step is 0.05 pixels, meaning class i to class $i + 1$ has 20 additional subpixel labels. From now on the labels c can assume continuous

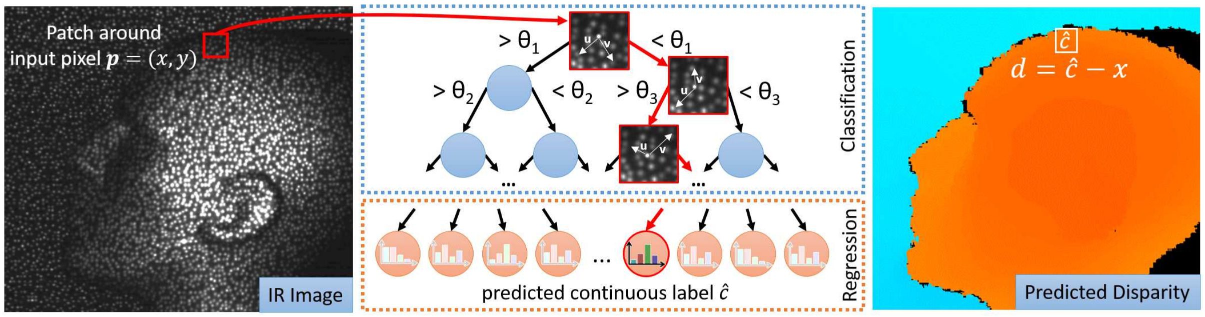


Figure 1. **HyperDepth Algorithm**. Overview of the disparity estimation algorithm using decision trees. For each pixel $\mathbf{p} = (x, y)$ in the input IR image **(left)** we run a Random Forest **(middle)** that predicts the class $\hat{c}$ by sparsely sampling a 2D neighborhood around $\mathbf{p}$. The forest starts with classification and then switches to regression to predict continuous class labels $\hat{c}$ that maintain subpixel accuracy (see text for details). The mapping $d = \hat{c} - x$ gives the actual disparity d **(right)**.

values. It is natural to recast this second step into a regression problem: we first predict the class, then a regression function will produce the final result.

2.2. HyperDepth Algorithm

The core part of our algorithm is a recognizer that predicts continuous class labels independently for each pixel in the image I. Pixels on the same scanline will share the same recognizer. In this work we resorted to an ensemble of random forests per scanline, which have shown great performances in pixel-wise classification using very simple sparse features [44, 13]. Decision trees [8] can be naturally used to train a mixed objective function: we start with a classification objective and then we switch to a regression one to obtain subpixel accuracy.

Given an input pixel $\mathbf{p} = (x, y)$ and the infrared image I, a random forest infers a probability distribution $p(c|\mathbf{p}, I)$ over the C classes. The forest learns to map the pixel into one of the classes looking at its spatial context.

Node Features: Each split node contains a set of learned parameters $\delta = (\mathbf{u}, \mathbf{v}, \theta)$, where $(\mathbf{u}, \mathbf{v})$ are 2D pixel offsets and θ represents a threshold value. The split function f is evaluated at pixel $\mathbf{x}$ as

$$f(\mathbf{p}; \theta) \quad = \quad \begin{cases} \mathrm{L} & \text{if } I(\mathbf{p}+\mathbf{u}) - I(\mathbf{p}+\mathbf{v}) < \theta \\ \mathrm{R} & \text{otherwise} \end{cases} \tag{1}$$

where I is the input IR image. This kind of pixel difference test is commonly used with decision forest classifiers due to its efficiency and discriminative power. The features are also invariant to illumination variations, which helps to generalize across different ambient light levels. The relative offsets $\mathbf{u}$ and $\mathbf{v}$ are sampled within a maximum patch size of 32×32[1], which uniquely identifies all the patterns in the scanline.

[1]Note that the tree will learn automatically to select the best offsets. The maximum window size is selected to minimize the number of bits needed to store a tree node at runtime. Currently we encode each node in 32 bits, 20 bits for two offsets $\mathbf{u}$ and $\mathbf{v}$ and 12 for the threshold θ.

Note the forest predictions are extremely efficient as only a small set of these simple feature tests are performed for each pixel. Furthermore, each pixels (and associated trees) can be processed in parallel.

Training We assume ground truth data for the class labels are available, in the next subsection we describe in details the acquisition procedure. For each scanline of the image we train multiple trees independently on a subset S of the training data. For the first few levels of the tree we consider classes as integer values, solving the classification problem. For our application, set S contains training examples $(\mathbf{p}, c)$ where $\mathbf{p}$ identifies a pixel within a particular training image and c is the pixel's ground truth label. Starting at the root, a set of candidate split function parameters δ are proposed at random. For each candidate, S is partitioned into left $S_{\mathrm{L}}(\delta)$ and right $S_{\mathrm{R}}(\delta)$ child sets, according to Eq. 1. The objective function

$$Q(\delta) = E(S) - \sum_{d \in \{\mathrm{L},\mathrm{R}\}} \frac{|S_d(\delta)|}{|S|} E(S_d(\delta)) \,. \tag{2}$$

is evaluated given each of these partitions, and the candidate δ that maximizes the objective is chosen. The entropy $E(S)$ is the Shannon entropy of the (discrete) empirical distribution $p(c|S)$ of the class labels c in S:

$$E(S) \quad = \quad -\sum_{c=1}^{C} p(c|S) \log p(c|S), \text{with} \tag{3}$$

$$p(c|S) \quad = \quad \frac{1}{|S|} \sum_{(\cdot,\cdot,c') \in S} [c = c'] \,. \tag{4}$$

Training then continues greedily down the tree, recursively partitioning the original set of training pixels into successively smaller subsets. Training stops when a node reaches a maximum depth, contains too few examples, or has too low entropy or differential entropy.

After we learned integer disparities, we continue the training with a regression function for the last 6 levels of the trees in order to obtain subpixel accuracy. In particular we maximize Eq. 2, where the entropy is generalized to handle continuous values. We tested other regression objective functions, based on the variance computed from the samples $(\mathbf{p}, c) \in S$, however they led to considerably worse results. We also tried a direct regression approach, but given the complexity of the problem we noticed substantial overfitting.

2.3. Training Data Generation

In the previous section we assumed the availability of class labels for training. Recall that disparity $d = c - x$, where x is the horizontal position of the pixel in the image and $c = \hat{x}$ the class label. Hence, given a pixel $p = (x, y)$ and disparity d, we can generate a class label via $c = d + x$. To compute disparities for training, we have to recover the reference IR pattern R, the relative pose of R with respect to I and then rectify both R and I. Finally, an accurate stereo matching algorithm computes a disparity map on the rectified images that are used for training. Note that when using the Kinect sensor with our approach, we could directly use the depth maps from the Kinect to generate training labels. However, we found that the depth maps from Kinect are affected by a high level of quantization. Moreover this procedure would not generalize to other structured light systems where Kinect output is not available.

To infer the reference pattern, we leverage the calibration procedure proposed by McIlroy *et al.* [29]. In contrast to this prior work that uses a static camera and a moving IR projector, in our setup we have a rigid assembly of the camera and projector, which we exploit in our calibration process. We first calibrate the intrinsic parameters of the IR camera and we capture images of a flat surface by moving the camera-projector assembly to multiple positions. Similar to [29], we then solve a non-linear optimization via Levenberg-Marquardt to obtain the pattern projected by the IR projector, and the extrinsics of the projector with respect to the camera. We assume that the projector is a virtual camera that is located at a known initial displacement from the camera (the approximate physical location of the projector with respect to the camera), which is used to initialize the solver. In addition to the extrinsics, we recover the reference pattern as seen at the image plane resulting in a simple camera model for the projector with zero distortion. The recovered reference pattern is shown in the supplementary material. Note that similar to [29] we recover the distorted pattern as seen at the image plane. Given this reference pattern and the calibration parameters of camera-projector setup we perform a standard bilinear stereo rectification to recover the rectified reference pattern and the rectified IR image.

Once the rectified reference pattern and the current rectified IR image are available, we can generate disparity maps using an accurate, but offline, stereo matching method. For our training data generation we use PatchMatch stereo [7], which is a state of the art local stereo matching approach that in comparison to other methods gives excellent reconstruction quality for slanted/curved surfaces and estimates sub-pixel disparities without quantization. Note that a local method performs very well for our scenario since there are virtually no untextured regions in the input images. In the future, more sophisticated global stereo methods could be employed to further raise the quality of the training data.

2.4. Runtime Algorithm

At runtime the inputs of the algorithm are the IR image I and H random forest classifiers, one per scanline, where each random forest comprises of F trees. Given a pixel $\mathbf{p} = (x, y)$, the output of a single tree f will be:

$$\hat{c}_f = \arg\max_{c} \; p_{(y,f)}(c|\mathbf{p}, \mathbf{I}), \qquad (5)$$

where $p_{(y,f)}(\cdot)$ denotes the probability of the floating point class c computed with the f-th tree for the scanline y. The aggregation strategy we used among different trees is based on their agreement on the current class $\hat{c}_f$. Let $\hat{c}_1$ and $\hat{c}_2$ the output of 2 different trees. If $|\hat{c}_1 - \hat{c}_2| < 0.2$ we aggregate the prediction $\hat{c} = \frac{p_1\hat{c}_1 + p_2\hat{c}_2}{p_1 + p_2}$ as well as the expected probabilities $\hat{p} = p_1 + p_2$. The disparity $d = \hat{c} - x$ is then assigned to the pixel $\mathbf{p}$ based on the highest score $\hat{p}$.

Invalidation Criteria We invalidate pixels with inaccurately predicted disparities by using the posterior $p(c|\mathbf{p}, \mathbf{I})$ as indication of the confidence for the prediction. In particular, we use the following invalidation criteria:

- **Signal Check**. We invalidate pixels that traverse the random forest and do not observe a sufficient amount of signal $\tau = 500$ in the IR image I.
- **Probability Check**. If the probability $\hat{p}$ of the winning class $\hat{c}$ is smaller than 0.6 we invalidate the pixel.
- **Winners Check**. We make sure that the top 2 predictions are consistent, i.e. they lie in a very close disparity range: we invalidate if $|\hat{c}_1 - \hat{c}_2| > 1$.
- **Disparity Check**. If the predicted label is wrong the disparity $d = \hat{c} - x$ could belong to a non valid range. We invalidate every $d < 0$ and every $d > 422$, which is the maximum number of possible disparities.

For our model configuration, with 4 trees and 12 levels, only 48 pixel differences are processed to compute depth per pixel. The running time does not depend on number of disparities or patch size and it is fully parallel for each pixel.

3. Experiments

We now systematically evaluate our proposed algorithm. It is important to highlight that all results shown were computed on completely new scenes that are not part of the

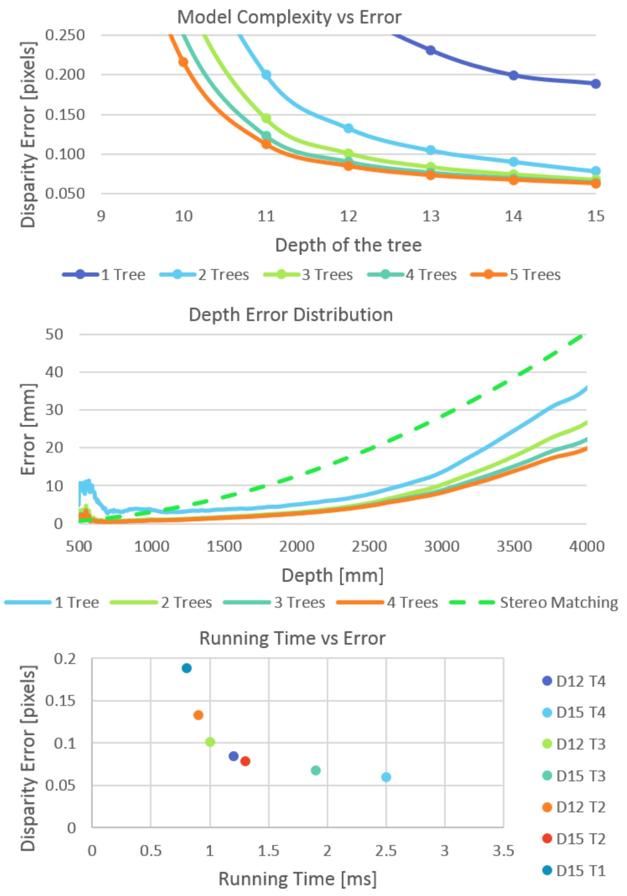


Figure 2. **Quantitative Experiments**. (**top**) Our disparity error to ground truth (PatchMatch stereo) as a function of number of trees and their depth. (**middle**) Our depth error to ground truth (PatchMatch stereo) for different numbers of trees. (**bottom**) Run time of HyperDepth on 1.3 Megapixel images for different tree depths ($D12$, $D15$) and different tree numbers per scanline ($T1$ to $T4$). The most accurate configuration with 4 trees and depth 15 takes only 2.5msec per image.

training data. Unless noted otherwise, we use the camera-projector hardware of the Microsoft Kinect and access the raw IR images using OpenNI.

3.1. HyperDepth Parameters

We first show how we set the HyperDepth parameters and analyze the computational performance. In order to train the algorithm we acquired 10000 frames generating training labels using PatchMatch stereo as described in Sec 2.3. We trained up to 5 trees per line with 15 levels per tree. Training 1 tree per line takes 1 day on a single GPU implementation on a NVIDIA Titan X.

We test the performance of our method on 1000 images that were acquired completely independently from our training data. Fig. 2 evaluates different configurations of our models on the test data. The top graph in Fig. 2 shows the absolute disparity difference between PatchMatch and our

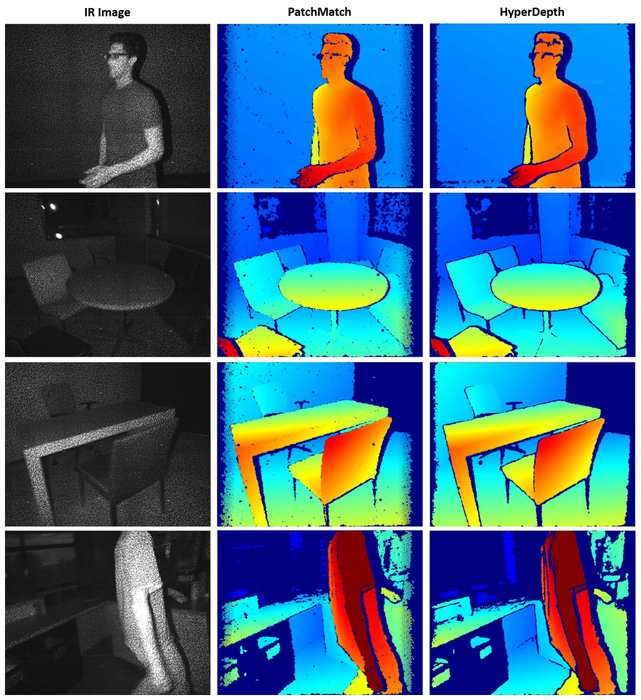


Figure 3. **Qualitative Comparisons**. We compare disparity maps generated on the test data with PatchMatch and HyperDepth. Note how we generalize even in regions where PatchMatch is invalidated.

model with respect to the number of levels per tree. We can see that after level 14 little improvement is gained since the disparity error goes down by only 0.005 pixels on the average. Similarly, more than 4 trees show little improvement, only 0.001 in disparity error.

To see how this disparity error translates to depth error, Fig. 2, middle graph, shows a similar analysis in the depth domain. Notice how the method exhibits very low error ($< 1cm$) up to 3 meters. As a baseline, we also plot the theoretical error for a high quality stereo system, with disparity precision of 0.25 pixels (note: this precision is below the Middlebury subpixel benchmark of 0.5 pixels). We show that our error would be 3 times lower than this baseline. This also shows that our algorithm is able to model the disparity prediction within the precision of our training data (PatchMatch stereo) and the quality of our results is currently limited by the accuracy of the training data. Fig. 2, bottom, reports the running time on 1.3 megapixel images using a NVIDIA Titan X GPU. The top accurate configuration with 4 trees and 15 levels has a disparity error of 0.064 pixels with a running time of $2.5msec$. To reach 1KHz 3 trees with 12 levels are used with an disparity error of only 0.1 pixels. In Fig. 3 we show the quality of the disparity maps generated on the test data with PatchMatch and HyperDepth. Our approach computes correct disparities in image regions where PatchMatch shows holes or fattened edges. This is impressive given the fact that our method was trained with results from PatchMatch.

3.2. Error Analysis

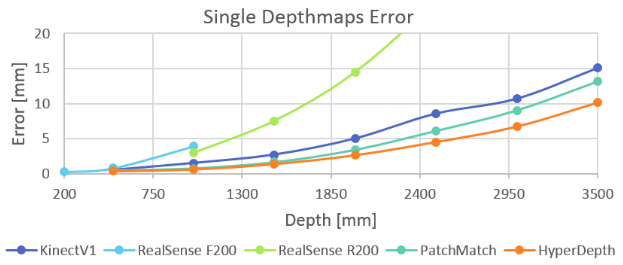
To quantitatively analyze the error of single depthmaps we use a setup similar to [32]: we place a camera in front of a (large) white plane at multiple (known) distances $20, 50, 100, 150, 200, 250, 300, 350$cm. For each distance we record multiple frames and compute the error with respect to the (known) plane equation. We depict the results in Fig. 4 and Fig. 5. We compare HyperDepth with Microsoft KinectV1 (PrimeSense), Intel RealSense F200, Intel RealSense R200 and PatchMatch stereo.

As shown our depth maps contain less error. KinectV1 depth maps suffer from heavy quantization. The F200 contains higher error within the working range of our sensor $> 50cm$, but works at a closer distance up to 20cm. Note this sensor uses temporal structured light, and clearly exhibits motion artifacts, and limited working range (from 20cm to 100cm), with a large performance degradation after 50cm. The R200 is an active stereo camera, and this exhibits extremely high error. Whilst the underlying stereo algorithm is unpublished, it clearly demonstrates the trade-off in accuracy that needs to be made to achieve real-time performance. In this experiment we also outperform the accuracy of PatchMatch: thanks to the ensemble of multiple trees per line our depthmaps we are more robust to noise.

Qualitative comparisons are shown in Fig. 5. We also analyzed the noise characteristic of the algorithm computing the standard deviation (jitter) of the depthmaps over multiple frames. Results show (Fig. 4, bottom) that our method exhibits noise level very similar to the KinectV1, which is expected. Again, the RealSense cameras poorly performed with respect to HyperDepth, KinectV1 and PatchMatch. The latter seems to have higher noise at the end of the range. We further investigate the level of quantization in KinectV1 by designing a qualitative experiment where we placed some objects at 2.5m distance from the camera and we compute the depth maps with both our method and KinectV1. We show the point clouds in Fig. 6: notice how KinectV1 depth maps are heavily quantized, whereas our method produces smooth and quantization free disparities. This is the main advantage of the regression approach, which does not explicitly test subpixel disparities but automatically recovers the output disparity with high precision.

3.3. 3D Scanning Results

We evaluated the precision of the algorithm for object scanning. We generated groundtruth 3D models for multiple objects with different shape, texture and material. The groundtruth is generated via ATOS, an industrial 3D scanning technology [1]. The precision of the ATOS scanner is up to 0.001mm. We then generated 360° 3D models using our method and multiple state of the art depth acquisition technologies: KinectV1, KinectV2 (Time of Flight), PatchMatch [7], Intel RealSense F200 and RealSense R200. To



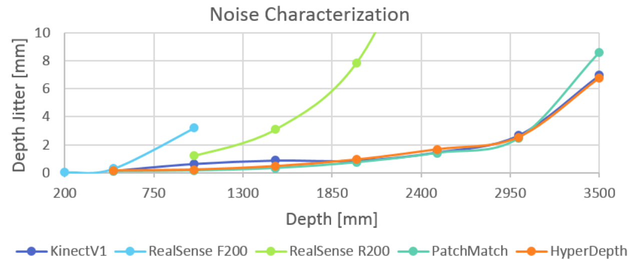


Figure 4. **Error and Noise Analysis**. We plot the depth error of HyperDepth and baseline technologies for a planar target at distances between 20cm and 350cm. The average error of single depth maps is shown on the **top**, whereas the variance within multiple depth maps is shown in the **bottom** figure. Our method exhibits lower error than all baselines.

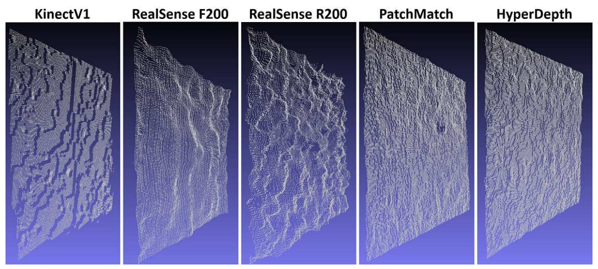


Figure 5. **Plane Fitting Comparison**. We visualize 3D point clouds of a planar target at 1m distance. We compare our results against baseline technologies. Notice the quantization artifacts in KinectV1 and the high noise in the RealSense cameras. Our method and PatchMatch produce smoothest results.

this end, we placed each object on a turntable and captured hundreds of depth maps from all viewpoints from a distance of 50cm (an exception was Intel RealSense R200 where we used the minimum supported distance of 65cm). We then feed the depth maps into KinectFusion [21] to obtain the 3D mesh. We used the same KinectFusion parameters for generating results for all methods. We then carefully aligned each generated mesh with the groundtruth scans and computed the Hausdorff distance to measure the error between the two meshes. In Fig. 7 we report the reconstructed objects and their Root Mean Square Error (RMSE) from the groundtruth. Our HyperDepth consistently outperforms KinectV1 on all the objects, especially areas with high level of details are better reconstructed by our method. This is mainly due to the absence of quantization and the ability to produce higher resolution depth maps. KinectV2 is sensitive to multipath effects, causing errors in those areas where multiple reflec-

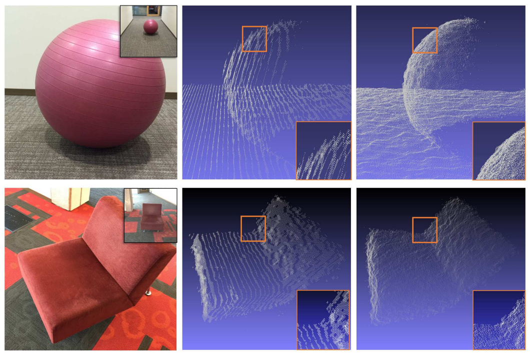
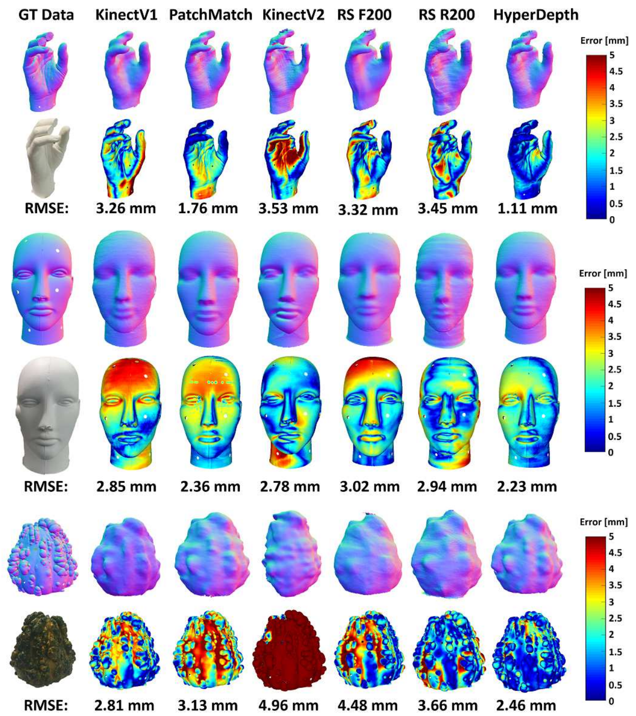


Figure 6. **Quantization Experiment**. We show point clouds generated with KinectV1 (**middle**) and our HyperDepth algorithm (**right**). Notice the heavy quantization in the KinectV1 results, whereas our method infers precise depth.


GT Data
KinectV1
PatchMatch
KinectV2
RS F200
RS R200
HyperDepth
Error [mm]
RMSE: 3.26 mm 1.76 mm 3.53 mm 3.32 mm 3.45 mm 1.11 mm
RMSE: 2.85 mm 2.36 mm 2.78 mm 3.02 mm 2.94 mm 2.23 mm
RMSE: 2.81 mm 3.13 mm 4.96 mm 4.48 mm 3.66 mm 2.46 mm
Figure 7. **3D Scanning Results**. Quantitative comparisons between our method and state of the art depth technologies.

tions occur. As a result, objects are substantially deformed. Our method provides results on par with, and superior to PatchMatch, but at a fraction of the compute. Note we use PatchMatch for training data, and this shows that HyperDepth could be further improved given improvements in training data. Both RealSense sensors failed in capturing most of the details, due to the high noise in the depth maps.

3.4. High-Speed Camera Setup

Our algorithm can be used to create extremely high framerate depth cameras useful for solving tracking related problems. We built a prototype sensor (see Fig. 8 bottom right) capable of generating depth maps at 375Hz. We combine the

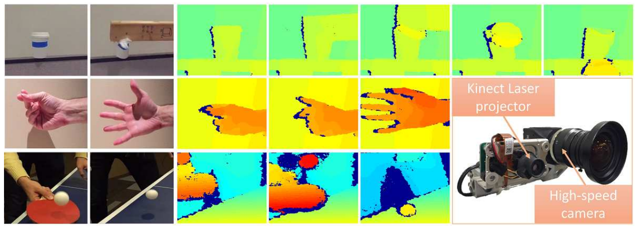


Figure 8. **High Speed Camera Results**. HyperDepth results recorded at 375Hz. (**top**) smashing a paper cup, (**middle**) fast moving hand, (**bottom**) playing ping-pong.

Kinect IR projector and a USB3 Lumenera Lt425 camera with an IR bandpass filter. This camera reaches 375Hz with a 640×480 central crop of the original 4MP image. Notice that in order to operate at this framerate we use an exposure time of 2.5msec, meaning the SNR of the IR images is lower than in Kinect, making the depth estimation more challenging. We calibrated the system and generated training data for our method following the procedure described in Sec. 2.3. We tested this configuration in different sequences to prove the feasibility of high speed depth maps. In particular we show three sequences: a high speed moving hand, capturing a ping-pong ball hitting a racket, and a cup being smashed with a wooden stick. We show qualitative results on Fig. 8. HyperDepth is able to retrieve smooth disparities even in this challenging configuration.

4. Conclusion

We have reframed the correspondence problem for spatial structured light as a learning-based classification-regression task, instead of a stereo matching task. Our novel formulation uses an ensemble of random forests, one per scan line, to efficiently solve this problem, in a pixel independent manner with minimal operations. Our algorithm is independent of matching window size or disparity levels. We have demonstrated a parallel GPU implementation that infers depth for each pixel independently at framerates over 1KHz, with 2^{17} disparity levels, and no sequential propagation step. Finally we have demonstrated, high quality and high resolution, quantization free, depth maps produced by our method, with quality superior to state of the art methods for both single frame prediction, and fused 3D models. Our method can be employed in many new scenarios where high speed and high resolution depth is needed such as hand tracking and 3D scanning applications.

Acknowledgment. We thank Cristian Canton Ferrer for contributing to the calibration process.

References

[1] Atos - industrial 3d scanning technology. http://www.gom.com/metrology-systems/3d-scanner.html. 7

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. PatchMatch: A randomized correspondence algorithm for

structural image editing. *ACM SIGGRAPH and Transaction On Graphics*, 2009. 3

[3] J. Batlle, E. Mouaddib, and J. Salvi. Recent progress in coded structured light as a technique to solve the correspondence problem: a survey. *Pattern recognition*, 31(7):963–982, 1998. 2

[4] P. J. Besl. Active, optical range imaging sensors. *Machine vision and applications*, 1(2):127–152, 1988. 2

[5] M. Bleyer and C. Breiteneder. Stereo matchingstate-of-the-art and research challenges. In *Advanced Topics in Computer Vision*, pages 143–179. Springer, 2013. 1, 2

[6] M. Bleyer and M. Gelautz. Simple but effective tree structures for dynamic programming-based stereo matching. In *VISAPP (2)*, pages 415–422. Citeseer, 2008. 2

[7] M. Bleyer, C. Rhemann, and C. Rother. PatchMatch Stereo - Stereo Matching with Slanted Support Windows. In *BMVC*, 2011. 2, 3, 5, 7

[8] A. Criminisi and J. Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. Springer, 2013. 4

[9] J. Davis, R. Ramamoorthi, and S. Rusinkiewicz. Spacetime stereo: A unifying framework for depth from triangulation. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–359. IEEE, 2003. 3

[10] L. De-Maeztu, S. Mattoccia, A. Villanueva, and R. Cabeza. Linear stereo matching. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1708–1715. IEEE, 2011. 2

[11] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 3

[12] I. Ernst and H. Hirschmüller. Mutual information based semi-global stereo matching on the gpu. In *Advances in Visual Computing*, pages 228–239. Springer, 2008. 2, 3

[13] S. R. Fanello, C. Keskin, S. Izadi, P. Kohli, D. Kim, D. Sweeney, A. Criminisi, J. Shotton, S. Kang, and T. Paek. Learning to be a depth camera for close-range human capture and interaction. *ACM SIGGRAPH and Transaction On Graphics*, 2014. 3, 4

[14] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli. Depth mapping using projected patterns, Apr. 3 2012. US Patent 8,150,142. 1, 2, 3

[15] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *ACCV*, 2010. 2

[16] J. Geng. Structured-light 3d surface imaging: a tutorial. *Advances in Optics and Photonics*, 3(2):128–160, 2011. 1, 2

[17] M. Hansard, S. Lee, O. Choi, and R. P. Horaud. *Time-of-flight cameras: principles, methods and applications*. Springer Science & Business Media, 2012. 1

[18] H. Hirschmüller. Stereo processing by semiglobal matching and mutual information. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):328–341, 2008. 2

[19] E. Horn and N. Kiryati. Toward optimal structured light patterns. *Image and Vision Computing*, 17(2):87–97, 1999. 2

[20] I. Ishii, K. Yamamoto, T. Tsuji, et al. High-speed 3d image acquisition using coded structured light projection. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 925–930. IEEE, 2007. 2

[21] 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 *ACM UIST*, 2011. 7

[22] M. Ju and H. Kang. Constant time stereo matching. pages 13–17, 2009. 2

[23] K. Konolige. Projected texture stereo. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 148–155. IEEE, 2010. 3

[24] J. J. Le Moigne and A. M. Waxman. Structured light patterns for robot mobility. *Robotics and Automation, IEEE Journal of*, 4(5):541–548, 1988. 2

[25] J. Lu, H. Yang, D. Min, and M. Do. Patch match filter: Efficient edge-aware filtering meets randomized search for fast correspondence field estimation. In *CVPR*, 2013. 3

[26] M. Martinez and R. Stiefelhagen. Kinect unleashed: getting control over high resolution depth maps. In *Proceedings of the 13. IAPR International Conference on Machine Vision Applications, MVA*, pages 247–250, 2013. 1, 2

[27] M. Maruyama and S. Abe. Range sensing by projecting multiple slits with random cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(6):647–651, 1993. 2

[28] S. Mattoccia. Stereo vision algorithms for fpgas. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 636–641. IEEE, 2013. 3

[29] P. McIlroy, S. Izadi, and A. Fitzgibbon. Kinectrack: 3d pose estimation using a projected dense dot pattern. *IEEE Trans. Vis. Comput. Graph.*, 20(6):839–851, 2014. 3, 5

[30] M. Michael, J. Salmen, J. Stallkamp, and M. Schlipsing. Real-time stereo vision: Optimizing semi-global matching. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 1197–1202. IEEE, 2013. 2

[31] D. Min, J. Lu, and M. N. Do. A revisit to cost aggregation in stereo matching: How far can we reduce its computational redundancy? In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1567–1574. IEEE, 2011. 2

[32] C. Nguyen, S. Izadi, and D. Lovell. Modeling kinect sensor noise for improved 3d reconstruction and tracking. In *3DIMPVT*, 2012. 7

[33] H. Nishihara. Prism: A practical real-time imaging stereo matcher mit ai memo no. 780. *Cambridge, Mass., USA*, 1984. 3

[34] K. Papadimitriou, S. Thomas, and A. Dollas. An fpga-based real-time system for 3d stereo matching, combining absolute differences and census with aggregation and belief propagation. In *VLSI-SoC: At the Crossroads of Emerging Trends*, pages 168–187. Springer, 2015. 3

[35] J. Posdamer and M. Altschuler. Surface measurement by space-encoded projected beam systems. *Computer graphics and image processing*, 18(1):1–17, 1982. 2

[36] V. Pradeep, C. Rhemann, S. Izad, C. Zach, M. Bleyer, and S. Bathiche. Monofusion: Real-time 3d reconstruction of small scenes with a single web camera. 2013. 3

[37] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. In *CVPR*, pages 3017–3024, 2011. 2, 3

[38] C. Richardt, D. Orr, I. Davies, A. Criminisi, and N. A. Dodgson. Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *Computer Vision–ECCV 2010*, pages 510–523. Springer, 2010. 2

[39] J. Salvi, S. Fernandez, T. Pribanic, and X. Llado. A state of the art in structured light patterns for surface profilometry. *Pattern recognition*, 43(8):2666–2680, 2010. 1, 2

[40] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827–849, 2004. 2

[41] K. Sato. Range imaging system utilizing nematic liquid crystal mask. In *Proc. 1st Int. Conf. Comp. Vision, 1987*, pages 657–661, 1987. 2

[42] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *PAMI*, 31(5):824–840, 2009. 3

[43] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1-3), Apr. 2002. 1, 2

[44] J. Shotton, A. W. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *Proc. CVPR*, 2011. 4

[45] P. Vuylsteke and A. Oosterlinck. Range image acquisition with a single binary-encoded light pattern. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(2):148–164, 1990. 1, 2

[46] P. M. Will and K. S. Pennington. Grid coding: A preprocessing technique for robot and machine vision. *Artificial Intelligence*, 2(3):319–329, 1972. 2

[47] Q. Yang. Stereo matching using tree filtering. *PAMI*, 37(4):834 – 846, 2015. 2

[48] Q. Yang, L. Wang, and N. Ahuja. A constant-space belief propagation algorithm for stereo matching. In *Computer vision and pattern recognition (CVPR), 2010 IEEE Conference on*, pages 1458–1465. IEEE, 2010. 2

[49] J. Žbontar and Y. LeCun. Computing the stereo matching cost with a convolutional neural network. *arXiv preprint arXiv:1409.4326*, 2014. 3

[50] K. Zhang, G. Lafruit, R. Lauwereins, and L. Van Gool. Joint integral histograms and its application in stereo matching. In *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pages 817–820. IEEE, 2010. 2

[51] L. Zhang, B. Curless, and S. M. Seitz. Rapid shape acquisition using color structured light and multi-pass dynamic programming. In *3D Data Processing Visualization and Transmission, 2002. Proceedings. First International Symposium on*, pages 24–36. IEEE, 2002. 2

[52] L. Zhang, B. Curless, and S. M. Seitz. Spacetime stereo: Shape recovery for dynamic scenes. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–367. IEEE, 2003. 3

[53] S. Zhang. Recent progresses on real-time 3d shape measurement using digital fringe projection techniques. *Optics and lasers in engineering*, 48(2):149–158, 2010. 2

[54] M. Zollhöfer, M. Nießner, S. Izadi, C. Rhemann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (TOG)*, 33(4):156, 2014. 3