



DepthCut: improved depth edge estimation using multiple unreliable channels

Paul Guerrero¹ · Holger Winnemöller² · Wilmot Li² · Niloy J. Mitra¹

Published online: 12 May 2018
© The Author(s) 2018

Abstract

In the context of scene understanding, a variety of methods exists to estimate different information channels from mono or stereo images, including disparity, depth, and normals. Although several advances have been reported in the recent years for these tasks, the estimated information is often imprecise particularly near depth discontinuities or creases. Studies have however shown that precisely such *depth edges* carry critical cues for the perception of shape, and play important roles in tasks like depth-based segmentation or foreground selection. Unfortunately, the currently extracted channels often carry conflicting signals, making it difficult for subsequent applications to effectively use them. In this paper, we focus on the problem of obtaining high-precision depth edges (i.e., depth contours and creases) by jointly analyzing such unreliable information channels. We propose DEPTHCUT, a data-driven fusion of the channels using a convolutional neural network trained on a large dataset with known depth. The resulting depth edges can be used for segmentation, decomposing a scene into depth layers with relatively flat depth, or improving the accuracy of the depth estimate near depth edges by constraining its gradients to agree with these edges. Quantitatively, we compare against 18 variants of baselines and demonstrate that our depth edges result in an improved segmentation performance and an improved depth estimate near depth edges compared to data-agnostic channel fusion. Qualitatively, we demonstrate that the depth edges result in superior segmentation and depth orderings. (Code and datasets will be made available.)

Keywords Depth estimation · Monocular · Stereo · Deep learning · Depth layering

1 Introduction

A central task in scene understanding is to segment an input scene into objects and establish a (partial) depth ordering among the detected objects. Since photographs remain the most convenient and ubiquitous option to capture scene infor-

mation, a significant body of research has focused on scene analysis using single (mono) or pairs of (stereo) images. However, extracting high-quality information about scene geometry from such input remains a challenging problem.

Most recent mono and stereo scene estimation techniques attempt to compute disparity, depth or normals from the input image(s). State-of-the-art methods largely take a data-driven approach by training different networks using synthetic (3D rendered) or other ground-truth data. Unfortunately, the resulting estimates still suffer from imperfections, particularly near depth discontinuities. Mono depth estimation is imprecise especially around object boundaries, while stereo depth estimation suffers from disocclusions and depends on the reliability of the stereo matching. Even depth scans (e.g., Kinect scans) have missing or inaccurate depth values near depth discontinuity edges.

In this work, instead of aiming for precise depth estimates, we focus on identifying depth discontinuities, which we refer to as *depth edges*. Studies (see Chapter 10 in [4,12]) have shown that precisely such depth edges carry critical cues for

Electronic supplementary material The online version of this article (<https://doi.org/10.1007/s00371-018-1551-5>) contains supplementary material, which is available to authorized users.

✉ Paul Guerrero
paulauguerrero@gmail.com; paul.guerrero@ucl.ac.uk

Holger Winnemöller
hwinnemo@adobe.com

Wilmot Li
wilmotli@adobe.com

Niloy J. Mitra
n.mitra@ucl.ac.uk

¹ University College London, London, UK

² Adobe Research, Seattle, USA

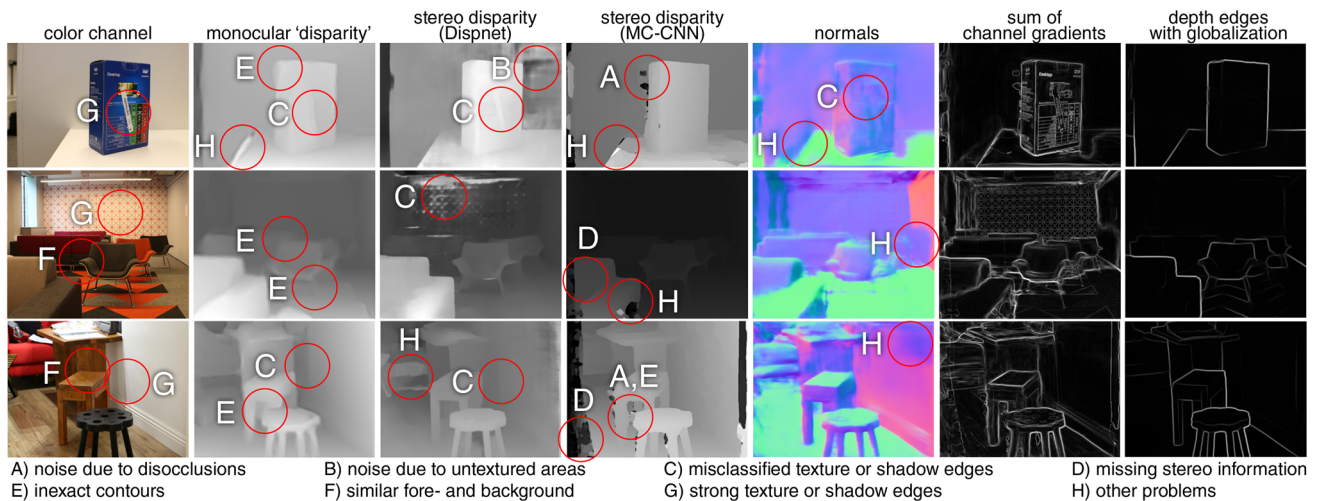


Fig. 1 Our input channels contain various sources of noise and errors: areas of disocclusion, large untextured areas where stereo matching is difficult, and shadow edges that were incorrectly classified during creation of the channels. The color channel may also contain strong texture or shadow edges that have to be filtered out. The gradients of these channels do generally not align well, as shown in the second column from the right. We train DEPTHCUT to learn how to combine color,

disparity, and normal channels to generate a cleaner set of depth edges, shown in the last column after a globalization step. In contrast to the sum of gradients, these depth edges now correspond to the probability of being a true depth contour or crease, giving them a larger intensity range. The optional globalization we show here only retains the most salient edges

the perception of shapes, and play important roles in tasks like depth-based segmentation or foreground selection. Due to the aforementioned artifacts around depth discontinuities, current methods mostly produce poor depth edges, as shown in Fig. 1. Our main insight is that we can obtain better depth edges by fusing together multiple cues, each of which may, in isolation, be unreliable due to misaligned features, errors, and noise. In other words, in contrast to absolute depth, depth edges often correlate with edges in other channels, allowing information from such channels to improve global estimation of depth edge locations.

We propose a data-driven fusion of the channels using DEPTHCUT, a convolutional neural network (CNN) trained on a large dataset with known depth. Starting from either mono or stereo images, we investigate fusing three different channels: color, estimated disparity, and estimated normals (see Fig. 1). The color channel carries good edge information wherever there are color differences. However, it fails to differentiate between depth and texture edges or to detect depth edges if adjacent foreground and background colors are similar. Depth disparity, estimated from stereo or mono inputs, tends to be more reliable in regions away from depth edges and hence can be used to identify texture edges picked up from the color channel. It is, however, unreliable near depth edges as it suffers from disocclusion ambiguity. Normals, estimated from left image (for stereo input) or mono input, can help identify large changes in surface orientation, but they can be polluted by misclassified textures, etc.

Combining these channels is challenging, since different locations on the image plane require different combinations, depending on their context. Additionally, it is hard to formulate explicit rules how to combine channels. We designed DEPTHCUT to combine these unreliable channels to obtain robust depth edges. The network fuses multiple depth cues in a context-sensitive manner by learning what channels to rely on in different parts of the scene. For example, in Fig. 2-top, DEPTHCUT correctly obtains depth segment layers separating the front statues from the background ones even though they have very similar color profiles, while in Fig. 2-bottom, DEPTHCUT correctly segments the book from the clutter of similarly colored papers. In both examples, the network produces good results even though the individual channels are noisy due to color similarity, texture and shading ambiguity, and poor disparity estimates around object boundaries.

We use the extracted depth edges for segmentation, decomposing a scene into depth layers with relatively flat depth, or improving the accuracy of the depth estimate near depth edges by constraining its gradients to agree with the estimated (depth) edges. We extensively evaluate the proposed estimation framework, both qualitatively and quantitatively, and report consistent improvement over state-of-the-art alternatives. Qualitatively, our results demonstrate clear improvements in interactive depth-based object selection tasks on various challenging images (without available ground truth for evaluation). We also show how DEPTHCUT can produce qualitatively better disparity estimates near depth edges. From a quantitative perspective, our depth edges

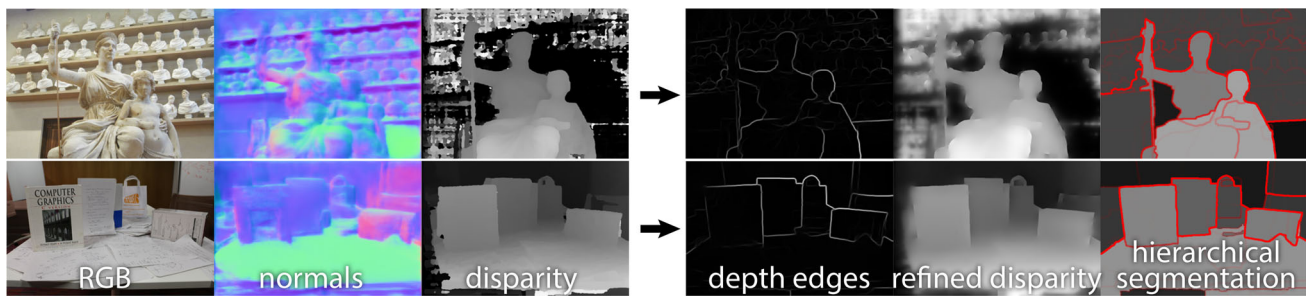


Fig. 2 We present DEPTH CUT, a method to estimate *depth edges* with improved accuracy from unreliable input channels, namely: color images, normal estimates, and disparity estimates. Starting from a single image or pair of images, our method produces depth edges consisting of depth contours and creases, and separates regions of smoothly varying

depth. Complementary information from the unreliable input channels is fused using a neural network trained on a dataset with known depth. The resulting depth edges can be used to refine a disparity estimate or to infer a hierarchical image segmentation

lead to large improvements in segmentation performance compared to 18 variants of baselines that either use a single channel or perform data-agnostic channel fusion. On a manually captured and segmented test dataset of natural images, our DEPTH CUT-based method outperforms all baseline variants by at least 9%.

2 Related work

Shape analysis In the context of scene understanding, a large body of work focuses on estimating attributes for indoor scenes by computing high-level object features and analyzing inter-object relations (see [26] for a survey). More recently, with renewed interest in deep neural networks, researchers have explored data-driven approaches for various shape and scene analysis tasks (cf., [38]). While there are too many efforts to list, representative examples include normal estimation [5,10], object detection [36], semantic segmentation [8,13], localization [32], pose estimation [3,37], and scene recognition using combined depth and image features from RGBD input [41].

At a coarse level, these data-driven approaches produce impressive results, but are often noisy near discontinuities or fine detail. Moreover, the various methods tend to produce different types of errors in regions of ambiguity. Since each network is trained independently, it is hard to directly fuse the different estimated quantities (e.g., disparity and normals) to produce higher-quality results. Finally, the above networks are largely trained on indoor scene datasets (e.g., NYU dataset) and do not usually generalize to new object types. Such limitations reduce the utility of these techniques in applications like depth-based segmentation or disparity refinement, which require clean, accurate depth edges. Our data-driven approach is to jointly learn the error correlations across different channels in order to produce robust depth edges from mono or stereo input.

General segmentation In the context of non-semantic segmentation (i.e., object-level region extraction without assigning semantic labels), one of the most widely used interactive segmentation approaches is GrabCut [30], which builds GMM-based foreground and background color models. The state of the art in non-semantic segmentation is arguably the method of Arbeláez et al. [2], which operates at the level of contours and yields a hierarchy of segments. Classical segmentation methods that target standard color images have also been extended to make use of additional information. For example, Kolmogorov et al. [20] propose a version of GrabCut that handles binocular stereo video, Sundberg et al. [35] compute depth-ordered segmentations using optical flow from video sequences, and Dahan et al. [9] leverage scanned depth information to decompose images into layers. Ren and Bo [28] forgo handcrafted features in favor of learned sparse code gradients (SCG) for contour detection in RGB or RGBD images. In this vein, DEPTH CUT leverages additional channels of information (disparity and normals) that can be directly estimated from input mono or stereo images. By learning to fuse these channels, our method performs well even in ambiguous regions, such as textured or shaded areas, or where foreground–background colors are very similar. In Sect. 8, we present comparisons with state-of-the-art methods and their variants.

Layering Decomposing visual content into a stack of overlapping layers produces a simple and flexible ‘2.1D’ representation [27] that supports a variety of interactive editing operations [24]. Previous work explores various approaches for extracting 2.1D representations from input images. Amer et al. [1] propose a quadratic optimization that takes in image edges and T-junctions to produce a layered result, and later generalize the formulation using convex optimization. More recently, Yu et al. [39] propose a global energy optimization approach. Chen et al. [7] identify five different occlusion cues (semantic, position, compactness, shared boundary, and

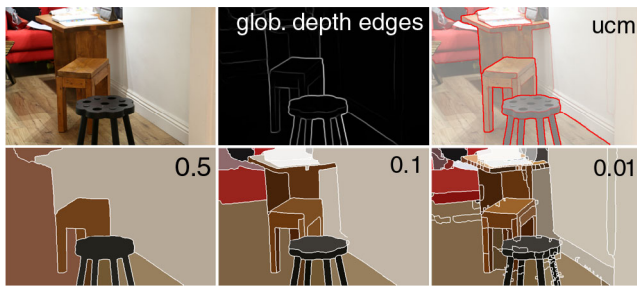


Fig. 3 Example of a region hierarchy obtained using depth edges estimated by DEPTHCUT. The cophenetic distance between adjacent regions (the threshold above which the regions are merged) is based on the strength of depth edges. The *Ultrametric Contour Map* [2] shows the boundaries of regions with opacity proportional to the cophenetic distance. Thresholding the hierarchy yields a concrete segmentation, we show three examples in the bottom row

junction cues) and suggest a preference function to combine these cues to produce a 2.1D representation. Given the difficulty in extracting layers from complex scenes, interactive techniques have also been proposed [16]. We offer an automatic approach that combines color, disparity, and normal information to decompose input images into layers with relatively flat depth.

3 Overview

DEPTHCUT estimates depth edges from either a stereo image pair or a single image. Depth edges consist of depth contours and creases that border regions of smoothly varying depth in the image. They correspond to approximate depth- or depth gradient discontinuities. These edges can be used to refine an initial disparity estimate, by constraining its gradients based on the depth edges, or to segment an image into a hierarchy of regions, giving us a depth layering of an image. Regions higher up in the segmentation hierarchy are separated by stronger depth edges than regions further down, as illustrated in Fig. 3.

Given an accurate disparity and normal estimate, depth edges can be found based on derivatives of the estimates over the image plane. In practice, however, such estimates are too unreliable to use directly (see Figs. 1, 4). Instead, we fuse multiple unreliable channels to get a more accurate estimate of the depth edges. Our cues are the left input image, as well as a disparity and normal estimate obtained from the input images. These channels work well in practice, although additional input channels can be added as needed. In the raw form, the input cues are usually inconsistent, i.e., the same edge, for example, may be present at different locations across the channels, or the estimates may contain edges that go missing in the other channels due to estimation errors.

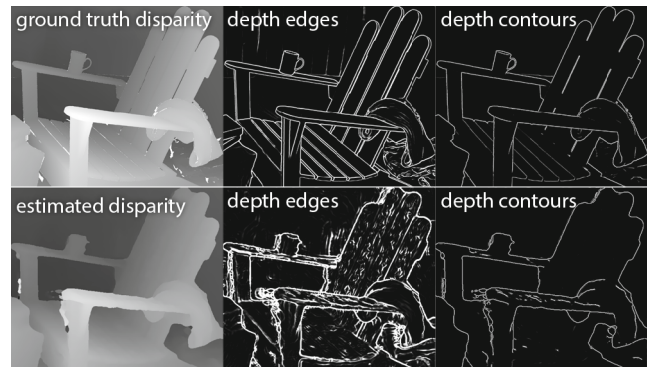


Fig. 4 Depth edges and contours computed by applying their definition directly to ground-truth disparities (top row) and estimated disparities (bottom row). High-order terms in the definition result in very noisy edges for the disparity estimate

The challenge then lies in fusing these different unreliable cues to get a consistent set of depth edges. The reliability of such channel features at a given image location may depend on the local context of the cue. For example, the color channel may provide reliable locations for contour edges of untextured objects, but may also contain unwanted texture and shadow edges. The disparity estimate may be reliable in highly textured regions, but inaccurate at disocclusions. Instead of hand-authoring rules to combine such conflicting channels, we train a convolutional neural network (CNN) to provide this context-sensitive fusion, as detailed in Sect. 5.

The estimated depth edges may be noisy and are not necessarily closed. To get a clean set of closed contours that decompose the image into a set of 2.1D regions, we adapt the non-semantic segmentation method proposed by Arbeláez et al. [2]. Details are provided in Sect. 6. The individual steps of our method are summarized in Fig. 5.

4 Depth edges

Depth edges consist of depth contours and creases. These edges separate regions of smoothly varying depth in an image, which can be used as segments, or to refine a disparity estimate. Our goal is to robustly estimate these depth edges from either a stereo image pair or a single image.

We start with a more formal definition of depth edges. Given a disparity image as continuous function $D(u, v)$ over locations (u, v) on the image plane, a *depth contour* is defined as a C^0 discontinuity of D . In our discrete setting, however, it is harder to identify such discontinuities. Even large disparity gradients are not always reliable as they are also frequently caused by surfaces viewed at oblique angles. Instead, we define the probability P_c of *depth contour* on the positive part of the Laplacian of the gradient:

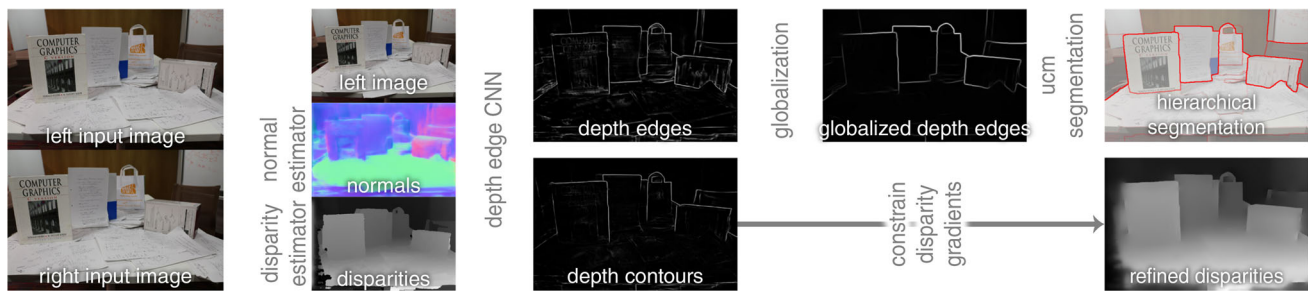


Fig. 5 Overview of our method and two applications. Starting from a stereo image pair, or a single image for monocular disparity estimation, we estimate our three input channels using any existing method for normal or disparity estimation. These channels are combined in a data-driven fusion using our CNN to get a set of depth edges, which are then used in two applications: segmentation and refinement of the

estimated disparity. (For the latter, see the supplementary material.) For segmentation, we perform a globalization step that keeps only the most consistent contours, followed by the construction of a hierarchical segmentation using the gPb-UCM framework [2]. For refinement, we use depth contours only (not creases) and use them to constrain the disparity gradients

$$P_c(u, v) := \sigma_\alpha((\Delta \|\nabla D\|)^+(u, v)),$$

where $\|\nabla D\|$ is the gradient magnitude of D , Δ is the Laplace operator, $(f)^+$ denotes the positive part of a function, and σ is a sigmoid function centered at α that defines a threshold for discontinuities. We chose a logistic function $\sigma_\alpha(x) = 1/(1 + e^{-10(x/\alpha-1)})$ with a parameter $\alpha = 1$.

Creases of 3D objects are typically defined as strong maxima of surface curvature. However, we require a different definition, since we want our creases to be invariant to the scale ambiguity of objects in images; objects that have the same appearance in an image should have the same depth creases, regardless of their world-space size. We therefore take the normal gradient of each component of the normal separately over the *image plane* instead of the divergence over geometry surfaces. Given a normal image $N(u, v) \in \mathbb{R}^3$ over the image plane, we define the probability P_T of *depth creases* on gradient magnitude of each normal component:

$$P_T(u, v) := \sigma_\beta((\|\nabla N_x\| + \|\nabla N_y\| + \|\nabla N_z\|)(u, v)),$$

where N_x , N_y , and N_z are the components of the normal, and σ is the logistic function centered at $\beta = 0.5$. The combined probability for a *depth edge* $P_c(u, v)$ is then given as:

$$P_c(u, v) := (1 - (1 - P_c)(1 - P_T))(u, v).$$

This definition can be computed directly on reliable disparity and normal estimates. For unreliable and noisy estimates, however, the high-order derivatives amplify the errors, as shown in Fig. 4. In the next section, we discuss how DEPTH-CUT estimates the depth edges using unreliable disparity and normals.

5 Depth edge estimation

We obtain disparity and normal estimates by applying state-of-the-art estimators either to the stereo image pair or to the left image only. Any existing stereo or mono disparity, and normal estimation method can be used in this step. Later, in Sect. 8, we report performance using various disparity estimation methods.

The estimated disparity and normals are usually noisy and contain numerous errors. A few typical examples are shown in Fig. 1. The color channel is more reliable, but contains several other types of edges as well, such as texture and shadow edges. By itself, the color channel alone provides insufficient information to distinguish between depth edges and these unwanted types of edges.

Our key insight is that the reliability of individual channels at each location in the image can be estimated from the full set of channels. For example, a short color edge at a location without depth or normal edges is likely to be a texture edge, or edges close to a disocclusion are likely to be noise if there is no evidence for an edge in the other channels. It would be hard to formulate explicit rules for these statistical properties, especially since they may be dependent on the specific estimator used. This motivates a data-driven fusion of channels, where we avoid hand-crafting explicit rules in favor of training a convolutional neural network to learn these properties from data. We will show that this approach gives better depth edges than a data-agnostic fusion.

5.1 Model

Deep networks have a large capacity to learn a context-sensitive fusion of channels based on different cues for their local reliability. We use a convolutional neural network (CNN), a type of network that has shown remarkable performance on a large range of image processing tasks [21,33].

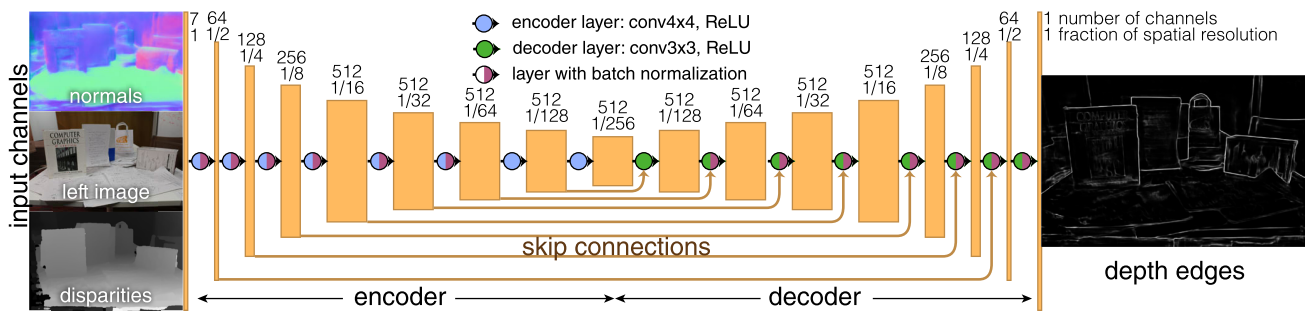


Fig. 6 CNN architecture for depth edge estimation. The orange boxes are layer input/output multi-channel images; colored disks are layers. Starting from a set of input channels, the encoder extracts a set of features of increasing abstraction while downsampling the feature maps.

The decoder uses these abstracted features to construct the depth edges. Skip connections let information flow around the bottleneck between encoder and decoder

The image is processed through a series of successive non-linear filter banks called layers, each operating on the output of the previous layer. The output of each layer is a multi-channel image $x \in \mathbb{R}^{w \times h \times c}$, where w and h are the width and height of the image and c corresponds to the number of filters in the layer:

$$\mathcal{L} : \mathbb{R}^{w \times h \times c} \rightarrow \mathbb{R}^{w' \times h' \times c'}$$

Each output channel can be understood as a *feature map* extracted from the input by one of the filters.

We base our network on the encoder–decoder architecture [15,18]. This architecture encodes image patches into a set of latent features and then decodes the desired output from these latent features. In a network of n layers, the first $n/2$ layers act as an encoder, where consecutive layers extract features of increasing abstraction from the input. The remaining $n/2$ layers act as decoder, where the features are used to construct the depth edge probability image. Figure 6 illustrates the architecture.

The input to the first layer is composed of the color channels I of the input image with size $W \times H \times 3$, the disparity estimate \tilde{D} , and the xyz channels of the normal estimate \tilde{N} , giving a total size of $W \times H \times 7$. The output of the last layer is the estimated probability \tilde{P}_e for a depth edge over the image:

$$\tilde{P}_e := (\mathcal{D}_1(p_1^d) \circ \dots \circ \mathcal{D}_n(p_n^d) \circ \mathcal{E}_n(p_n^e) \circ \dots \circ \mathcal{E}_1(p_1^e))(X),$$

where X is the concatenation of I , \tilde{D} , and \tilde{N} into a single multi-channel image, while \mathcal{E} and \mathcal{D} are encoder and decoder layers with parameters p_i^e and p_i^d , respectively.

Encoder An encoder layer is defined as:

$$\mathcal{E}(X | p) = (\gamma_2 \circ \sigma \circ bn \circ cv(p))(X),$$

where cv is a convolution layer with parameters p , bn denotes batch normalization [17], and γ_n denotes subsampling by a

factor of n . For the activation function σ , we choose a ‘leaky’ versions of the traditional rectified linear unit (ReLU) that has been shown [14] to reduce the well-known problem of inactive neurons due to the vanishing gradient of ReLUs.

Then the subsampling factor, the spatial extent of the filter kernels in cv , and the number of encoder layers determine the size of the image patches used to construct the latent features, with larger patches capturing more global properties. DEPTHCUT comprises of 8 encoder layers with a subsampling factor of 2, each with a kernel size of 4×4 , for a patch size of 256×256 pixels.

Decoder A decoder layer upsamples the input and is defined analogous to a decoder layer as:

$$\mathcal{D}(X | p) = (\sigma \circ bn \circ cv(p) \circ \lambda_2)(X),$$

where λ_n denotes upsampling by a factor of n . We set the upsampling factor equal to the subsampling factor of the encoder layers, so that chaining an equal amount of encoder and decoder layers results in an output of the same size. The last layer of the decoder replaces the leaky ReLU activation function with a sigmoid function to clamp the output to the $[0, 1]$ range of depth edge probabilities.

Skip connections The latent features in the bottleneck of our network contain a more abstract representation of each patch at the cost of a reduced spatial resolution. However, our output depth edges need to be spatially aligned to the fine details of the input channels. Similar to U-Net [29], we compensate the loss of spatial resolution by adding skip connections between encoder and decoder layers of the same resolution. The output of the encoder layer is appended as additional channels to the input of the decoder layer before the activation function. This provides the decoder with the needed spatial information.

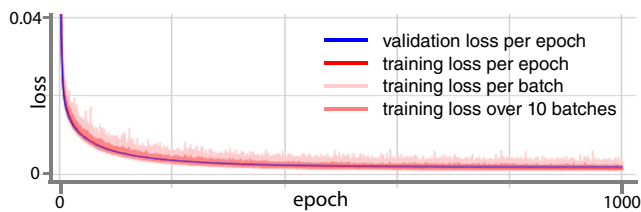


Fig. 7 Typical loss curve when training our model. Notice that the validation and training loss are nearly identical, suggesting little overfitting to the training set

5.2 Loss and training

We trained our model by comparing our output to ground-truth depth edges. In our experiments, the mean squared error performed best among various well-known loss functions. We did find, however, that comparisons in our datasets were biased to contain more loss from false negatives due to errors or inaccuracies of the disparity estimate (i.e., ground-truth depth edges that were missing in the output because they were not present in the disparity estimate) and than false positives due to texture or shadow edges (i.e., depth edges in the output due to texture or shadow edges that are not in the ground truth). To counteract this bias, we multiply the loss at color channel edges that do *not* coincide with depth edges by a factor of 10. Thus, we have

$$E(\tilde{P}_e, P_e, M) = \frac{1}{n} \|M \odot (\tilde{P}_e - P_e)\|_{\text{FRO}}^2,$$

where \tilde{P}_e and P_e are the estimated and ground-truth depth edges, respectively, M is a mask that takes on the value 10 at color edges that do not coincide with depth edges and 1 everywhere else, \odot denotes elementwise multiplication, and $\|X\|_{\text{FRO}}^2$ is the squared Frobenius norm of X .

We train the model using the Adam optimizer [19]. To combat overfitting, we randomly sample patches of size 256×256 from the input images during training and add an L_2 regularization term $\lambda \|p\|_2^2$ to the loss, where p are the parameters of our model and the scaling λ is set to 10^{-5} , effectively eliminating overfitting on our validation set. Figure 7 shows typical loss curve. In our experiments, we trained with a batch size of 5 input patches. For high-resolution images, our patch size only covers a relatively small fraction of the image, giving our model less global information to work with. To decrease the dependence of our network on image resolution, we downsample high-resolution images to 800 pixel width while maintaining the original aspect ratio.

6 Segmentation

Since depth edges estimated by DEPTH CUT separate regions of smoothly varying depth in the image, as a first application

we use them toward improved segmentation. Motivated by studies linking depth edges to perception of shapes, it seems plausible that regions divided by depth edges typically comprise simple shapes, that is, shapes that can be understood from the boundary edges only. Intuitively, our segmentation can then be seen as an approximate decomposition of the scene into simple shapes.

The output of our network typically contains a few small segments that clutter the image (see Fig. 6, for example). This clutter is removed in a globalization stage, where boundary segments are connected to form longer boundaries and remaining segments are removed.

To construct a segmentation from these globalized depth edges, we connect edge segments to form closed contours. The OWT-UCM framework introduced by Arbeláez et al. [2] takes a set of contour edges and creates a hierarchical segmentation, based on an oriented version of the watershed transform (OWT), followed by the computation of an *Ultra-metric Contour Map* (UCM). The UCM is the dual of a hierarchical segmentation; it consists of a set of closed contours with strength corresponding to the probability of being a true contour. (Please refer to the original paper for details.) A concrete segmentation can be found by merging all regions separated by a contour with strength lower than a given threshold (see Fig. 3).

The resulting UCM correctly separates regions based on the strength of our depth edges, i.e., the DEPTH CUT output is used to build the affinity matrix. We found it useful to additionally include a term that encourages regions with smooth, low curvature boundaries. Please see the supplementary material for details.

7 Depth refinement

As a second application of our method, we can refine our initial disparity estimates. We train our network to output *depth contours* as opposed to depth edges, i.e., a subset of the depth edges. Due to our multi-channel fusion, the depth contours are usually considerably less noisy than the initial disparity estimate. They provide more accurate spatial locations for strong disparity gradients. In addition to depth contours, we also train to output *disparity gradient directions* as two normalized components \tilde{d}_u and \tilde{d}_v , which provide robust gradient orientations at depth contours. However, we do not obtain the actual gradient magnitudes. Note that getting an accurate estimate of this magnitude over the entire image would be a much harder problem, since it would require regressing the gradient magnitude instead of classifying the existence of a contour.

We obtain a smooth approximation of the gradient magnitude from the disparity estimate itself and only use the depth contours and disparity directions to decide whether a location

on the image plane should exhibit a strong gradient or not, and to constrain the gradient orientation. The depth refinement can then be computed by solving a linear least squares problem, where strong gradients in the disparity estimate are optimized to be located at depth edges and to point in the estimated gradient directions. For details and results, please refer to the supplementary material.

8 Results and discussion

To evaluate the performance of our method, we compare against several baselines. We demonstrate that fusing multiple channels results in better depth edges than using single channels by comparing the results of our method when using all channels against our method using *fewer channels* as input. (Note that the extra channels used in DEPTHCUT are estimated from mono or stereo image inputs, i.e., all methods have the same source inputs to work with.) To support our claim that a data-driven fusion performs better than a *data-agnostic* fusion, we compare to a baseline using manually defined fusion rules to compute depth edges. For this method, we use large fixed kernels to measure the disparity or normal gradient across image edges. We also test providing the *un-fused* channels directly as input to a segmentation, using the well-known gPb-UCM [2] method, the UCM part of which we use for our segmentation application, as well. Finally, we compare against the method of Ren and Bo [28], where sparse code gradient (SCG)-based representations of local RGBD patches have been trained to optimize contour detection.

For each of these methods we experiment with different sets of input channels, including the color image, normals, and 3 different disparity types from state-of-the-art estimators: the mc-cnn stereo matcher [40], dispnets [23], and a monocular depth estimate [6], for a total of 18 baseline variations. More detailed segmentation results and all results for depth refinement are provided as supplementary materials, while here we present only the main segmentation results.

8.1 Datasets

We use two datasets to train our network, the Middlebury 2014 Stereo dataset [31], and a custom synthetic indoor scenes dataset we call the ROOM-dataset. We cannot use standard RGBD datasets typically captured with noisy sensors like the Kinect, because the higher-order derivatives in our depth edge definition are susceptible to noise (see Fig. 4). Even though the Middlebury dataset is non-synthetic, it has excellent depth quality. It consists of 23 images of indoor scenes containing objects in various configurations, each of which was taken under several different lighting conditions and with different exposures. We perform data augmentation

by randomly selecting an image from among the exposures and lighting conditions during training and by randomizing the placement of the 256×256 patch in the image, as described in Sect. 5.2.

The ROOM-dataset consists of 132 indoor scenes that were obtained by generating physically plausible renders of rooms in the scene synthesis dataset by Fisher et al. [11] using a light tracer. Since this is a synthetic dataset, we have access to perfect depth. Recently, several synthetic indoor scene datasets have been proposed [25,34] that would be good candidates to extend the training set of our method; we plan to explore this option in future work. The ground truth on these datasets is created by directly applying the depth edge definition in Sect. 4 to the ground-truth disparity. Please see the supplementary material for details.

Our network performs well on these two training datasets, as evidenced by the low validation loss shown in Fig. 7, but to confirm the generality of our trained network, we tested the full set of baselines on an unrelated dataset of 8 images (referred to as the CAMERA-dataset) taken manually under natural (non-studio) conditions, for a total of 144 comparisons with all baselines. These images were taken with three different camera types: a smartphone, a DSLR camera (hand-held or on a tripod), and a more compact handheld camera. They contain noise, blur, and the stereo alignment is not perfect. For these images, it is difficult to get accurate ground-truth depth, so we generated ground-truth depth edges by manually editing edge images obtained from the color channel, adding missing depth edges, and removing texture- and shadow edges, as well as edges below a threshold. We only keep prominent depth edges vital to a good segmentation to express a preference toward these edges (see the supplementary materials for this ground truth). For a future larger dataset, we could either use Mechanical Turk to generate the ground truth or use an accurate laser scanner, although the latter would make the capturing process slower, limiting the number of images we could generate.

In addition to this dataset, we test on a dataset of four Kinect RGBD images (referred to as the KINECT-dataset). Unlike for the other disparity types, our network was not specifically trained for Kinect disparity; thus, we expect lower-quality results than for the other disparity types. However, a performance above data-agnostic methods is an indication that our data-driven channel fusion can generalize to some degree to previously unseen disparity types.

8.2 Segmentation

In the segmentation application, we compute a hierarchical segmentation over the image. This segmentation can be useful to select objects from the image, or to composite image regions, where our depth edges allow for correct occlusion at region boundaries. For our segmentation application,



Fig. 8 Hierarchical segmentations based on depth edges. We compare directly segmenting the individual channels, performing a data-agnostic fusion, and using our data-driven fusion on either a subset of the input channels or all of the input channels. Strongly textured regions in the

color channel make finding a good segmentation difficult, while normal and disparity estimates are too unreliable to use exclusively. Using our data-driven fusion gives segmentations that better correspond to scene objects

we provide both qualitative and quantitative comparisons of the hierarchical segmentation on both the CAMERA- and the KINECT-datasets.

Qualitative comparisons on three images of the CAMERA-dataset are shown in Fig. 8. For each image, the hierarchical segmentation of all 21 methods (including our 3 results) is shown in 3×7 tables. The four images in the KINECT-dataset are shown in Fig. 9. The large labels on top of the figure denote the method, while the smaller labels on top of the images denote the input channels used to create the image. ‘Dispnet’ and ‘mc-cnn’ denote the two stereo disparity estimates and ‘mono’ the monocular disparity estimate.

Red lines show the UCM; stronger lines indicate a stronger separation between regions.

As is often the case in real-world photographs, these scenes contain a lot of strongly textured surfaces, making the objects in these scenes hard to segment without relying on additional channels. This is reflected in the methods based on color channel input that fail to consistently separate texture edges from depth edges. Another source of error is inaccuracies or errors in the estimates. This is especially noticeable in the normal and monocular depth estimates, where contours only very loosely follow the image objects. Methods with fewer input channels have less information to correct

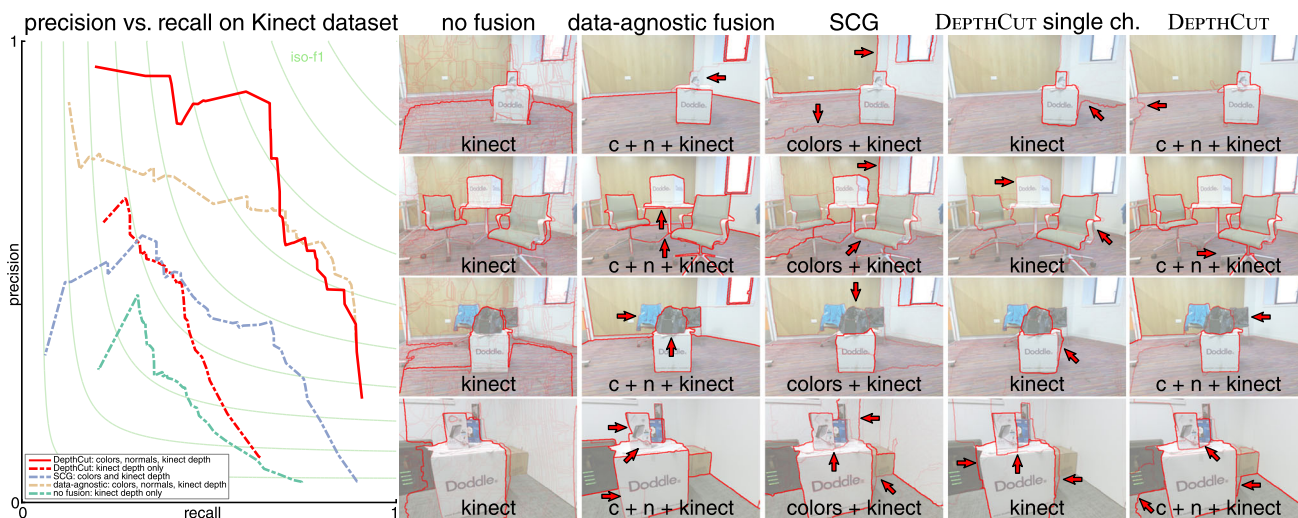


Fig. 9 Comparison to all baselines on the KINECT-dataset. Precision versus recall is shown on the left, and a qualitative comparison is given on the right. Arrows highlight some important differences and errors.

Even though DEPTHCUT was not trained specifically on this disparity type, it still performs better than other methods, although with a smaller advantage

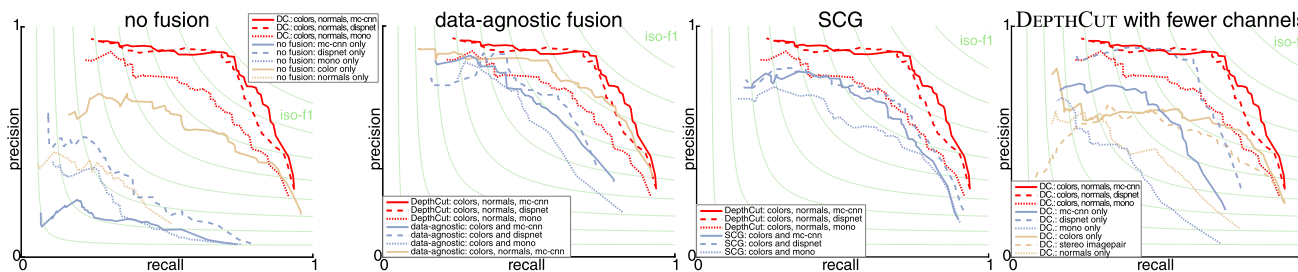


Fig. 10 Quantitative comparison to all baselines on our CAMERA-dataset. We show precision versus recall over thresholds of the segmentation hierarchy. Note that depth edges from monocular disparity estimates (finely dotted lines) work with less information than the other

two disparity estimates and are expected to perform worse. The depth edge estimates of our data-driven fusion, shown in red, consistently perform better than other estimates

these errors and have therefore generally less accurate and less robust contours. Using multiple channels without proper fusion does not necessarily improve the segmentation, as is especially evident in the multi-channel input of the un-fused method in lower-left corner of Fig. 8, and to a lesser extent in the more error-prone edges of the data-agnostic fusion. In Kinect disparities, the main errors are due to missing disparity values near contours and corners and on non-diffuse surfaces. Many estimates therefore contain false positives that are in some cases even preferred over true positives, in particular for methods with fewer channels and simpler fusion, as shown in Fig. 9. DEPTHCUT can correct errors in the individual channels giving us more robust region boundaries that are better aligned to depth edges, as shown in the right-most column.

Quantitative comparisons were made with all baselines images of the CAMERA-dataset. We compare to the ground truth using the Boundary Quality Metric [22] that computes precision and recall of the boundary pixels. We use the less

computationally expensive version of the metric, where a slack of fixed radius is added to the boundaries to not overly penalize small inaccuracies. Since we have hierarchical segmentations, we compute the metric for the full range of thresholds and report precision versus recall.

Results for the CAMERA-dataset are shown in Fig. 10. The four plots show precision versus recall for each method, averaged over all images. The *f1* score, shown as iso-lines in green, summarizes precision and recall into a single statistic where higher values (toward the top-right in the plots) are better. Monocular depth estimates that operate with less information than stereo estimates are expected to perform worse. Note that our fusion generally performs best, only the monocular depth estimates have a lower score than the stereo estimates of some other methods.

Figure 9 shows results for the KINECT-dataset. Recall that our method is trained specifically for each type of disparity estimate to identify and correct typical errors in this estimate with the help of additional channels. Somewhat surprisingly,

applying DEPTH CUT trained only on ‘dispnet’ disparities to Kinect disparities still gives an advantage over data-agnostic fusion. This suggests that our method is able to partially generalize the learned channel fusion to other disparity types. The advantage over fewer input channels is also evident.

9 Conclusions

We present a method that produces accurate depth edges from mono or stereo images by combining multiple unreliable information channels (color images, estimated disparity, and estimated normals). The key insight is that the above channels, although noisy, suffer from different types of errors (e.g., in texture or depth discontinuity regions), and a suitable context-specific filter can fuse the information to yield high-quality depth edges. To this end, we trained a CNN using ground-truth depth data to perform this multi-channel fusion, and our qualitative and quantitative evaluations show a significant improvement over alternative methods.

We see two broad directions for further exploration. From an analysis standpoint, we have shown that data-driven fusion can be effective for augmenting color information with estimated disparity and normals. One obvious next step is to try incorporating even more information, such as optical flow from input videos. While this imposes additional constraints on the capture process, it may help produce even higher-quality results. Another possibility is to apply the general data-driven fusion approach to other image analysis problems beyond depth edge estimation. The key property to consider for potential new settings is that there should be good correlation between the various input channels.

Another area for future research is in developing more techniques that leverage estimated depth edges. We demonstrate how such edges can be used to refine disparity maps and obtain a segmentation hierarchy with a partial depth ordering between segments. While our work already demonstrates how such edges can be used to refine disparity maps, we feel there are opportunities to further improve depth and normal estimates. The main challenge is how to recover from large depth errors, as our depth edges only provide discontinuity locations rather than the gradient magnitudes. It is also interesting to consider the range of editing scenarios that could benefit from high-quality depth edges. For example, the emergence of dual camera setups in mobile phones raises the possibility of on the fly, depth-aware editing of captured images. In addition, it may be possible to support a class of pseudo-3D edits based on the depth edges and refined depth estimates within each segmented layer.

Acknowledgements This work was supported by the ERC Starting Grant SmartGeometry (StG-2013-335373), the Open3D Project (EPSRC Grant EP/M013685/1), and gifts from Adobe.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflicts of interest.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Amer, M.R., Raich, R., Todorovic, S.: Monocular extraction of 2.1d sketch. In: ICIP (2010)
2. Arbeláez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. *IEEE PAMI* **33**(5), 898–916 (2011)
3. Bakry, A., Elhoseiny, M., El-Gaaly, T., Elgammal, A.M.: Digging deep into the layers of cnns (2015). CoRR [arXiv:1508.01983](https://arxiv.org/abs/1508.01983)
4. Bansal, A., Kowdle, A., Parikh, D., Gallagher, A., Zitnick, L.: Which edges matter? In: IEEE ICCV (2013)
5. Boulch, A., Marlet, R.: Deep learning for robust normal estimation in unstructured point clouds. In: SGP (2016)
6. Chakrabarti, A., Shao, J., Shakhnarovich, G.: Depth from a single image by harmonizing overcomplete local network predictions. In: NIPS, pp. 2658–2666 (2016)
7. Chen, X., Li, Q., Zhao, D., Zhao, Q.: Occlusion cues for image scene layering. *Comput. Vis. Image Underst.* **117**, 42–55 (2013)
8. Couprie, C., Farabet, C., Najman, L., LeCun, Y.: Indoor semantic segmentation using depth information (2013). CoRR [arXiv:1301.3572](https://arxiv.org/abs/1301.3572)
9. Dahan, M.J., Chen, N., Shamir, A., Cohen-Or, D.: Combining color and depth for enhanced image segmentation and retargeting. *TVCI* **28**(12), 1181–1193 (2012)
10. Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture (2014). CoRR [arXiv:1411.4734](https://arxiv.org/abs/1411.4734)
11. Fisher, M., Ritchie, D., Savva, M., Funkhouser, T., Hanrahan, P.: Example-based synthesis of 3d object arrangements. *ACM TOG* **31**(6), 135:1–135:11 (2012)
12. Gibson, J.J.: *The Ecological Approach to Visual Perception*. Routledge, Abingdon (1986)
13. Gupta, S., Girshick, R.B., Arbeláez, P.A., Malik, J.: Learning rich features from rgb-d images for object detection and segmentation. In: ECCV (2014)
14. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: IEEE ICCV, pp. 1026–1034 (2015)
15. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. *Science* **313**(5786), 504–507 (2006)
16. Iizuka, S., Endo, Y., Kanamori, Y., Mitani, J., Fukui, Y.: Efficient depth propagation for constructing a layered depth image from a single image. In: CGF, vol. 33, No. 7 (2014)
17. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML, pp. 448–456 (2015)
18. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks (2016). CoRR [arXiv:1611.07004](https://arxiv.org/abs/1611.07004)
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)

20. Kolmogorov, V., Criminisi, A., Blake, A., Cross, G., Rother, C.: Bi-layer segmentation of binocular stereo video. In: *IEEE CVPR*, vol. 2, pp. 407–414 (2005)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: *NIPS*, pp. 1097–1105 (2012)
22. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE PAMI* **26**(5), 530–549 (2004)
23. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: *IEEE CVPR*, pp. 4040–4048 (2016)
24. McCann, J., Pollard, N.: Local layering. *ACM TOG* **28**(3), 84:1–84:7 (2009)
25. McCormac, J., Handa, A., Leutenegger, S., Davison, A.J.: Scenet RGB-D: 5m photorealistic images of synthetic indoor trajectories with ground truth (2016). CoRR [arXiv:1612.05079](https://arxiv.org/abs/1612.05079)
26. Mitra, N.J., Wand, M., Zhang, H., Cohen-Or, D., Kim, V., Huang, Q.X.: Structure-aware shape processing. In: *ACM SIGGRAPH 2014 Courses*, pp. 13:1–13:21 (2014)
27. Nitzberg, M., Mumford, D.: The 2.1-d sketch. In: *IEEE ICCV*, pp. 138–144 (1990)
28. Ren, X., Bo, L.: Discriminatively trained sparse code gradients for contour detection. In: *NIPS* (2012)
29. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: *MICCAI*, pp. 234–241 (2015)
30. Rother, C., Kolmogorov, V., Blake, A.: grabcut: interactive foreground extraction using iterated graph cuts. *ACM TOG* **23**(3), 309–314 (2004)
31. Scharstein, D., Hirschmiller, H., Kitajima, Y., Krathwohl, G., Nesić, N., Wang, X., Westling, P.: High-resolution stereo datasets with subpixel-accurate ground truth. In: *GCPR, Lecture Notes in Computer Science*, vol. 8753, pp. 31–42. Springer, Berlin (2014)
32. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: OverFeat: integrated recognition, localization and detection using convolutional networks (2013). CoRR [arXiv:1312.6229](https://arxiv.org/abs/1312.6229)
33. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: *ICLR* (2015)
34. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: *IEEE CVPR* (2017)
35. Sundberg, P., Brox, T., Maire, M., Arbelaez, P., Malik, J.: Occlusion boundary detection and figure/ground assignment from optical flow. In: *IEEE CVPR* (2011)
36. Szegedy, C., Toshev, A., Erhan, D.: Deep neural networks for object detection. In: *NIPS* (2013)
37. Toshev, A., Szegedy, C.: Deeppose: human pose estimation via deep neural networks. In: *IEEE CVPR* (2014)
38. Xu, K., Kim, V.G., Huang, Q., Mitra, N., Kalogerakis, E.: Data-driven shape analysis and processing. In: *SIGGRAPH ASIA 2016 Courses*, pp. 4:1–4:38 (2016)
39. Yu, C.C., Liu, Y.J., Wu, M.T., Li, K.Y., Fu, X.: A global energy optimization framework for 2.1d sketch extraction from monocular images. *Graph. Models* **76**(5), 507–521 (2014). *Geometric Modeling and Processing 2014*
40. Žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.* **17**(1), 2287–2318 (2016)
41. Zhu, H., Weibel, J.B., Lu, S.: Discriminative multi-modal feature fusion for rgbd indoor scene recognition. In: *IEEE CVPR* (2016)



Paul Guerrero is a post doc at University College London, working on shape analysis and image editing, combining methods from machine learning, optimization, and computational geometry. He received his Ph.D. in computer science from Vienna University of Technology. Paul has published several research papers in high-quality journals and is a regular reviewer for conferences and journals and a conference IPC member.



Holger Winnemöller received his M.Sc. from Rhodes University in South Africa and his Ph.D. from Northwestern University in the USA, investigating perceptual aspects of Non-photorealistic Rendering. Currently, Holger is a principal scientist with Adobe Research, working on tools to support casual creativity for novices. In addition to his academic work, Holger has published several popular iOS creativity apps.



Wilmot Li is a Principal Scientist at Adobe Research. He joined Adobe in 2008 after completing his Ph.D. in computer science in the Graphics and Imaging Lab at the University of Washington. Wils research lies at the intersection of computer graphics and human-computer interaction, and in recent years, he has focused on performance-based 2D animation and design tools for vector graphics and fabrication.



Niloy J. Mitra is a Professor of Geometry Processing in the Department of Computer Science, University College London. His research interests include shape analysis, computational design and fabrication, and data-driven geometry processing. He received an ACM SIGGRAPH Significant New Researcher Award and a BCS Roger Needham Award and currently holds an ERC Starting Grant.