MSEdge: A Multi-Scale Edge Chain Detector

Anonymous cvm submission

Paper ID 19

Abstract

This paper presents a novel multi-scale edge chain detector (MSEdge) to robustly extract edge chains from images at different scales captured in a wide variety of scenes. In contrast to the traditional edge pixel based methods, the proposed algorithm is based on edge chains which leads to a robuster and more complete edge detection result than the traditional ones. Firstly, the edge chains are extracted and validated on a set of pyramid images that are obtained by resizing the original input image with different scales. Then, for each downsampled image, the edge chains on it are projected onto the original image to create a mask map. The multi-scale contrast of each pixel on the mask map is calculated and a soft non-maximum suppression method (Soft-NMS) is proposed to be applied on these pixels to get the edge pixels on the original image. Thirdly, the edge chains in different scales are merged to get a single edge map by a novel chain based merging procedure. Finally, the Guo-Hall thinning algorithm [13] and a simple connecting procedure are applied on the edge map to get the final single pixel width edge chains. Experimental results on several common used images and both the ROC dataset and the BSDS dataset sufficiently demonstrate the efficiency and robustness of the proposed MSEdge as a multi-scale edge chain detector by comparing with five state-of-the-art edge/boundary detectors.

1. Introduction

One of the most intensively studied problems in computer vision concerns with how to extract edge chains on an image for some advanced applications. Edge chains are one of the most widely used geometric structures which can be used to represent the silhouettes of an image. As a low level information of an image, edge chains can be applied on line segment detection [2, 12], object recognition [26], image segmentation [4], and so on.

The scale of an edge is an unavoidable issue since the very beginning of the studying on edge detection [6]. Studies on natural images have strongly suggested that the scale-invariance or the multi-scale structure is an intrinsic property of natural images. In general, fine scales are expected to provide spatially accurate results, but also to be particularly sensitive to noise. Edge chains detected in coarse scales are more robust against noise, textures and spurious edges, but tend to suffer from displacements of the edges from their actual position. According to the work of [18], the multi-scale edge detection methods can be classified into three categories based on "the way they manage the information obtained at different scales".

The first group detects the edges in different scales, and then applies some fusion procedures to get a single edge map [6, 18]. Edge tracking is one of the most widely used methods in the fusion procedure. In the pioneering work of [6], the edges were firstly detected using a high degree of smoothing on a coarse level of scale, and then a tracking procedure was applied to determine their precise locations over decreasing scales. Lopez-Molina *et al.* [18] used the same strategy which first sampled a finite set of images from the Gaussian Scale-Space, then the Sobel operator was applied on each of these images, and finally the edges were tracked from the coarse scales to the fine ones.

The second group collects edge cues and informations in different scales first and these multi-scale informations are then aggregated to discriminate the edge pixels [16, 23, 22] from an image. Training a classifier is the most widely used method in edge cues aggregation [23, 16]. In the work of [23], the local boundary cues including contrast, localization and relative contrast are collected in multi-scales, and then a classifier is trained to integrate them across scales. The detection of boundaries (edges) is formulated as a classification between boundary and non-boundary pixels. There are also other cues aggregation methods, for example, Özkan and Işık [22] applied the common vector approach (CVA) to aggregate the gradient maps computed at each scale to form a single gradient map on which a smart edge extraction procedure was performed to give an edge map. In [5], the responses of filters at adjacent scales were multiplied to enhance edge structures. Recently, Liu *et al.* [17] introduced a method to first detect the scales of edge pixels using 3D-Harris based on the informations in different scales, and then the edge segments were extracted based on the edge pixels and their scales.

The last one determines the scale to be used at each pixel or subregion based on the local characteristics [15, 11]. Jeong and Kim [15] defined an energy function over the continuous scale space, based on which the scale of each site in the image plane was determined via minimization. Elder and Zucker [11] defined the conception of "minimum reliable scale" (MRS) of an event as the minimum scale in which the event can be reliably detected by a certain operator. The MRS of each point on the image was locally calculated and applied in edge detection.

Boundary detection is a research field close to the edge detection, especially on the aspect of multi-scale edge detection. The only difference between them may be that many of the boundary detectors "focus on large-scale salient regions/boundaries and tend to ignore details on an image" [23]. Recently, with the development of machine learning, many boundary detection methods have been proposed [24, 7, 29], which have achieved great improvement over the traditional feature based methods.

Most of the former multi-scale detectors focus on the properties of edge pixel across scales, however these edge pixel based methods use only local information for edge detection, which suffers from the influence of noise. For example, a long edge chain may be split into many small fragments due to the influence of noise by the pixel based methods. In this paper, in contrast to the traditional edge pixel based methods, we propose a novel multi-scale edge chain based detector which can get a robuster and more complete edge detection result than the traditional ones. The proposed multi-scale edge chain detector (MSEdge) belongs to the above-mentioned first group, which detects edge chains in different scales, then merges, instead of tracking, those edge chains to form a single edge map, and finally applies the Guo-Hall thinning algorithm [13] on the edge map to get the single-pixel width edge chains.

2. Our Algorithm

Fig. 1 shows the framework of the proposed MSEdge detector, which consists four steps. Firstly, the input image is down-sampled with different scales to construct a set of pyramid images and the edge chains are detected and validated on these pyramid images, respectively. Then, we propose a soft non-maximum suppression method (Soft-NMS) to get the edge pixels from different pyramid images. Thirdly, the edge chains in different scales are merged into a single edge map by a novel chain based merging procedure. Finally, a morphological dilation method, named as the Guo-Hall thinning algorithm [13], and a simple edge chain connecting procedure are applied to get the final single pixel width edge chains.

2.1. Edge Chain Detection

2.1.1 Edge Chain Tracking

To track edge chains from the edge map, there are generally two methods. The one is performed on an edge map after non-maximum suppression (NMS), on which the edge chains are mostly in single pixel width. In this condition, the tracking procedure can be achieved easily by searching for the unprocessed edge pixels in the 8-neighbors of the current seed pixel. Another one is proposed in the work of [25] and named as Edge Drawing, which computes a set of anchor edge points on an image and then links these anchor points by drawing edges guided by gradient orientation. In this work, we apply the former method.

Given a gray image $\mathbf{I}$, firstly a 3×3 Gaussian filter with the standard deviation $\sigma = 1$ is applied to suppress noise and smooth out the image. Then the Canny operator with the low threshold and high threshold set as $(g_{\text{low}}, g_{\text{high}})$ is performed on the smoothed image to get an edge map $\mathbf{E}$. The edge pixels on $\mathbf{E}$ are roughly sorted in the descending order according to their gradient magnitudes and recorded in a set $\mathcal{P}$. After that, the foremost unprocessed edge pixel in $\mathcal{P}$ is selected as the initial seed pixel $\mathbf{p}_{seed}$. The 8-neighbors of the $\mathbf{p}_{seed}$ are searched, if there exists a 8-neighbor that is an unprocessed edge pixel, we consider it as the next seed pixel and add it into the current edge chain, and then begin 8-neighbors searching from this newly added pixel. The seed growing of the current edge chain is conducted iteratively until all the pixels on this chain is processed, and then we add the current edge chain into the edge chain set $\mathcal{C}$ and begin with another edge chain from the remaining pixels in $\mathcal{P}$.

2.1.2 Edge Chain Validation

There are many false detections in the edge chain set $\mathcal{C}$, thus an edge chain validation procedure is required to get rid of the false ones. Wang *et al*. [28] proposed the "supporting range" to distinguish those weak edge pixels from their surroundings and applied a segment-based hysteresis thresholding approach to verify the edge segments. In the work of [10], the use of the Helmholtz principle gives a new view on both boundary and edge chain validation, which achieves good performance. However the Helmholtz principle on edge chain validation is based on level lines of the image [10]. In the work of EDPF [3], the level lines are replaced with edge chains detected by the Edge Drawing [25] method for convenience, but no convincing proofs are given.

In this work, the edge chains that are weak on the low levels will be salient on the high levels with the increasing of scales, so no more specific strategy is needed to distinguish the weak edges from the noises. Thus, we can sim-

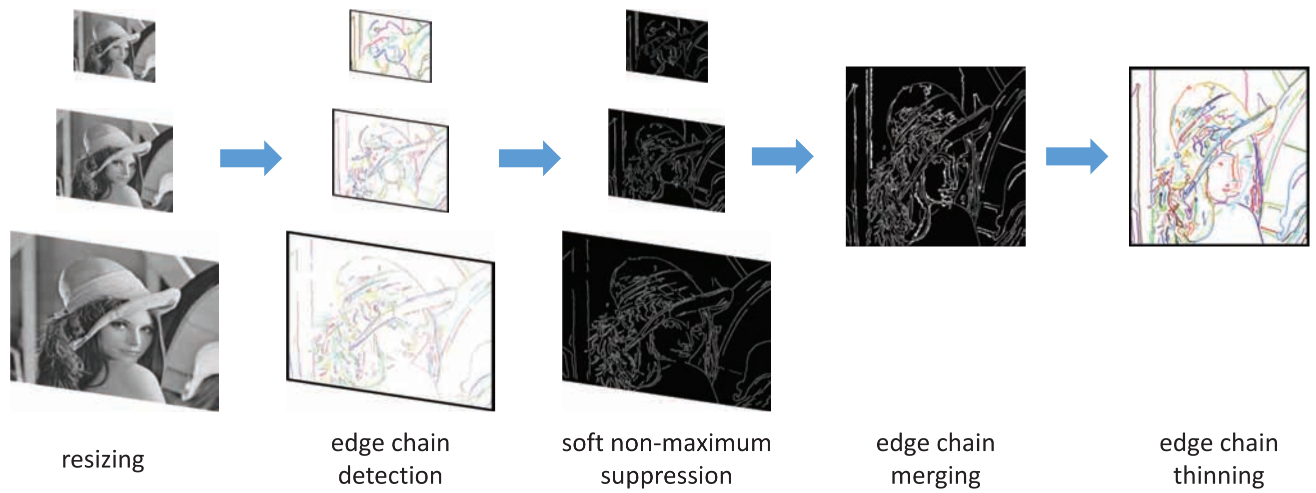


Figure 1. The framework of the proposed multi-scale edge chain detector.

plify the segment-based hysteresis thresholding approach proposed by Wang *et al.* [28], and use the summation of the gradients of all the pixels on the edge chain to measure the saliency of the edge chain. Given an image with a size of $w \times h$, the detected edge chain set is denoted as $\mathcal{C}$, the edge chain validation procedure is performed as follows. Firstly, the minimum edge chain length threshold l_{mm} can be easily defined as follows according to the work of LSD [27] and CannyLines [19]:

$$l_{\text{mm}} = -2.5 \log(M)/\log(p), \tag{1}$$

where $M = w \times h$ is the size of $\mathbf{I}$ and $p = 1/8$. Then, for each edge chain $\mathbf{C} \in \mathcal{C}$, whose length is denoted as $l(\mathbf{C})$, the summation of the gradients of all the pixels on $\mathbf{C}$ is calculated and denoted as $g(\mathbf{C})$, and the edge chain $\mathbf{C}$ is considered as salient enough to be kept if the following conditions are satisfied:

$$l(\mathbf{C}) \geq l_{\text{mm}} \;\; \&\& \;\; g(\mathbf{C}) \geq l_{\text{mm}} \times g_{\text{th}}, \tag{2}$$

where g_{th} is the predefined gradient threshold.

2.2. Multi-Scale Edge Chain Detection

Our proposed multi-scale edge chain detection method is a general framework which is suitable for all the edge chain detectors. There are four steps of the proposed multi-scale edge chain detector (MSEdge), including: edge chain detection, soft non-maximum suppression, edge chain merging and edge chain thinning.

2.2.1 Edge Chain Detection

For a given image with a size of $w \times h$, we reduce the size of the image by half with the increment of the level of scales, which means that the size of the image on the level n is $w/2^n \times h/2^n$, $n \in \{0, 1, 2, ..., N\}$, N is the highest level predefined, which is recommended as $N = 3$ for general applications. The set of these resized images is denoted as $\mathcal{I}$. Then for each image $\mathbf{I}_n \in \mathcal{I}$, the edge chains on it are detected by applying the edge chain detector introduced in Section 2.1. The set of edge chains on $\mathbf{I}_n$ is denoted as $\mathcal{C}_n$.

2.2.2 Soft Non-maximum Suppression

For each level n, the scale of this level is defined as $s_n = 2^n$, we try to get the corresponding edge pixels of the edge chains in $\mathcal{C}_n$ on the original image $\mathbf{I}_0$, which is performed as follows. Firstly, we traverse the image $\mathbf{I}_0$, for any pixel $\mathbf{p}_0 = (x_0, y_0)$ on $\mathbf{I}_0$ if its corresponding scaled pixel $\mathbf{p}_n = (x_0/s_n, y_0/s_n)$ on $\mathbf{I}_n$ is an edge pixel, we consider $\mathbf{p}_0$ to be an edge pixel hypothesis. For all these edge pixel hypotheses, a mask image with the same size of $\mathbf{I}_0$ is created. Then, for each pixel $\mathbf{p}_0$ on the mask image, its gradient orientation is defined as that of the scaled pixel $\mathbf{p}_n$. The reason is that the gradient orientation of $\mathbf{p}_n$ is more robust than that of $\mathbf{p}_0$. The gradient orientation is divided into 4 directions as the same in the traditional non-maximum suppression applied in the Canny operator [9]. After that, the image intensities of the pixels within a span equal to s_n on each side of $\mathbf{p}_0$ along the gradient direction are accumulated on the $\mathbf{I}_0$, we denote them as v_1 and v_2, respectively. The multi-scale contrast of $\mathbf{p}_0$ is then defined as $|v_1 - v_2|/s_n$. A contrast map is created to record the multi-scale contrasts of all the edge pixel hypotheses. Finally, a soft non-maximum suppression method (Soft-NMS) is proposed to be applied to get real edge pixels, which is conducted as follows. For each edge pixel hypothesis $\mathbf{p}_0^i$ on the mask image, whose contrast is c_i, we search the neighboring pixels from $\mathbf{p}_0^i$ on each side of its direction with a span equal to s_n. If for every neighboring pixel $\mathbf{p}_0^j$ whose contrast is c_j, and the following condition is always satisfied:

$$c_i > c_j - c_{\text{th}}, \quad (3)$$

where c_{th} is the contrast threshold of the soft non-maximum suppression, we consider $\mathbf{p}_0^i$ to be an edge pixel. Eq. (3) means that an edge pixel hypothesis $\mathbf{p}_0^i$ is considered as a real edge pixel if there is no other neighboring edge pixel hypothesis whose contrast is c_{th} greater than that of $\mathbf{p}_0^i$.

Fig. 2 is a comparison between the proposed Soft-NMS method and the traditional NMS method on a coarse edge. We can see that the edge pixels detected by the traditional NMS method tend to breaks into fragments while the Soft-NMS can keep the completeness of the edge chain well. In fact, it is difficult to detect a single pixel width edge chain on a coarse edge, because the neighboring pixels near the real position of the edge share very close intensity informations. With the influence of noise and quantization error, it is not stable to obtain sequential single pixel width edges via the traditional NMS method on a coarse edge. However, by setting a contrast threshold c_{th}, the proposed Soft-NMS method can achieve the goal of suppressing the non-edge pixels as well as keeping the completeness of the edges.

The soft non-maximum suppression procedure is applied on each level $n, n > 0$, while for the original (level 0) image there is no need to apply the Soft-NMS, so all the pixels of the edge chains on the level 0 are kept. Thus, for each level, an edge map, which is in the same size as $\mathbf{I}_0$, with each edge pixel labeled with the ID of the corresponding edge chain is obtained, the set of these edge maps is denoted as $\mathcal{E}$.

2.2.3 Edge Chain Merging

Fusing the edge detection result on multi scales to form a single edge map is a tough task, both the spatial accuracy and the completeness of the edges should be taken into consideration. Tracking across scales is a widely applied method, and usually the tracking is started from the edge pixels on the coarse scales to determine their precise locations over decreasing scales [18, 6]. However, the edge pixels based tracking procedure suffers from the influence of noise especially on the fine scales, for example, the tracking procedure may terminate in the local area due to noises. To solve this problem, more constraints on tracking are attached, which however makes the tracking procedure very complicated.

In contrast to those edge pixel based tracking methods, we propose a merging method which is based on the edge chains detected in different scales. We consider the multi-scale edge merging problem as a procedure of making up the edge map at level 0 with the edge chains on higher levels. Considering the fact that there are generally two types of making up categories: 1) *cover* the clutter area where exists multiple edge chain detections (type 1 in Fig. 3 (a)); 2) *fill* the gap between two consecutive edge chains (type 2 in Fig. 3 (a)), the proposed edge chain merging method is conducted as follows.

Firstly, a single edge map $\mathbf{E}_{\text{merged}}$ is created. All the edge chain pixels on $\mathbf{E}_0$ are kept unchanged in $\mathbf{E}_{\text{merged}}$, which means $\mathbf{E}_{\text{merged}} = \mathbf{E}_0$, initially.

Then, for each level n, $n > 0$, the corresponding edge chain set and the edge map on this level are denoted as $\mathcal{C}_n$ and $\mathbf{E}_n$, respectively. For each edge chain $\mathbf{C} \in \mathcal{C}_n$, we traverse each pixel in $\mathbf{C}$ sequentially from the beginning to the end to find the type of each pixel. For a pixel $\mathbf{p}_n$ on $\mathbf{C}$, the set of the corresponding pixels of $\mathbf{p}_n$ on the edge map $\mathbf{E}_n$ is denoted as $\mathcal{P}$. Then we search the pixels in $\mathcal{P}$ on the edge map $\mathbf{E}_{\text{merged}}$, and calculate the number of edge chains num by the ID values of edge chains recorded in $\mathbf{E}_{\text{merged}}$. The type of $\mathbf{p}_n$ can then be categorized into:

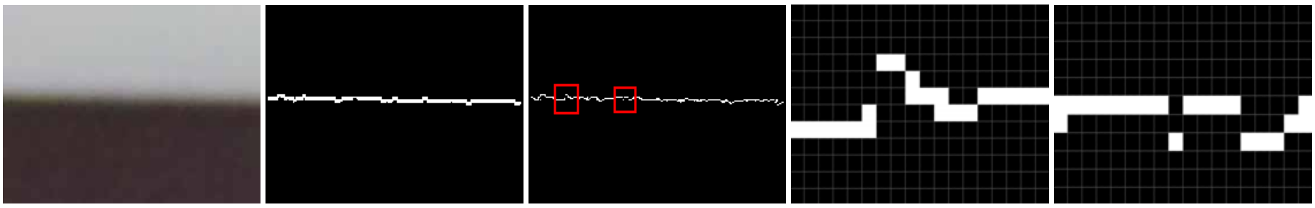
- **type 0**: if $num = 1$, which means there exists only one low level edge chain overlapped with $\mathbf{C}$, thus the low level edge chain is kept for precision.
- **type 1**: if $num \geq 2$, which means there exist more than one low level edge chains overlapped with $\mathbf{C}$, and there are chaos detections in $\mathcal{P}$, thus these pixels should be *covered* with the current edge chain $\mathbf{C}$.
- **type 2**: if $num = 0$, which means there exist no low level edge chain in $\mathcal{P}$, thus these pixels should be *filled* with the current edge chain $\mathbf{C}$.

With the categories of each pixel on $\mathbf{C}$, the intervals of these pixels that belong to type 1 or type 2 are founded, these intervals are extended one pixel on each terminal for better connection with the edge chains already recorded in $\mathbf{E}_{\text{merged}}$. For example, if there is an edge chain whose category list is $\{0, 0, 1, 1, 0, 0, 0, 2, 2, 0, 0, 0, 1, 1, 2, 2, 0, 0\}$, then three intervals [2,5], [7,10] and [12,17] will be discovered. For each interval, the corresponding edge pixels on $\mathbf{E}_n$ are found, then those pixels are recorded into the merged edge map $\mathbf{E}_{\text{merged}}$ with a new unique value of ID.

Fig. 3 illustrates the edge chain merging procedure on the level 0 and level 1. Fig. 3(a) shows the edge chains detected on the level 0, from which we can see that there exist many blank gaps on the coarse edges, while these gaps and the miss detections are well discovered on the level 1 as shown in Fig. 3(b), where the pixels in green stand for the ones made up from the edge chains on the level 1. Fig. 3(c) is a close look of the "type 1" rectangular area marked in Figs. 3(a) and (b), in which two edge chains (distinguished in red and blue) are considered to be chaos detections and thus covered by the green pixels from the level 1.

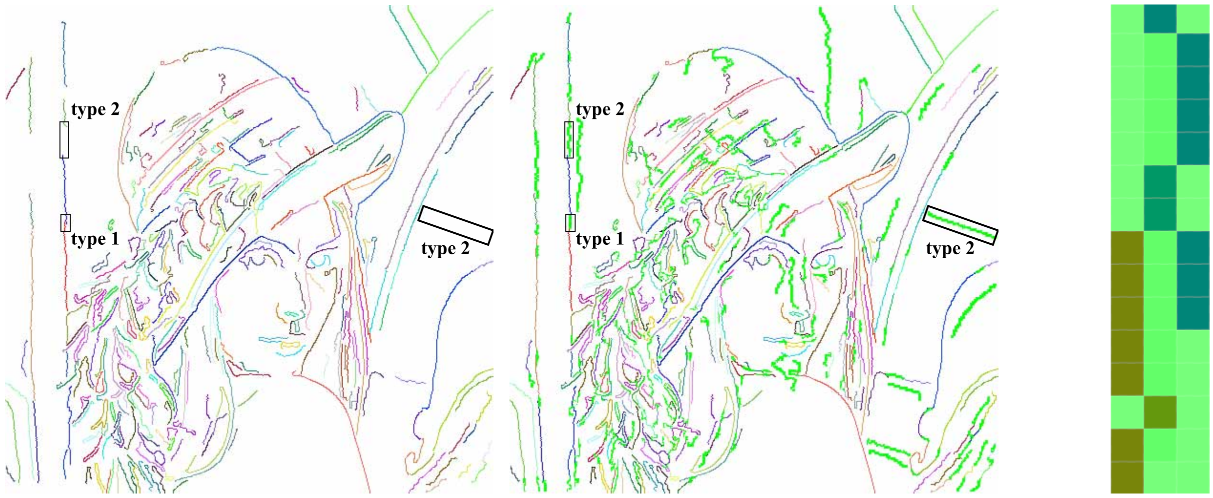
2.2.4 Edge Chain Thinning

The final output of the proposed MSEdge detector is a set of single pixel width edge chain detection. The result of the



(a) Image (b) Soft-NMS (c) NMS (d) NMS Detail (e) NMS Detail

Figure 2. Comparison between the proposed Soft-NMS method and the traditional NMS method. The last two figures (d) and (e) are the details of the rectangular areas in figure (c).



(a) level 0 (b) level 0 + 1 (c) details of type 1

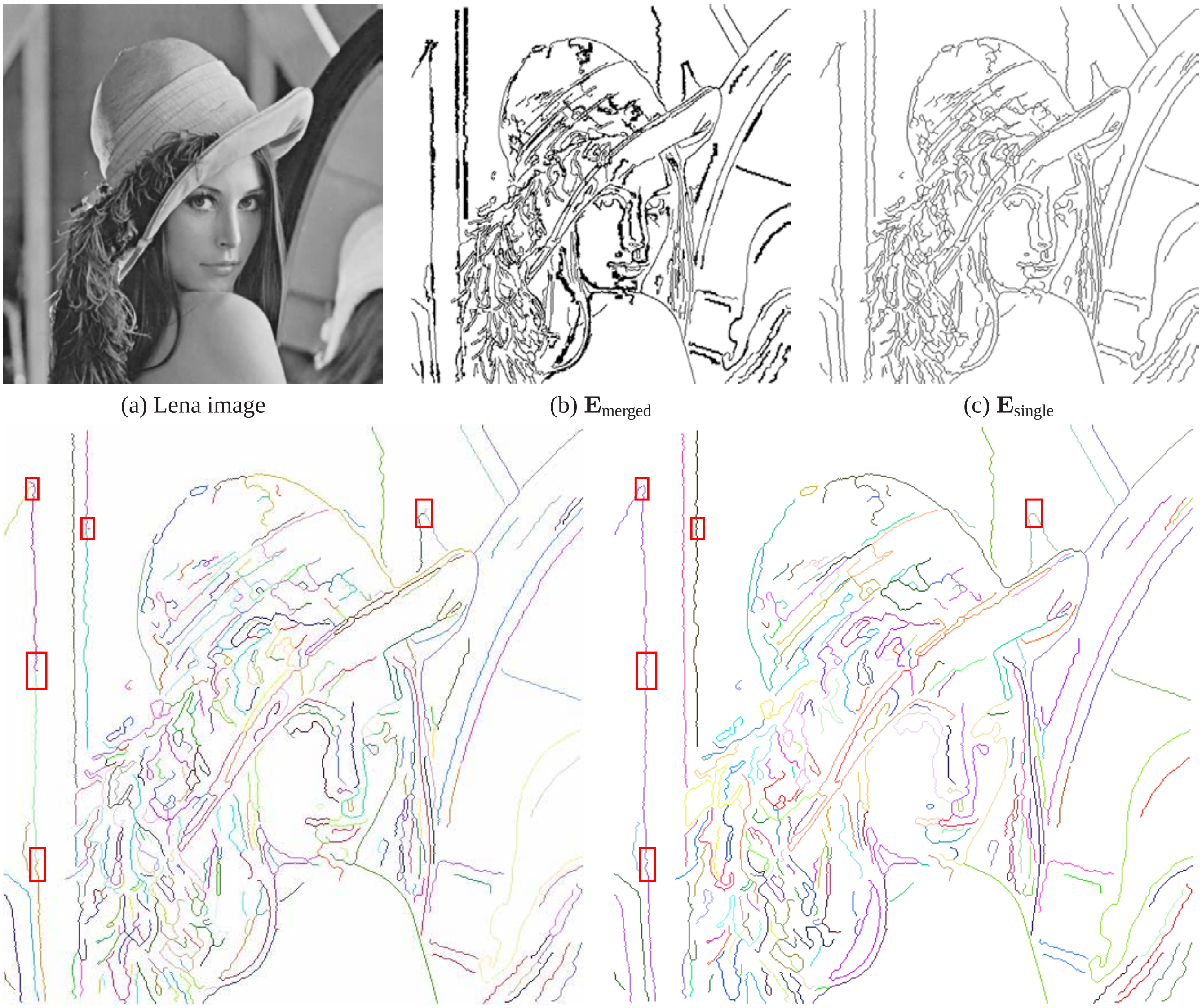
Figure 3. An illustration of the edge chain merging procedure on the level 0 and the level 1 of the Lena image.

edge chain merging is a single edge map $\mathbf{E}_{\text{merged}}$, as shown in Fig. 4(b), to obtain single pixel width edge chains from an edge map, tracking is the most widely used method [19, 25]. However, the 8-neighbors tracking [19] is not suitable for the edge pixel tracking on the multi-pixel width edge map $\mathbf{E}_{\text{merged}}$, while the Edge Drawing [25] also suffers from the problem of anchor points choosing which is also a tough issue on the coarse edges.

The edge map obtained by the edge chain merging from low levels to high levels preserves all the single pixel width edge chains, which will keep unchanged by applying a morphological dilation procedure. As to these pixels on the multi-pixel width edge, it is unnecessary to find out the strongest ones in them because the neighboring pixels near an edge pixel share very close edge responses, thus a morphological dilation procedure is also suitable for these pixels. Thus in this work a morphological dilation method named Guo-Hall thinning [13] is applied, which takes the 3×3 neighboring region of each pixel into consideration. Firstly, the north and east and then the south and west boundary pixels are alternatively deleted for locating the middle of the edge region as the edge chain pixel. Then, a thinning operator to one of two subfields is alternatively applied. The Guo-Hall thinning algorithm is iteratively performed until there is no more pixels dilated. In our work the iteration is around 2^N where N is the highest level predefined because there is little edge chain whose width is greater than 2^N. Generally, $N = 3$, the iteration is 8. In our work, only the edge pixels are examined at each iteration, which makes the edge chain thinning procedure very fast. The result of the Guo-Hall thinning is a single pixel width edge map, which is denoted as $\mathbf{E}_{\text{single}}$.

After applying Guo-Hall thinning, a single edge map $\mathbf{E}_{\text{single}}$ is obtained, as Fig. 4(c) shows, and then the single pixel width edge chains can be detected from $\mathbf{E}_{\text{single}}$ by applying the same tracking procedure as that introduced in Section 2.1.1. Fig. 4 (d) shows the original edge chains detected on the Lena image, we can observe in the rectangular areas that most detected edge chains are complete, but there still exits few fragment detections.

For further refinement of the detected edge chains, a simple edge chain connecting procedure is applied to get more complete edge chains, which is conducted as follows: for each edge chain $\mathbf{C}_i$, the 8-neighbors of the foremost 10 pixels on each terminal of $\mathbf{C}_i$ is searched, if there is a edge pixel $\mathbf{p}$ that belongs to another edge chain $\mathbf{C}_j$ is searched, and $\mathbf{p}$ is also one of the foremost 10 pixels on one terminal of $\mathbf{C}_j$, we consider $\mathbf{C}_i$ and $\mathbf{C}_j$ to be two connected edge



(a) Lena image (b) $\mathbf{E}_{\text{merged}}$ (c) $\mathbf{E}_{\text{single}}$

(d) Original edge chains (e) Edge chains after connection

Figure 4. An illustration of the edge chain thinning procedure applied on the Lena image.

chains and then merged them to form a new edge chain. Fig. 4 (e) shows the edge chains after the connection procedure, we can see that most of the fragmentary detections are connected quite well.

3. Experimental Results

To evaluate the performance of the proposed MSEdge, we tested it on both the ROC dataset [1] [8] and the BSDS dataset [2] [20]. The ROC dataset is made up of 60 images with ground truth edge chains, while the BSDS dataset consists of 300 images with labeled object boundaries. We use the ROC dataset and BSDS dataset to evaluate the performance of the proposed MSEdge as an edge chain detector and a boundary detector, respectively. The used measurements are FP, TP, F-score, the precision (P) and recall ratio (R) for the ROC dataset, and F-score, P and R for the BSDS dataset. Let DC be the set of edge pixels detected by a certain method, GT denotes that of the ground truth data, the precision (P) and recall ratio (R) are defined as follows:

$$P = \frac{\#\{\text{DC} \cap \text{GT}\}}{\#\{\text{GT}\}} \quad \text{and} \quad R = \frac{\#\{\text{DC} \cap \text{GT}\}}{\#\{\text{DC}\}}. \qquad (4)$$

The F-score is defined as $F = 2PR/(P+R)$.

3.1. Discussion on Parameters

There are two families of parameters to be set in the proposed method: (1) $(g_{\text{low}}, g_{\text{high}})$ for the Canny edge detec-

[1] Available at http://figment.csee.usf.edu/edge/roc/

[2] Available at https://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

tion and g_{th} for edge chain validation; (2) c_{th} for soft non-maximum suppression.

Tuning the two thresholds of Canny has been studied for many years [14], in this work we do not mean to go into this tough issue, because the edge chains detected in multiple scales are finally merged, so on each level, a Canny operator with $(g_{\text{low}}, g_{\text{high}}) = (30.0, 60.0)$ will be suitable enough to capture the salient edge chains for general applications. For specific cases, we also recommend to set $g_{\text{high}} = 2.0 \times g_{\text{low}}$, and adjust the value of g_{low} to get the best performance. The tunning of $(g_{\text{low}}, g_{\text{high}})$ leads to abrupt change of the edge detection result, while the adjusting of g_{th} for edge chain validation gives finer adjustment of the final result. Table 1 shows the average detection results on the ROC dataset with $(g_{\text{low}}, g_{\text{high}}) = (30.0, 60.0)$ and g_{th} varying from 20 to 120, where L_{avg} denotes the average length of all the edge chains. From Table 1, we can see that the TP, F-score and P are improved gradually with the increasing of g_{th}, while the recall ratio R is decreased on the contrary, which means that a high value of g_{th} leads to less but more meaningful detections. We can also see in Table 1 that $g_{\text{th}} = 60$ gives a good balance between precision and recall ratio. As a conclusion, for general cases, we recommend to set $(g_{\text{low}}, g_{\text{high}}, g_{\text{th}}) = (30.0, 60.0, 60.0)$.

The parameter c_{th} affects the width of the edge pixel area on the edge maps, and $c_{\text{th}} = 0$ means that only the local maximum pixel will be kept as an edge pixel, which coincides well with the definition of the traditional NMS. Table 2 shows the average detection results on the ROC dataset with $(g_{\text{low}}, g_{\text{high}}, g_{\text{th}}) = (30.0, 60.0, 60.0)$ and c_{th} varying from 0 to 11. From Table 2, we can see that $c_{\text{th}} = 0$ leads to a higher detection precision and F-score but a smaller value on the average edge chain length L_{avg}, which means that the Soft-NMS method can produce more complete detection results than the NMS method. We can also see in Table 2 that when $c_{\text{th}} \geq 5$, the increment of c_{th} gives little influence on the detection result, which means that $c_{\text{th}} = 5$ is good enough to make the edge pixels connected on the coarse edge, thus we recommend to set $c_{\text{th}} = 5$ for all the cases.

3.2. Comparison with State-of-the-Art Methods

To sufficiently evaluate the performance of our proposed MSEdge algorithm, we compared it with other five state-of-the-art edge detection methods, including: ED [25], EDPF [3], SREdge [28], a multi-scale edge detector SMED [5] and a boundary detector PS [21]. The source codes of ED and EDPF can be obtained from the Edge Drawing library [1], the source code of SREdge was implemented by us according to the original paper, and the source code of SMED is publicly available [3], that of the PS is available at the BSDS dataset [4].

3.2.1 Comparison On Benchmarks

We tested six algorithms on both the ROC dataset and the BSDS one. The ROC dataset can be used to evaluate the performance of the tested algorithms as an edge detector, while the BSDS dataset can be used to assess the quality of an algorithm on boundary detection. The parameters of the proposed MSEdge on the ROC dataset were set as those recommended in Section 3.1, while on the BSDS dataset, we set $g_{\text{th}} = 160$ to filter out the weak edges and keep the strong boundaries. The code of SMED requires square images as input, thus for each input image, we resized it into 500×500 for the ROC dataset and 400×400 for the BSDS dataset, and then applied the SMED method to obtain an edge map, the edge map was then resized back into the size of the original image, and the statistical measures were calculated on this resized edge map. Table 3 shows the average detection results of the six tested algorithms on both the ROC dataset and BSDS one. From Table 3, we can see that the SREdge method performs best on the ROC dataset with a F-score of 0.750, which is close to that of the SMED in the second place. The proposed MSEdge achieves a F-score of 0.726 which is close to the EDPF (0.726) and better than the ED (0.705) and the boundary detector PB (0.718). The reason is that the ground truth of the ROC dataset are salient edges, most of the coarse edges are not labeled out as ground truth, thus the edge detectors like SREdge, ED, EDPF works well in it. As to the BSDS dataset, we can observe that the boundary detector PB performs much better than other methods, and the proposed MSEdge ranks in the second place with EDPF. The F-scores of SREdge, SMED and ED are smaller than other methods due to their low precisions, which means that much false edges are detected by these methods on the natural images of the BSDS dataset. Fig. 5 shows the detection results of the six tested algorithms on two images of the ROC dataset and the BSDS dataset in vision, similar conclusion to that from Table 3 can be drawn from Fig. 5. Besides this we can also find that the proposed MSEdge can detect the coarse edges (marked in rectangles) much better than the other methods like EDPF and SMED. As a conclusion, in both datasets, the proposed MSEdge achieve moderate performances, which show the quality of the MSEdge method as both an edge chain detector and a boundary detector.

3.2.2 Comparison On Common Images

Besides the ROC dataset and the BSDS dataset, which can not represent the multi-scale characters of the images very well, we also tested the six algorithms on the Lena image

[3] Available at http://www4.comp.polyu.edu.hk/~cslzhang/code/ed.rar

[4] Available at https://www.eecs.berkeley.edu/Research/Projects/C

Table 1. Comparison of the MSEdge detection results on the ROC dataset with g_{th} varying from 20 to 120. $g_{\text{th}} = 60$ is recommended for general cases.

g_{th}	FP	TP	F-score	P	R	L_{avg}
20	0.035	0.552	0.683	0.552	0.945	59.914
40	0.026	0.576	0.701	0.576	0.941	61.442
60	**0.017**	**0.611**	**0.725**	**0.611**	**0.931**	**63.134**
80	0.015	0.638	0.739	0.638	0.915	64.693
100	0.012	0.662	0.750	0.662	0.897	66.129
120	0.011	0.685	0.757	0.685	0.878	67.478

Table 2. Comparison of the MSEdge detection results on the ROC dataset with c_{th} varying from 0 to 11. $c_{\text{th}} = 5$ is recommended for general cases.

c_{th}	FP	TP	F-score	P	R	L_{avg}
0	0.015	0.626	0.736	0.626	0.928	58.234
3	0.017	0.612	0.726	0.612	0.931	61.563
5	**0.017**	**0.611**	**0.726**	**0.611**	**0.932**	**63.126**
7	0.017	0.611	0.725	0.611	0.931	64.222
9	0.017	0.611	0.725	0.611	0.931	64.924
11	0.017	0.611	0.725	0.611	0.930	65.541

Table 3. Comparison of the six tested algorithms in the ROC dataset and BSDS dataset.

Dataset	ROC					BSDS		
Algorithms	FP	TP	F-score	P	R	F-score	P	R
MSEdge	0.017	0.611	0.726	0.611	0.932	0.550	0.480	0.640
ED [25]	0.020	0.576	0.705	0.576	0.967	0.520	0.430	0.670
EDPF [3]	0.016	0.600	0.726	0.600	0.954	0.570	0.480	0.690
SREdge [28]	0.015	0.625	**0.750**	0.635	0.955	0.530	0.420	0.700
SMED [5]	0.004	0.693	0.746	0.693	0.884	0.510	0.440	0.620
PB [21]	0.007	0.721	0.718	0.721	0.755	**0.610**	0.590	0.630

and two out-of-focus images. Considering the fact that the ED, EDPF and SREdge methods belong to the traditional single scale edge detector category, and both the SMED and PB are the multi-scale ones, thus in Fig. 6 only the EDPF and PB methods are compared with the proposed MSEdge method for simplification. We can see in Fig. 6 that unlike in the BSDS and ROC datasets, in this test the proposed MSEdge performs much better than other methods, with both the coarse edges and the details persevered quite well. The detection result of PB contains little fragments, but many edges are ignored by it because the PB determines the existence of edge pixel based on the informations in a neighbouring area of each pixel, thus it may be not suitable for the PB to be applied on the images with very coarse edges. The EDPF can also detects out most of the edges but there exist many fragments, also some faint edges are dismissed by EDPF. As a result, we can draw the conclusion that, as a simple edge chain detector, the proposed MSEdge can detect both the fine and coarse edges, and its performance is much better than other edge chain detectors, especially on the large size images.

3.3. Application on Line Segment Detection

Differing to the SMED and PB method which take an edge map as the final result, the output of the proposed MSEdge method is a set of single pixel width edge chains with consecutive pixels, which can be directly applied on the detection of line segments. To extract line segments from the edge chains, the same strategy as that used in the work of EDLines [2] is applied, which is conducted as follows. For each edge chain, we traverse the pixels in sequence, and fit line segments to the pixels via the Least Square Fitting method, once the perpendicular distance from the pixel to the current line segment exceeds a certain threshold, 1 pixel as we set, we generate a new line segment and continue this procedure until all the pixels on

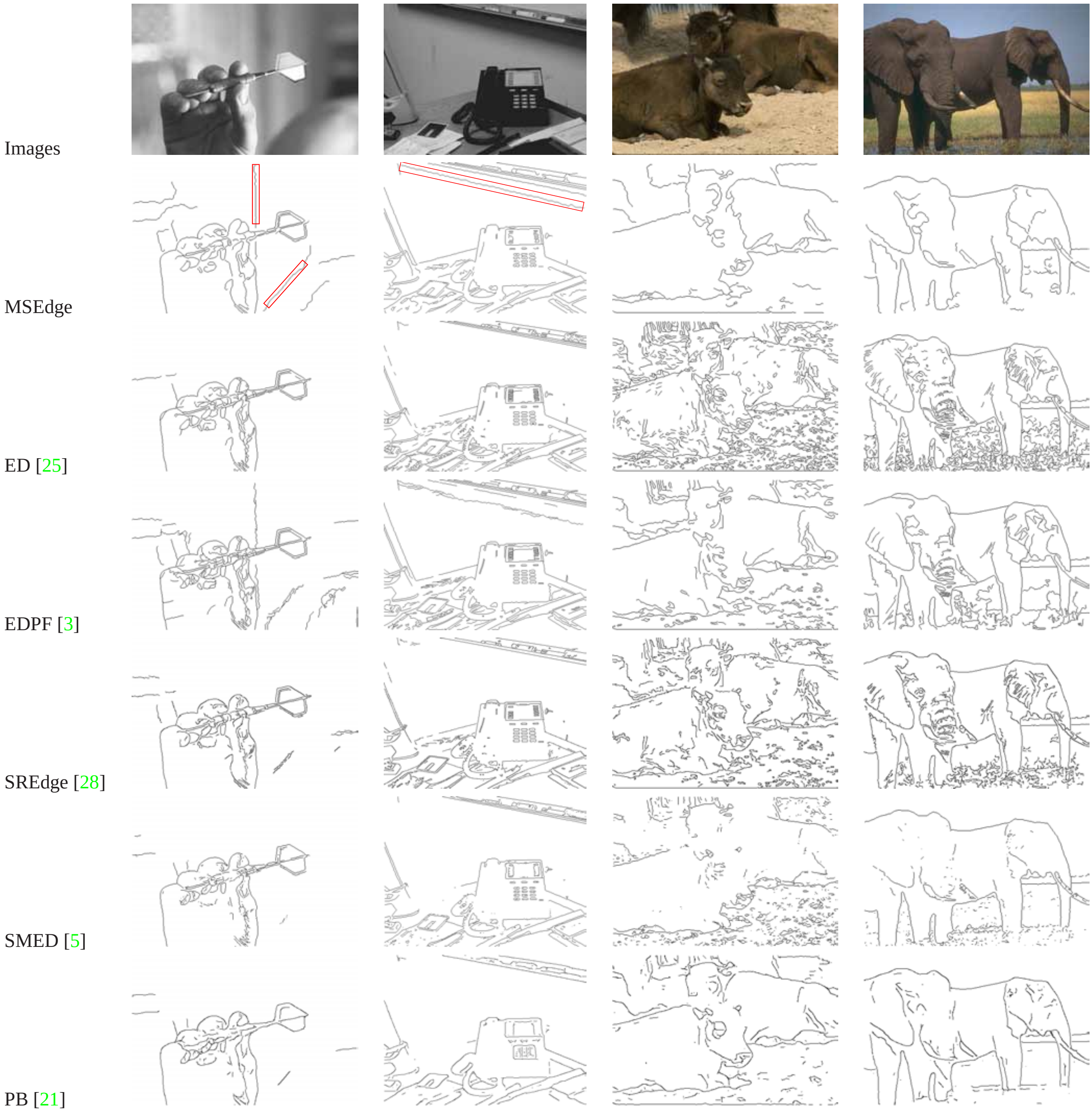


Figure 5. Edge detection results of the six tested algorithms on two images of the ROC dataset (the first and second columns) and the BSDS dataset (the third and fourth columns), respectively.

this edge chain are walked over. We denoted this line segment detector as MSLine, and Fig. 7 shows the comparison between the LSD [27] and the MSLine on three large size images. From the first column to the last one is the original image, the edge chains detected by the proposed MSEdge and its time consumption, the line segments detected by the MSLine and the LSD, respectively. We can see that the LSD detector fails on the coarse and blurred lines and a lot of fragmentary line segments exist, see the branches of the tree on the first row and the clouds on the second row. While the MSLine can recover almost all the line segments of the scene in all these three images, also the line segments extracted by MSLine are much longer and completer than those detected by the LSD. The time consumption of the

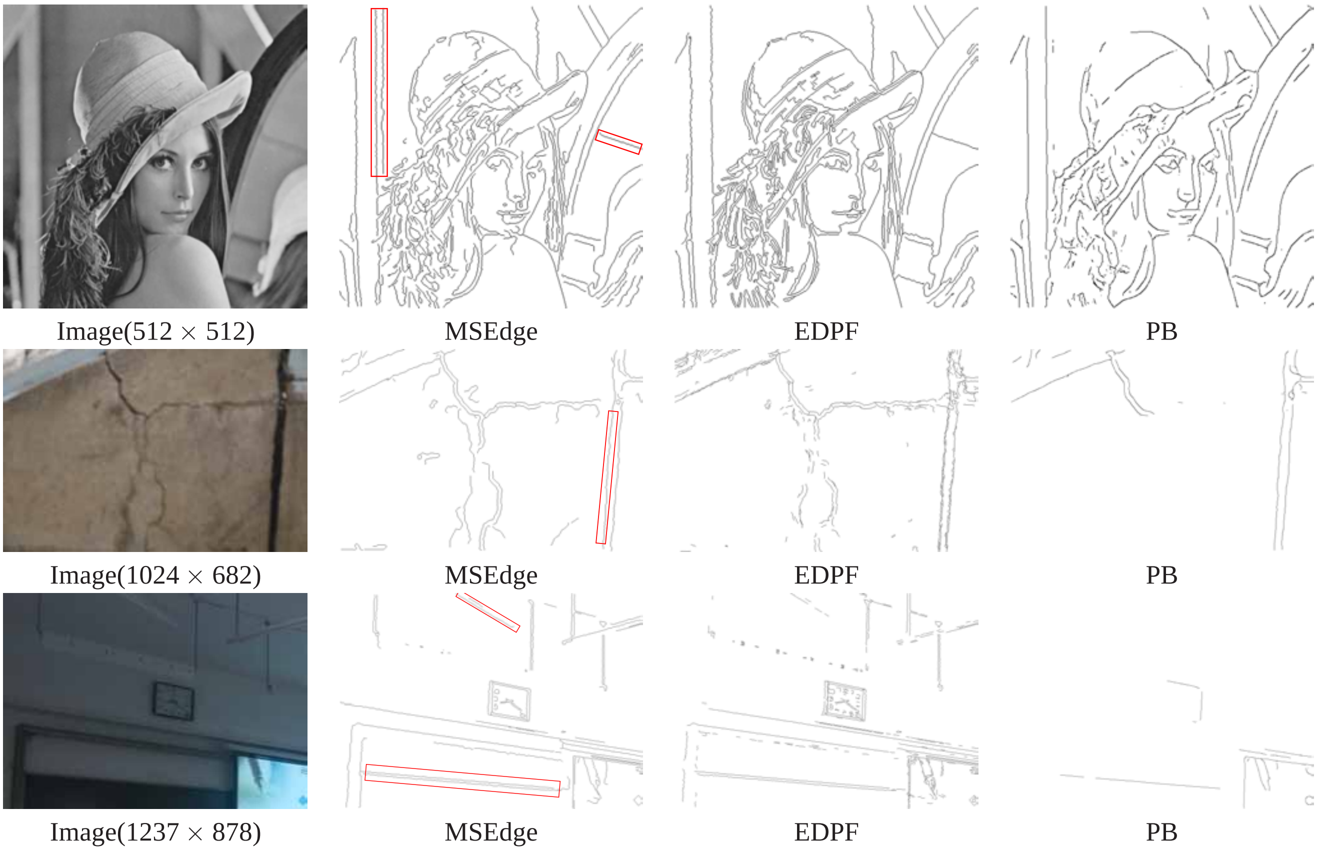


Figure 6. Edge detection results of the three tested algorithms on two out of focus images.

MSEdge on these large size images are around 1.6s on a computer with Intel Core i5-3550p CPU without any optimization, which means that there is a high potential for the MSEdge and MSLine to be applied on the real time tasks.

4. Conclusion

This paper presents a novel multi-scale edge chain detector (MSEdge) which can robustly extract edge chains from images at different scales, especially on the large size images. The proposed MSEdge is based on the edge chains detected and validated in different scales, which leads to a robuster and more complete edge detection result than the traditional edge pixel based ones. Experiments and comparison with other five edge/boundary detectors on the ROC dataset, the BSDS dataset and several common images sufficiently demonstrate the efficiency and robustness of the proposed MSEdge as a multi-scale edge chain detector.

References

[1] C. Akinlar and C. Topal. Edge Drawing Library. http://ceng.anadolu.edu.tr/cv/EdgeDrawing/. 7

[2] C. Akinlar and C. Topal. EDLines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642, 2011. 1, 8

[3] C. Akinlar and C. Topal. EDPF: A real-time parameter-free edge segment detector with a false detection control. *International Journal of Pattern Recognition and Artificial Intelligence*, 26(1):1255002, 2012. 2, 7, 8, 9

[4] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 1

[5] P. Bao, L. Zhang, and X. Wu. Canny edge detection enhancement by scale multiplication. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9):1485–1490, 2005. 1, 7, 8, 9

[6] F. Bergholm. Edge focusing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(6):726–741, 1987. 1, 4

[7] G. Bertasius, J. Shi, and L. Torresani. DeepEdge: A multi-scale bifurcated deep network for top-down contour detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 52, pages 4380–4389, 2015. 2

[8] K. Bowyer, C. Kranenburg, and S. Dougherty. Edge detector evaluation using empirical ROC curves. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 354–359, 1999. 6

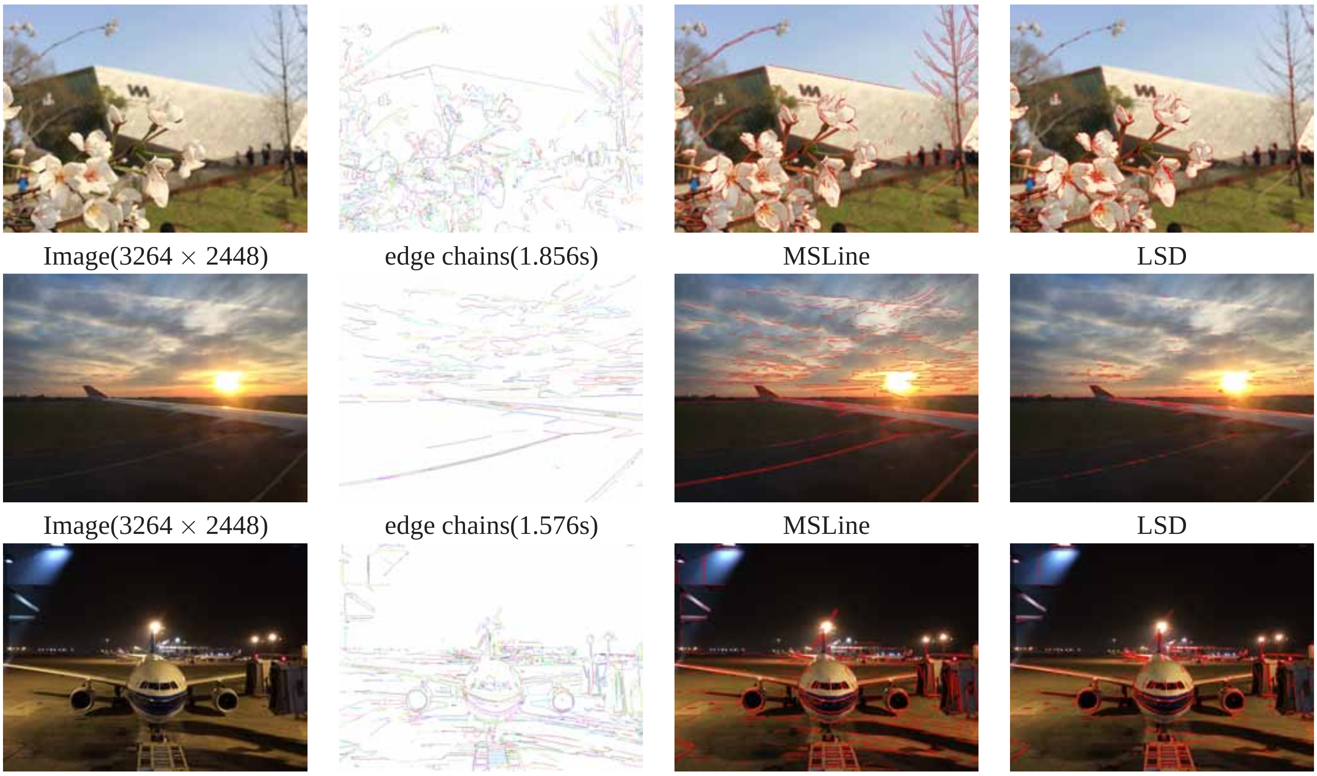


Figure 7. Comparison between the LSD and the line segments detector formed by combining the proposed MSEdge and the Least Square Fitting method on three large size images captured by a smart phone.

[9] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. 3

[10] A. Desolneux, L. Moisan, and J.-M. Morel. Edge detection by Helmholtz principle. *Journal of Mathematical Imaging and Vision*, 14(3):271–284, 2001. 2

[11] J. H. Elder and S. W. Zucker. Local scale control for edge detection and blur estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(7):699–716, 1998. 2

[12] L. A. Fernandes and M. M. Oliveira. Real-time line detection through an improved Hough transform voting scheme. *Pattern Recognition*, 41(1):299–314, 2008. 1

[13] Z. Guo and R. W. Hall. Parallel thinning with two-subiteration algorithms. *Communications of the ACM*, 32(3):359–373, 1989. 1, 2, 5

[14] Y.-K. Huo, G. Wei, Y.-D. Zhang, and L.-N. Wu. An adaptive threshold for the canny operator of edge detection. In *IEEE International Conference on Image Analysis and Signal Processing (IASP)*, pages 371–374, 2010. 7

[15] H. Jeong and C. Kim. Adaptive determination of filter scales for edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(5):579–585, 1992. 2

[16] O. Laligant, J. Miteran, P. Gorria, et al. Merging system for multiscale edge detection. *Optical Engineering*, 44(3):833–834, 2005. 1

[17] X.-M. Liu, C. Wang, H. Yao, and L. Zhang. The scale of edges. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 462–469, 2012. 1

[18] C. Lopez-Molina, B. De Baets, H. Bustince, J. Sanz, and E. Barrenechea. Multiscale edge detection based on gaussian smoothing and edge tracking. *Knowledge-Based Systems*, 44(1):101–111, 2013. 1, 4

[19] X. Lu, J. Yao, K. Li, and L. Li. CannyLines: A parameter-free line segment detector. In *IEEE International Conference on Image Processing (ICIP)*, 2015. 3, 5

[20] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 416–423, 2001. 6

[21] D. R. Martin, C. C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004. 7, 8, 9

[22] K. Özkan and Ş. Işık. A novel multi-scale and multi-expert edge detector based on common vector approach. *AEU-International Journal of Electronics and Communications*, 69(9):1272–1281, 2015. 1

[23] X. Ren. Multi-scale improves boundary detection in natural images. In *Computer Vision–ECCV 2008*, pages 533–545. Springer, 2008. 1, 2

[24] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. DeepContour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3982–3991, 2015. 2

[25] C. Topal and C. Akinlar. Edge Drawing: A combined real-time edge and segment detector. *Journal of Visual Communication and Image Representation*, 23(6):862C–872, 2012. 2, 5, 7, 8, 9

[26] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, 1991. 1

[27] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(4):722–732, 2010. 3, 9

[28] L. Wang, S. You, and U. Neumann. Supporting range and segment-based hysteresis thresholding in edge detection. In *IEEE International Conference on Image Processing (ICIP)*, 2008. 2, 3, 7, 8, 9

[29] S. Xie and Z. Tu. Holistically-nested edge detection. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1395–1403, 2015. 2