Dense connection decoding network for crisp contour detection

Guili Xu1,3 | Chuan Lin1,2 | Yuchua Cheng1

1 College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Yudao Street 29, Nanjing, People’s Republic of China
2 College of Electric and Information Engineering, Guangxi University of Science and Technology, Donghu Road 268, Liuzhou, Guangxi, People’s Republic of China
3 Nondestructive Detection and Monitoring Technology for High Speed Transportation Facilities, Key Laboratory of Ministry of Industry and Information Technology, Nanjing, China

Abstract
In the past few years, contour detection algorithm has made obvious progress with the help of convolutional neural networks. The aim of this paper is to present a novel network connecting low- and high-resolution features to make the network achieving richer feature representation. First, VGG net is used as encoding part with outputting the features of different resolutions, and then the feature maps are combined in some specific resolution with up- or down-sample method. The combining process can be stack step-by-step. The proposed network makes the encoding part deeper to extract richer convolutional features. The experiments have shown that the proposed method improves the contour detection performances and outperform some existed convolutional neural networks based methods on BSDS500 and NYUD-V2 datasets.

1 | INTRODUCTION
Contour detection plays a fundamental role in the field of computer vision applications such as image segmentation and recognition [1]. In these applications, a contour normally represents a change in pixel ownership from one object or surface to another in natural images. Contour detection is generally considered as a low-level technology, and the development of contour detection has greatly benefited from various high-level tasks, such as image segmentation [2] and object recognition [3].

Early contour detection approaches focused on finding local discontinuities, normally brightness, in image features, such as Sobel [4] and Canny [5]. More recent approaches account for multiple image features, such as colour and texture information, and use statistical and learning techniques [2, 6], active contours [7] and graph theory [8]. Generally, machine learning methods first extract local cues of brightness, colours, gradients and textures or other manually designed features, and then use logistic regression to classify edge and non-edge pixels, such as Pb [9] and gPb [10].

Convolutional neural networks (CNNs) have become popular in the field of computer vision in the past few years. CNNs substantially improve many tasks, including image classification [11–13], object detection [14], semantic segmentation [15] etc. Recently, many well-known CNNs-based contour detection methods have been proposed, such as HED [16], RCF [17] and CED [18]. Early CNNs-based contour detection model treat contour detection task as an image classification problem, which used different image patch as input to predict the category of the central pixel of the block, such as DeepContour [19] and DeepEdge [20]. Inspired by fully convolutional networks [21] and deconvolutional networks [22], subsequent model used VGG [12] or ResNet [13] as encoding network to extract features and exploit deconvolution layer to up-sample feature map to original size as decoding network. This structure can fully exploit the performance of CNNs and can easily do end-to-end training, thus has achieved a superior performance even human level [16, 18].

These models, such as HED [16] illustrated in Figure 1(a), is a holistically nested architecture, which uses side outputs...
as feature maps and produces contour from multiple scales. For further development the feature representation of network, we propose dense connection decoding network (DCDN) that constructs hierarchical architecture to extract more complex and abstractive features, as in Figure 1(b). The first level receives the side output from certain encoding networks, such as VGG16 [23] and ResNet [24], and the followed levels integrate features from the low-level outputs.

Beyond the common criteria for assessing “correctness” of edges (distinguishing edge pixels from non-edge pixels), Isola et al. [25] discussed that an accurate edge detector must balance the “correctness” and “crispness” of the boundary (precisely localizing edge pixels). Therefore, they matched the ground-truth edges for “crispness” by decreasing the maximal permissible matching distance. In addition, both qualitative and quantitative results show that edge maps from a CNN are highly “correct” yet less “crisp”—edges are not sufficiently localised. This problem is deeply existed in modern CNNs architecture [11], because pooling layers will cause fuzzy output and fully convolutional architecture encourages neighbouring pixels to produce similar responses, so it unavailable to generate thinning edge map.

The proposed DCDN is expert in learning crisp edge. Compared with HED [16], this structure is more powerful in fusing different scales, and thus make a significant improvement in normal and crisp contour detection. DCDN is an extension of HED. As in Figure 1(a), HED only combines multiple side output once, while the proposed DCDN, as in Figure 1(b), uses hierarchical architecture. In addition, we use 3 × 3 convolutions and ReLU activating function instead of 1 × 1 convolutions in HED. These enable a more expressive model and improve the performance of crispness of edges. The experiments show that proposed methods outperform compared method on BSDS500 and NYUD-v2 datasets.

2 | RELATED WORKS

Early contour detection approaches are focused on finding local discontinuities, normally brightness, in image features. The Prewitt [26] operators detect edges by convolving a greyscale image with local derivative filters. Marr and Hildreth [27] used zero crossings of the Laplacian of Gaussian operator to detect edges. The Canny detector [5] also computes the gradient magnitude in the brightness channel, adding post-processing steps including non-maximum suppression (NMS) and hysteresis thresholding.

More recent approaches use multiple image features, such as colour and texture information, and apply biologically motivated methods [28–30] or learning techniques [6, 31–33]. Martin et al. [6] built a statistical framework for the difference of brightness (BG), colour (CG), and texture (TG) channels, and used these local cues as inputs for a logistic regression classifier to predict the probability of boundary (Pb) . Dollar et al. [31] proposed a Boosted Edge Learning algorithm that aims to learn and create a probabilistic boosting tree classifier to detect contours by using thousands of simple features computed on image patches. For integrating multiple scales of information, Ren et al. [32] used local boundary cues including contrast, localisation, and relative contrast. In order to make full use of global visual information, Arbelaez et al. [2] proposed a global Pb (gPb) algorithm designed to extract contours from global information by using the eigenvectors obtained from spectral partitioning. Ren et al. [33] used Sparse Code Gradients (SCG) to extract salient contours from sparse codes in oriented local neighbourhoods. Khoreva et al. [34] introduced the problem of weakly supervised object-specific boundary detection and suggest that good performance can be obtained on many datasets using only weak-supervision (i.e., leveraging bounding box detection annotations without the need of instance-wise object boundary annotations).
In recent years, CNNs have been widely used in the field of computer vision and machine learning. Ganin et al. [35] used a deep architecture to extract features of image patches. They approached contour detection as a multiclass classification task, by matching the extracted features to predefined ground-truth features. Bertasius et al. [20] made use of features generated by pre-trained CNNs to regress and classify the contours. They proved that object-level information provides powerful cues for the prediction of contours. Shen et al. [19] learnt deep features using mid-level information. Xie et al. [16] developed an end-to-end CNN to boost the efficiency and accuracy of contour detection, called HED, using convolutional feature maps and a novel loss function. HED connects its side output layers, which is composed of one conv layer with kernel size 1, one deconv layer and one softmax layer, to the last conv layer of each stage in VGG16. Based on HED architecture, Kokkinos [36] built multi-scale HED and improved the results by tuning the loss function and adding globalisation. Liu et al. [17] added the side output layers upon HED to extract richer convolutional features. Wang et al. [18] combined the refinement scheme with sub-pixel convolution [37] into a novel architecture, which was specifically designed for learning crisp edge detector. Yang et al. [38] proposed a novel encoder-decoder architecture to tackle the contour detection task. Xu et al. [39] introduced a hierarchical model to extract multi-scale features and a gated conditional random field to fuse them. He et al. [40] proposed a bi-directional model to further fuse multi-scale information. However, those methods do not further consider the problem of full fusion of features. Features of different resolutions should promote the performance of contour detection in mutual fusion. Based on this idea, we proposed a network trying to explore the problem of multi-scale full integration.

3 | PROPOSED METHOD

The proposed architecture has multiple levels with different resolutions. We first describe the DCDN architecture in Section 3.1, and then we describe the loss function in Section 3.2. Finally, we introduce the multi-scale version of the proposed method.

3.1 | Architecture

The generic contour detection model includes an encoding network, such as VGG16 or ResNet, and a decoding network which aims at fusing multi-resolution side outputs of encoding parts. Inspired by HED, we design our network by modifying VGG16 as encoding part and proposed a novel decoding network DCDN. The VGG16 network, consisting of 13 conv layers and 3 fully connected layers, has reached significant performances in a variety of tasks. Its conv layers are divided into five stages, where a pooling layer is connected after each stage. The useful information obtained by each conv layer will become coarser as its receptive field size increases. In this paper, we dropout the fully connected layers and the pool5 layer because we want to design a pixel-to-pixel prediction rather classify model. DCDN combines each side output via refinement blocks and expands it several times, which makes the network deeper to extract richer and more complex features and improves the generalisation ability. The network architecture is shown in Figure 1(b) and the fusion layer is shown in Figure 2.

3.1.1 | Refinement block

The refinement block includes conv, up-sample and fusion layers, as in Figure 2. We first use a 3 × 3 conv layer to filtering, and then use bilinear interpolation method resizing the feature maps into specific resolution. In addition, we joint all the maps with a 1 × 1 conv layer to fuse multiple inputs. Note that each conv will follow a ReLU activation function.

3.1.2 | Treating different levels

In our model, we use 3 level to fusing feature with 3, 3, 1 refinement block. The first level combines all the resolution maps (1, 1/2, 1/4, 1/8, 1/16) and outputs three resolution maps (1, 1/2, 1/8). Subsequently, the level in middle receives three resolution maps and outputs three feature maps with the same resolution. As the last level, the refinement block outputs one map with original size and then follows a 1 × 1 conv layer with sigmoid function. All conv layer will be activated by ReLU function except the last.

The core of DCDN is the deeper stacking using with resolution combining of refinement block. We aim to exploit the integration of multi-level features for high-resolution prediction in end-to-end computer vision tasks. Comparing to the traditional networks which just had several refinement modules in one level, DCDN has multiple levels which make the network deeper and more robust. Another important characteristic is the reusability of feature maps. As shown in Figure 1(b), the features of the previous level convey information to all scales in the next

![FIGURE 2 Detailed fusion layer in Figure 1(b). The conv layer followed a ReLU activation function; interpolate part is bilinear interpolate algorithm; concat layer joint interpolate output with channel dimension](image-url)
level. More importantly, DCDN can easily stacking refinement level to extend the network for more complication.

3.2 Loss function

We denote our input training data set by \( \mathcal{S} = \{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\} \), where sample \( x^n \) and \( y^n \) denote the raw input image the corresponding ground truth binary edge map for image \( x^n \). We subsequently drop the symbol \( n \) for notational simplicity. Following HED, the loss function is as follows:

\[
\mathcal{L} = -\beta \sum_{i \in I_+} \log p_i - (1 - \beta) \sum_{i \in I_-} \log (1 - p_i)
\]  

(1)

where \( I_+ \) and \( I_- \) denote the edge pixel set and non-edge pixel sets, respectively. \( \beta = |I_-|/|I| \) and \( 1 - \beta = |I_+|/|I| \). \( p_i \) is the CNN output processed by sigmoid function at pixel \( i \). By using weights \( \beta \) and \( 1 - \beta \), this solution help CNNs better train the network.

However, only using the loss between predication contour map and Ground Truth is far from enough, because our network is to deep and the gradient is hard propagation from the final output to the preliminary layer. Hence, we use the deep supervision technique to train our model.

Considering we sample five side output from the VGG network and each of them has a different resolution. We first adopt five \( 1 \times 1 \) conv layer to squeeze the channels into one and interpolate the feature to original size. Then, we process them with sigmoid activation and push them in supernumerary loss of \( \mathcal{L} \). We denote these loss as \( \mathcal{L}_{\text{side}}^i \), where \( i = 1, 2, \ldots, m \). Thus, the total loss is shown below

\[
\mathcal{L}_{\text{total}} = \mathcal{L} + \alpha \sum_{i} \mathcal{L}_{\text{side}}^i
\]  

(2)

where \( \alpha = 0.2 \). With deep supervision, the gradient through back propagation can easily train the hold network.

In BSDS training, we adopt a consensus sampling strategy [17] to prevent problematic convergence behaviour, because the BSDS dataset has multiple Ground Truth annotations for one image. This strategy is described as: if a pixel is marked as an edge by at least three annotators, it is assigned a positive label. Pixels are treated as background as anyone marked it as negative. Ignore the remaining pixels during training.

3.3 Multi-scale contour detection

We build image pyramids to detect multi-scale contours, as used in paper [17, 18]. In single-scale edge detection, we input the original image into our fine-tuned CNN network and then output an edge probability map. To further improve the quality of edges, we use image pyramids during the test. Specifically, we resize the images to build an image pyramid and input each image separately to our single-scale detector. Then, all resulting edge probability maps are resized to the original image size using bilinear interpolation. Finally, these maps are averaged to obtain the final prediction map. Considering the trade-off between accuracy and speed, we use three scales 0.5, 1.0 and 2.0 in this paper.

4 EXPERIMENTS

A pre-trained VGG16 model on ImageNet [41] was used to initialise our network. In training, the weights of the conv layer are initialised from a zero-mean Gaussian distribution (standard deviation is 0.01), and the bias is initialised to 0. Mini-batch of Random gradient descent (SGD) is set to 10. The global learning rate is set to \( 2 \times 10^{-5} \). Momentum and weight decay are set to 0.9 and \( 2 \times 10^{-4} \), respectively.

The standard methods to evaluate the contour map and Ground Truth including the optimal dataset scale (ODS), which uses a fixed threshold for all images in the dataset, the optimal image ratio (OIS), which selects the best threshold for each image, and the average accuracy (AP). We use F-measure \( (F = \frac{2P \times R}{P + R}) \) of both ODS and OIS in our experiments, where \( R \) denotes the recall ratio reflecting the probability that the detected edge is valid and \( P \) denotes the precision ratio reflecting the probability that the ground truth edge was detected.

### 4.1 BSDS500 dataset

BSDS500 [2] is a widely used dataset in edge detection. It consists of 200 training, 100 validation and 200 test images, each image being labelled by multiple annotators. We use training and validation for fine-tuning, and test set for evaluation. Data augmentation is the same as [16]. During evaluating, the standard NMS [42] was applied to thin the detected edges.

### 4.1.1 Ablation study

To evaluate the performance of the proposed method, we conducted experiments with several settings, including using the original HED as our baseline, training different versions of DCDN by setting different parameters. Table 1 shows some experimental results. We can see that the proposed DCDN has
better ODS and OIS, whereas the lower AP because HED-fused multi-scale on different side output but we only use deep supervision technique for training yet testing. We can find that the 3-layer, 3-block is the best choice for our network compared with 3-layer, 5-block version, which has more computational time in training and testing. We also see that DCDN (5-layer, 5-block) version has lower performances because the deeper layer will cause hard gradient to propagate, so this version is difficult to convergence. The more layer and blocks will significantly add the computation cost while improving little or even decreasing the performance.

As can be observed from Table 1, in comparison with the single-scale version, the multi-scale DCDN (DCDN-MS) improve the contour performance on ODS and OIS, but decrease the AP.

4.1.2 Compared with the state-of-the-art

We compare our method with some deep-learning and non-deep-learning methods. The non-deep-learning methods include Canny [5], Pb [6], gPb-UCM [2], MCG [43], SE [42] and OEF [44]. The deep-learning methods include DeepContour [19], DeepEdge [20], HED [16], etc. Figure 3 shows the Precision-Recall curves and Table 2 shows the evaluation results in terms of ODS, OIS and AP. As we see, our DCDN outperform the compared methods in the measures of ODS, OIS and AP. This result near the human level (ODS 0.803) on its multi-scale version. Note that we use 3-layer DCDN-MS for this comparison.

4.2 NYUD-v2 dataset

The NYUD [46] dataset consists of 1449 densely labelled paired RGB and depth image pairs. Recently, it has been marginally evaluated in many works, such as [17, 33, 42]. The NYUD dataset was divided into 381 trainings, 414 validations and 654 test images [47]. We utilise depth information by using HHA [45], where the depth information is encoded into three channels: horizontal disparity, height above ground and angle with gravity. Therefore, HHA features can be represented as colour images. Then, two models of RGB image and HHA feature image were trained respectively. We rotate the image and corresponding annotations to 4 different angles (0, 90, 180 and 270 degrees) and flip them at each angle. During the training, the network settings were the same as those used in the BSDS500. In testing, the final edge prediction is defined by averaging the output of the RGB and the HHA model. In the evaluation, we increased the tolerance from $\text{maxDist} = 0.0075$ to 0.011 as in [17] because the image in the NYUD dataset is larger than the image in the BSDS500 dataset.

We compared our method with several non-deep-learning algorithms, such as gPb-UCM [2], SE [42], gPb+NG [47], SE+NG+ [48] and recent deep learning based approaches,
such as HED [16]. The precision-recall curves are shown in Figure 6 and the statistical comparison is shown in Table 3. DCDN achieves the best performance on the NYUD dataset (ODS = 0.752). We can see that DCDN achieves better results than HED on separated RGB, HHA and merged HHA- RGB data. It suggests that deeper architecture is very useful for contour detection.

4.3 Efficiency analysis

Computational efficiency is important for contour detection, which is often used as a pre-processing for high-level visual tasks. The model size of HED is 56.12 M, while the proposed model size is 68.35 M. The single-scale version of the proposed method achieved a value of 24 FPS when processing a 481 × 321 resolution image on a GTX 1080Ti GPU, while HED achieves 30FPS. The proposed method is slower than HED due to that we use more refinement blocks.

5 DISCUSSION AND CONCLUSIONS

The differences between our network and HED include in two parts. First, HED just considers one level combination in each stage of VGG16, such as Figure 1(a). The proposed DCDN make the decode network deeper and fusing multi-lever feature to detect contour. Thus, it can collect more information and improve the performances. Second, we not only up sample the low-resolution map to higher (see Figure 1(b) the down to up arrows), but also down sample the high-resolution feature to the lower (see Figure 1(b) the up to down arrows), which is more important to make the network to extract crisp boundary, such as Figures 4 and 5 for qualitative and quantitative experiments.
The quantitative results for the analysis of the two benchmark datasets (BSDS, NYUD-V2) demonstrated that the DCDN systematically exceeded the HED in most comparisons. In the standard correctness comparison, as in Figure 3, we outperform many CNNs-based methods on the BSDS500 and NYUD datasets. More importantly, the proposed dense decoding architecture significantly improved the crispness performance of contour on the BSDS500 dataset.

In this paper, we proposed a novel CNN architecture DCDN that makes use of multi-level refinement modules to build deep networks. Based on the DCDN, we explored a high-quality contour detector that achieves promising performance on BSDS500 and NYUD-v2.

ACKNOWLEDGEMENT

This study was supported by the National Natural Science Foundation of China (Nos. 61473148). Nondestructive Detection and Monitoring Technology for High Speed Transportation Facilities, Key Laboratory of Ministry of Industry and Information Technology.

REFERENCES

1. Papari, G., Perlov, N.: Edge and line oriented contour detection: State of the art. Image Vision Comput. 29(2), 79–103 (2011).
2. Arbelaez, P., et al.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 33(5), 898–916 (2011).
3. Zitnick, C.L., Dollár, P.: Edge boxes: Locating object proposals from edges. In: Proceedings of the European Conference on Computer Vision, Springer, pp. 391–405 (2014).
4. Duda, R.O., Hart, P.E., Pattern Classification and Scene Analysis. New York: Wiley (1973).
5. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. 8(6), 679–698 (1986).
6. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans. Pattern Anal. Mach. Intell. 26(5), 530–549 (2004).
7. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. Int. J. Comput. Vision 1(4), 321–331 (1988).
8. Hummel, R.A., Zucker, S.W.: On the foundations of relaxation labeling processes. IEEE Trans. Pattern Anal. Mach. Intell. 5(3), 267–287 (1983).
9. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans. Pattern Anal. Mach. Intell. 26(5), 530–549 (2004).
10. Arbelaez, P., et al.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 33(5), 898–916 (2011).
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012).
12. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Representation Learning (2015).
13. He, K., et al.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016).
14. Redmon, J., et al.: You only look once: Unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016).
15. Lin, G., et al.: Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (2017).
16. Xie, S., Tu, Z.: Holistically-nested edge detection. In: International Conference on Computer Vision, pp. 1395–1403 (2015).
17. Liu, Y., et al.: Richer convolutional features for edge detection. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp. 5872–5881 (2017).
18. Wang, Y., Zhao, X., Huang, K.: Deep crisp boundaries. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3892–3900 (2017).
19. Chen, W., et al.: Deeppostural: A deep convolutional feature learned by positive-sharing loss for contour detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3982–3991 (2015).
20. Bertasius, G., Shi, J., Torresani, L.: Deepscale: A multi-scale bifurcated deep network for top-down contour detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4380–4389 (2015).
21. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015).
22. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1520–1528 (2015).
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015).
24. He, K., et al.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016).
25. Isola, P., et al.: Crisp boundary detection using pointwise mutual information. In: Proceedings of the European Conference on Computer Vision, Springer, pp. 799–814 (2014)

26. Prewitt, J.M.: Object enhancement and extraction. Picture Processing and Psychophotometrics 10(1), 15–19 (1970)

27. Marr, D., Hildreth, E.: Theory of edge detection. Proc. R. Soc. London, Ser. B 207(1167), 187–217 (1980)

28. Akbarinia, A., Parraga, C.A.: Feedback and surround modulated boundary detection. Int. J. Comput. Vision. 126, 1367–1380 (2018)

29. Lin, C., et al.: Improved contour detection model with spatial summation properties based on nonclassical receptive field. J. Electron. Imaging 25(4), 043018–043018 (2016).

30. Lin, C., Xu, G., Cao, Y.: Contour detection model using linear and nonlinear modulation based on non-CRF suppression. IET Image Proc. 12(6), 993–1003 (2018).

31. Dollar, P., Tu, Z., Belongie, S.: Supervised learning of edges and object boundaries. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1964–1971 (2006)

32. Ren, X.: Multi-scale improves boundary detection in natural images. In: Proceedings of the European Conference on Computer Vision, Springer, pp. 533–545 (2008)

33. Xiaofeng, R., Bo, L.: Discriminatively trained sparse code gradients for contour detection. In: Advances in Neural Information Processing Systems, pp. 584–592 (2012)

34. Khoreva, A., et al.: Weakly supervised object boundaries. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 183–192 (2016)

35. Ganin, Y., Lempitsky, V.: N4-fields: Neural network nearest neighbor fields for image transforms. In: Asian Conference on Computer Vision, Springer, pp. 536–551 (2014)

36. Kokkinos, I.: Pushing the boundaries of boundary detection using deep learning. In: International Conference on Representation Learning (2016)

37. Shi, W., et al.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1874–1883 (2016)

38. Yang, J., et al.: Object contour detection with a fully convolutional encoder-decoder network. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 193–202 (2016)

39. Xu, D., et al.: Learning deep structured multi-scale features using attention-gated crfs for contour prediction. In: Advances in Neural Information Processing Systems, pp. 3961–3970 (2017)

40. He, J., et al.: Bi-directional cascade network for perceptual edge detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3828–3837 (2019)

41. Deng, J., et al.: Imagenet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009)

42. Dollar, P., Zitnick, C.L.: Fast edge detection using structured forests. IEEE Trans. Pattern Anal. Mach. Intell. 37(8), 1558–1570 (2015).

43. Arbeláez, P., et al.: Multiscale combinatorial grouping. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 328–335 (2014)

44. Hallman, S., Fowlkes, C.C.: Oriented edge forests for boundary detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1732–1740 (2015)

45. Maninis, K.-K., et al.: Convolutional oriented boundaries: From image segmentation to high-level tasks. IEEE Trans. Pattern Anal. Mach. Intell. 27, 1676–1688 (2017)

46. Silberman, N., et al.: Indoor segmentation and support inference from RGBD images. Lect. Notes Comput. Sci. 7576 746–760 (2012)

47. Gupta, S., Arbelaez, P., Malik, J.: Perceptual organization and recognition of indoor scenes from RGB-D images. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp. 564–571 (2013)

48. Gupta, S., et al.: Learning rich features from RGB-D images for object detection and segmentation. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8695. Springer, Cham. https://doi.org/10.1007/978-3-319-10584-0_23.

How to cite this article: Xu G, Lin C, Cheng Y. Dense connection decoding network for crisp contour detection. IET Image Process. 2021;15:956–963. https://doi.org/10.1049/ipt2.12076