Efficient and High-quality Ellipse Detection via Implicitly Excluding Most Useless Arc Groups and Enhancing Arc detection

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Detecting ellipses from images is an fundamental problem in computer vision and pattern recognition, and plays an important role in many applications. This paper presents a new edge-link method for efficient and high-quality ellipse detection, where the two steps of edge-link methods are improved by our two presented novel measures respectively. The first is to adaptively adjust the search direction in linking edge pixels to generate arcs as consistently as possible. The second is to develop a novel measure for grouping arcs to check whether these arcs are from a same ellipse, which is by employing a grid to manage the arcs and designing a traversal path to visit grid cells continuously, through which most useless arc groups can be implicitly excluded for efficiency. This is different from existing methods that need explicitly check all possible arc groups. Based on these measures, we design an algorithm to detect ellipses as many as possible. Experimental results show that we can significantly improve both the accuracy and efficiency of ellipse detection, much superior to existing methods. Thus, we can significantly improve many applications.

1 Introduction

Ellipse detection is an important task in image processing, and required in many applications such as industrial inspection [1], medical image analysis [2], autonomous driving [3], and robot vision [4]. With regard to this, the edge-link methods [5, 6, 7, 8, 9, 10, 11] are prominent due to their efficiency and effectiveness, to be discussed in Sec. 2. These methods work by first extracting arcs with continuous edge pixels and then checking arcs in groups whether they are co-elliptical, called *ellipse checks*. The arcs from a same ellipse are called *co-elliptical* ones, and they are used to generate an ellipse.

Arc groups are always in a large number, so that arc grouping for ellipse checks dominates the efficiency. Considering that most arc groups are composed of arcs from different ellipses, which cannot be used to generate ellipses, called *useless groups*, many methods have been proposed to employ cheap calculations to exclude useless groups as soon as possible for efficiency, such as constraining the search region for arc grouping [5], leveraging convex hulls to group [9], building an adjacent matrix to represent the grouping relationships between arcs [8], and excluding many useless groups by constraints from characteristic mapping [12] or the Candy's theorem [13]. Even so, any arc group should be checked once and this still wastes much time on useless groups.

In this paper, we address the challenge of implicitly excluding most useless groups for efficiency. It is by using a grid to manage arcs and then visiting grid cells orderly from the center outward. For a visited grid cell, each arc contained in the grid cell tries to find other arcs for arc grouping in its

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

	[8]	[7]	[14]	[10]	[15]	[12]	Ours
#TP	4	4	4	3	0	6	**7**
#FP	2	3	0	1	1	4	**0**
#FN	3	3	3	4	7	1	**0**
Time(ms)	7.86	8.12	15.28	11.24	52.23	7.07	**5.94**

Figure 1: Our method can obtain more accurate results than the state-of-the-art methods and cost less time, as illustrated for the example here. The detected ellipses are marked in red for the True Positive and green for the False Positive.

constrained search region [5]. This corresponds to have grid cells paired for arc grouping. As the constrained search region of an arc does not cover all grid cells in general, and a visited grid would not be processed again after it is visited, this would have many grid cells not paired for arc grouping, meaning their related useless arc groups are implicitly excluded. This will be discussed in detail in Sec. 3.

We also present a measure to generate arcs as consistently as possible, by which ellipses can be more effectively detected. This is by adaptively adjusting the search direction to link continuous edge pixels to generate contours, to be discussed in Sec. 4.1.

Based on our two novel measures, we develop an algorithm to detect ellipses as many as possible, where all formed arc groups are further checked by existing methods [9, 12] to exclude many more useless groups and finally co-elliptical arc groups are used to generate their corresponding ellipses with existing methods. As a result, we can detect many more ellipses and in a higher efficiency and a higher quality than existing methods, as illustrated in Fig. 1 and demonstrated in Sec. 5. Benefited from our improvements, many applications can be significantly promoted, as illustrated in Appendix C.

2 Related work

Ellipse detection methods can be coarsely classified into three categories: Hough transform based methods, edge-link based methods, and learning methods. Hough transform based methods [16, 17, 18, 19] take the ellipse detection task as a peak-finding process in a parametric voting space and use the Hough transform on pixels for a solution. Unfortunately, they are expensive and prone to incur incorrect results due to the complicated backgrounds and the lack of effective verification [20].

Recently, some learning methods [21, 22, 23, 24, 15, 25] have been proposed for ellipse detection. However, their potentials are limited by the difficulty of collecting high quality data for training, and they are always inefficient as they need learn a lot of features for a complex model, as shown in Fig. 1 and Table 3 for the result of [15].

Till now, edge-link based methods are prominent for ellipse detection [5, 6, 7, 8, 9, 10, 26, 11]. They link discrete edge pixels into arcs for ellipse detection, where local continuity information of contours can be well exploited to suppress interference from outliers and noise, and therefore increasing the detection accuracy. In the following, we have edge-link methods discussed briefly by their three sub-tasks, arc generation, arc grouping and ellipse checks.

For arc generation, Kim *et al.* [27] extract short straight line segments to approximate arcs, Prasad *et al.* [5] use curvature and convexity to extract smooth elliptic arcs, and there are two methods proposed for better corner detection to promote arc extraction, the adaptive Ramer Douglas Peucker (RDP) algorithm [28, 29] and a curvature-based method [30]. In implementing our method, we use the adaptive RDP algorithm [28] to divide contours into arcs because it need not frequent parameter adjustments. Based on the method of [8], Wang *et al.* [31] proposed a contrast-guided measure to enhance the extraction of arcs, but the improvement in detection capability is limited.

For arc grouping, some constraints are proposed to quickly exclude useless groups using simple computation, including arc-aware search regions [5], quadrant constraints [6], projection invariant

Table 1: Statistics about the ablation tests. The number of checked groups and time cost per image are the average results for all images in a dataset, where time refers to the total time cost on detecting ellipses in an image, including arc extraction, arc grouping and ellipse checks.

Datasets	Grouping via only arc-search regions		Our arc grouping measure		Implicit excluding rate
	Time(ms)	Checked groups	Time(ms)	Checked groups	
Prasad	8.44	18	4.09	4	(18-4)/18=77.8%
Prasad+	21.52	54	6.61	14	(54-14)/54=74.1%
Random	24.42	62	7.76	16	(62-16)/62=74.2%
Smartphone	58.82	231	11.81	26	(231-26)/231=88.7%

pruning [7], arc-support regions [14], characteristic mapping [12], the Candy's theorem [13] and coherent chord computation [11]. As useless groups always take a very large portion of all possible groups, these measures still take much time and prevent efficiency promotion. There are also some data structures studied for improving ellipse detection by collecting the arcs that are very possibly co-elliptical, including undirected graphs [9] and disjoint-set forests [10]. Even so, they need to enumerate possible groups, and this still need check a large amount of useless groups.

To check whether an ellipse is valid, a commonly used criterion is the ratio of inliers, defined as the proportion of arc points that fit the ellipse well [6, 7, 10]. When the ratio is high, it means the estimated ellipse is consistent with arcs. Other criteria include gradient consistency [10] and the completeness of ellipse [5, 7], which can filter out bad ellipses, but may prevent detection of imperfect ellipses in images. In our implementation, we use the measure of [9] for valid check of ellipses.

Different from existing edge-link methods, we present an arc grouping method to implicitly exclude most useless groups, where arc-aware search regions [5] are used for grouping arcs that are possibly co-elliptical. To our knowledge, this is the first method that can implicitly exclude useless arc groups. Our method is orthogonal to existing methods and so easy to be integrated with them for improving ellipse detection. For example, the useless groups that are not implicitly excluded by our method can be further quickly excluded by characteristic mapping constrains [12]. As for arc generation, we will mainly use the measures of [9] but replace its strategy for contour extraction, where an adaptive strategy is developed to extract contours as smooth as possible for generating arcs consistently.

3 Grid-based arc grouping

Our measure for arc grouping is by using a grid to manage the arcs and then visiting grid cells by a traversing path, through which arcs are grouped for ellipse detection. In the following, we first introduce the steps of our measure and then discuss their implementation and the effectiveness on implicitly excluding useless groups. With an ablation study by four data sets, it is known that we can greatly reduce the arc groups to be checked in comparison with only using arc-search regions [5] for arc grouping, as listed in Table 1. This shows we can implicitly exclude most useless arc groups.

The steps of our measure are as follows. Firstly, a grid is generated by the bounding box for all extracted arcs. Then, arcs are recorded in the grid cells that contain or intersect with them. Finally, a traversing path is designed to visit cells sequentially from the center outwards gradually, by which each arc in the currently visited cell is taken as an active one to search for possible co-elliptical arcs (called *inactive arcs*) in its improved arc-search region (to be discussed in Sec. 3.2) for arc grouping. As illustrated in Fig. 2, the active arc R_3 finds its inactive arc R_4 in its arc-search region in red to form a group. In this way, all possible co-elliptical arcs can be grouped. In summary, the algorithm for our arc grouping method is given in Alg. 1.

3.1 Grid resolutions

Clearly, the grid resolutions have much influence on the detection efficiency. A lower grid resolution means fewer cells, so that a cell would be larger to contain more arcs and prevent efficiency. In contrast, a higher grid resolution will lead to smaller cells containing fewer arcs, but this will generate more cells, also preventing efficiency. As an ideal expectation, if the arcs are evenly distributed in the grid cells for each grid cell to contain only one arc, the number of grid cells would not be large and a grid cell contains the fewest arcs, which would have ellipse detection in a high efficiency.

Algorithm 1 Arc grouping for ellipse detection

Input: Arcs: $R = \{r_i\}_{i=1}^n$, search regions: $\{\Omega_i\}_{i=1}^n$
Output: Arc groups: F

```
Define GC as the cell being processed;
Define IC as the set of VISITED cells;
Initialize GC as the central cell;
Mark all arcs as NOT_Active_USED;
while GC is not ∅ do
    for arc r_i ∈ GC that is NOT_Active_USED do
        for arc r_j ∈ Ω_i\IC that is NOT_Active_USED do
            if r_i ∈ Ω_j then
                Append arc group < r_i, r_j > to F;
            end if
        end for
        Mark r_i as USED;
    end for
    Add GC to IC;
    Let GC be the next cell by the search order;
end while
```

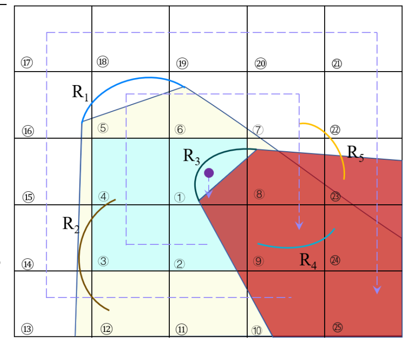


Figure 2: Our measure for arc grouping by orderly traversing the grid cells from the center outward, as marked by purple polylines with arrows.

Thus, we determine the grid resolution, NC_x and NC_y along the two axes, by Eq. 1,

$$\begin{aligned} NC_x &= \left\lfloor r_a \cdot \sqrt{N_{arcs}} \right\rfloor \\ NC_y &= \left\lfloor \frac{\sqrt{N_{arcs}}}{r_a} \right\rfloor \end{aligned} \tag{1}$$

where N_{arcs} is the number of arcs, and $r_a = \frac{\text{image\_height}}{\text{image\_width}}$ is the aspect ratio of the image.

With an investigation by many tests, such a grid resolution can always obtain good results and they are used in our implementation. Of course, arcs are generally in various lengths and distributed unevenly, which may influence the grid resolution in achieving high efficiency. As a future issue, we will further study these influences to optimize the grid resolution for high efficiency.

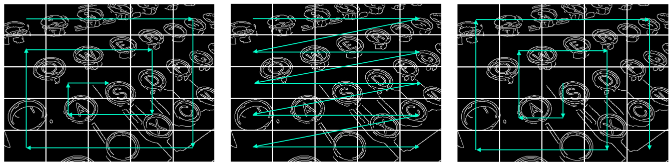
3.2 Improved arc-search regions

As discussed by Prasad *et al.* [5], an arc can only find its co-elliptical inactive arcs in a region, called an *arc-search region*. The arc-search region of an arc is bounded by the line connecting the two endpoints of the arc and two ray lines that are from its two endpoints and tangent to the arc, as illustrated by the red region for R_3 in Fig. 2. As we take the arcs of the visited grid cells each as active ones to find all their respective co-elliptical arcs, the visited grid cells would not be investigated in the following checks. Thus, our arc-search region for an arc should exclude the grid cells that have been visited. As illustrated in Fig. 2, the arc-search region of R_1 in the cell ⑤ should exclude the grid cells ①, ②, ③ and ④, as the cell ⑤ containing R_1 is visited after these cells. Clearly, this reduces the arc-search region of R_1 and implicitly exclude the arc group of R_1 and R_3. Such a reduced arc-search is called an *improved arc-search region*, as illustrated by the yellow region for R_1, which excludes the light green cells ①, ②, ③ and ④.

3.3 Traversing paths

For implicitly excluding useless groups as many as possible, we design a traversing path to visit grid cells from center outwards gradually. This is based on the following considerations:

- In general, active arcs nearer to the center of the grid often have smaller arc-search regions than those farther away from the center, e.g., the arc-search region of R_3 is smaller than that of R_1 in Fig. 2. Thus, first checking the arcs nearer the center of the grid can more effectively avoid checking useless groups, as a smaller arc-search region is less possible to have useless groups. Of course, it is also possible for an arc near the boundary of the grid to have a smaller arc-search region when it is towards the outside. However, such cases seldom occur, and so would not interfere with our efficiency.



traversing path	(a)	(b)	(c)
#arc groups	189	156	81

(a) Outside inward (b) Zig-Zag (c) Ours (d) Statistics

Figure 3: Comparison of the generated arc groups between using different traversing paths for the extracted arcs of the image in Fig. 10(c). The table shows the number of collected arc groups for different traversing paths.

- With such a traversing path, the grid cells far away from the center would have their arc-search regions improved, as discussed in Sec. 3.2. This is helpful for efficiency promotion. Otherwise, when the grid cells far away from the center are visited first, their arc-search region would be less improved, causing many useless groups generated. As illustrated in Fig. 2, if the grid cell containing R_1 is visited first, its arc-search region would be larger to include the light green cells, and so generating more useless groups.

As an investigation, we tested other traversing paths like the path from the outside inward and a zig-zag path, as illustrated in Fig. 3. The results show that using our path can generate much fewer arc groups than using the other paths. This shows the advantages of our designed traversing path for implicit exclusion of useless groups.

4 Improved ellipse detection

With our arc grouping measure, we present a new edge-link method for ellipses detection, where we mainly use the corresponding measures of [9] for arc generation and ellipse checks, and then take a new strategy for extracting ellipses as many as possible. The pipeline of our method is still by the steps for edge-link methods, extracting arcs, grouping arcs for ellipse checks and generating ellipses for co-elliptical arcs, as illustrated in Appendix A. For a complete introduction of our method for ellipse detection, we will first introduce the corresponding measures of [9] for arc generation, arc grouping and ellipse checks, and then discuss our improvements and our final algorithm for ellipse detection. Our improvements are as follows:

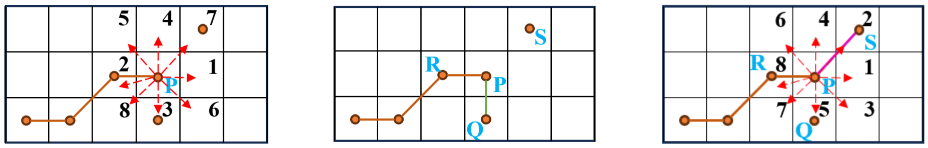
- In arc extraction, we present a novel measure to improve contour extraction for generating arcs more consistently than using the corresponding measure of [9].
- In arc grouping, our developed method in Sec. 3 is used, by which most useless groups can be implicitly excluded. Then, the collected arc groups could be further filtered by the characteristic mapping constraints [12] to more effectively obtain useful arc groups for ellipse checks.
- In ellipse generation, we take another strategy that first generates ellipse candidates as many as possible and then removes the redundant ones. Thus, ellipses can be detected many more than existing methods.

4.1 Arc extraction

The arc extraction measure of [9] includes the following five steps.

Edge detection. The Canny's algorithm [32] is used to detect the edge pixels. For obtaining high quality edges, Gaussian filtering with small kernels is applied to smooth out noise and bilateral filtering is applied to smooth out textures.

Contour extraction. Continuous edge pixels are collected to obtain contour curves. Here, we develop an adaptive measure to extract contours as smoothly as possible for generating arcs as consistently as possible for helping ellipse detection, to be discussed later in this section.



(a) (b) (c)

Figure 4: (a) The measure of [9] for contour generation is by starting from a seed edge pixel to extend gradually in a depth first search, and in a fixed search order of the left, right, down, up, up left, down right, up right, down left, as represented by the dashed arrow lines with numbers. (b) Shen *et al.* [9] may likely generate a very curved contour by extending from P to Q, not to S. (c) Our changed search order in extending a contour is by the angle difference from the last search order, as illustrated by the dashed arrow lines with numbers. Thus, we have P extended to S, not Q by the direction from R to P.

Contour segmentation. A contour may be composed of arcs from different ellipses. Thus, a contour should be segmented for arc extraction, which is by finding corner points, whose curvatures change abruptly in comparison with their respective neighboring points. More details are given in Appendix A.

Arc determination. The very short or very flat contour segments cannot be arcs of ellipses. They should be removed, and so the remained contour segments are the extracted arcs. Shen *et al.* [9] treat an arc as valid only when its length L satisfies $L > L_{arc}$, and the aspect ratio $B < B_{arc}$, where L_{arc} and B_{arc} are thresholds. Aspect ratio $B = \frac{box\_width}{box\_height}$ is used to describe the degree of flatness of the arc, where box_width and box_height are the longer side and the shorter side of the rotated rectangle with the minimum area bounding the arc.

In the above steps of [9] for arc extraction, there are some parameters. For the thresholds of these parameters to achieve good results, we set them by investigating the tested data sets, as done in existing methods [5, 7, 8, 30]. In our tests, we set $\theta_{arc} = 49°$, $L_{arc} = 52$ and $B_{arc} = 29$.

With the above steps for arc extraction, we can obtain many more arcs for detecting ellipses as many as possible, especially those overlapped ones. This is superior to many methods like the learning based method [25], which mainly extracts the arcs on the outer contours of objects and so would miss many overlapped ellipses, as shown in Appendix B.

4.1.1 Improvement on contour extraction.

Contour extraction is to connect the edge pixels by the neighboring relationships between them to generate edges. Shen *et al.* [9] extracts an contour by randomly selecting an unused edge pixel as a seed to search for neighboring edge pixels iteratively until the contour cannot be extended, where the depth first search is used. After all edge pixels are used, it means all contours are extracted. As illustrated in Fig. 4(a), starting from the left most yellow pixel, a contour is generated. In the depth first search of [9], the search order is fixed as shown by the dashed lines with numbers in Fig. 4(a). With such a search order, from pixel P, the contour will be next connected to pixel Q, not to pixel S, as illustrated in Fig. 4(b). Thus, pixel P will be taken as a corner point in contour segmentation to generate shorter arcs.

For generating arcs as long as possible for improving ellipse detection, we change the search order for extending a contour as smooth as possible, which is by the angle difference from the search direction of the last extension. The neighboring pixels with the smaller angel difference will be searched more preferentially. As illustrated in Fig. 4(c), according to the last search direction from pixel R to P, our search order for extending the contour from P is determined by the changed ordered numbers. Thus, the contour is extended from P to S.

The measure of [9] for contour generation is by starting from a seed edge pixel with a depth first search. When the seed edge pixel is at the middle of an arc, this is unsuitable for extracting the arc completely. Considering this, we have two directions searched from each seed edge pixel for generating arcs as long as possible. For example, if pixel R is selected as a seed, the contour can be generated by search along two directions from R, as shown in Fig. 4(c).

4.2 Arc grouping with valid checks

With the obtained arcs, we first use our method in Sec. 3 to collect arc groups. Afterwards, for the collected arc groups, we could use the characteristic mapping constrains [12] for a further exclusion of useless groups. At last, the remained arc groups are used for ellipse generation.

For a collected arc group, its arcs are used for generating an ellipse, where we mainly use the corresponding measures of [9]. At first, an ellipse is estimated for them by the Least-Squares fitting method. Then, it is checked whether the estimated ellipse is valid. The valid estimated ellipses are our detected ellipses.

For valid checks, it is by the measure of [9] using the ratios of inliers, which are computed by the following equation:

$$S(e) = \frac{1}{|g^*|} \sum_{p \in g^*} Ind(dist(p, e) < \varepsilon) \tag{2}$$

where $g*$ is the set of arcs in a group, p is an arc point, e is the estimated ellipse, $dist(p, e)$ is the algebraic distance from point p to the estimated ellipse e, and $Ind(.)$ refers to the indicator function.

When $S(e)$ has a high value, it means the estimated ellipse is valid. For this, a threshold S_{arc} is used for such a determination. By the suggestion of Shen *et al.* [9], we set $S_{arc} = 0.73$ in our tests, and always obtain good results.

4.3 Our algorithm for ellipse detection

With the measures discussed in the above subsections, we design an algorithm to detect ellipses as many as possible. Here, we generate candidates as many as possible and then remove the redundant ones, as discussed in the following.

Generating ellipse candidates. Our ellipse candidate generation is by the following steps. Firstly, an ellipse is estimated for each arc, as an arc may form an ellipse itself. Here, the estimated ellipse with its $S(e)$ less than S_{arc}, is discarded. Secondly, an ellipse is estimated for any a pair of arcs, where one arc is active and the other is one of its inactive arcs. It is by investigating the possibly estimated ellipses for pairs $(arc_{i,active}, arc_{j,inactive}), j = 1, 2, 3, \cdots$, where $arc_{j,inactive}$ are the inactive arcs in the improved arc-search region of the active arc $arc_{i,active}$, and remaining the ones whose $S(e)$ value is greater than S_{arc}. In this way, if many arcs are co-elliptical, any two of them are used for ellipse estimation respectively. Clearly, this may cause redundant ellipses, but would not miss ellipses.

Removing redundant ellipses. We use two measures to remove redundant ellipses. Firstly, we guarantee that an arc can be used only once for ellipse detection. We queue up ellipse candidates by their $S(e)$ values from the highest to the lowest, and iteratively select the ellipse with the highest $S(e)$ values from the candidates which contains only unused arcs. Secondly, we merge similar ellipse candidates with the corresponding measure by [9], which computes a weighted L_2 difference between the ellipse parameters.

5 Results and discussion

To verify the effectiveness and efficiency of our method, we conducted extensive experimental studies and collected results on a personal computer installed with an Intel(R) Core i7-8700 CPU@3.2GHz and 48GB RAM, where we have a comparison with the state-of-the-art methods [5, 6, 7, 14, 8, 9, 10, 15, 12, 31]. Their source codes can be obtained from the internet except for the code of [31]. For the method of [31], we implemented it by ourselves. Prasad *et al.*[5] have their codes implemented in Matlab, Lu *et al.*[14] implemented in Matlab and C++, Wang *et al.*[15] implemented in Python, and the other methods implemented in C++. All methods run on the CPU except for [15], which runs on GPU GTX1080Ti.

Datasets. In our tests, we used four synthetic datasets for testing our effectiveness on ellipse detection and four real-world datasets for comparing with existing methods. The used synthetic datasets

Table 2: The test results of the compared methods on the four synthetic datasets. P, R and F represent for precision, recall and F-measure, respectively. Here, the values for the metrics are the averaged ones for an image in a dataset, and the best results and the second best results are marked in red and yellow respectively.

Method	Occlusion [5]			Overlapping [5]			Concentric [8]			Concurrent [8]		
	P↑	R↑	F↑	P↑	R↑	F↑	P↑	R↑	F↑	P↑	R↑	F↑
[14]	0.4889	0.4559	0.4685	0.6024	0.5287	0.5231	0.6627	0.8546	0.7465	0.6635	0.8392	0.7411
[8]	0.5558	0.1774	0.2492	0.4910	0.2680	0.3462	0.7428	0.6692	0.7041	0.7727	0.7340	0.7528
[9]	0.5955	0.4587	0.5174	0.6048	0.4267	0.4686	0.8742	0.8435	0.8586	0.8193	0.9135	0.8638
[10]	0.4441	0.1350	0.2009	0.7238	0.3874	0.4498	0.8095	0.8446	0.8267	0.6996	0.9337	0.7999
[15]	0.0863	0.0280	0.0422	0.0934	0.0249	0.0366	0.0310	0.0096	0.0147	0.1386	0.0622	0.0859
Ours	0.7074	0.5558	0.6191	0.6773	0.4827	0.5282	0.9117	0.8860	0.8987	0.8737	0.9430	0.9070

Table 3: The average F-measure and time cost of the compared methods on the four real-world datasets. The best and the second best results are marked in red and yellow respectively. *Ours refers to using our method with relaxing constraints on arc generation. Ours+CM refers to checking our selected arc groups by characteristic mapping [12] before they are sent for ellipse checks.

Method	F-measure↑				Time(ms)↓			
	Prasad	Prasad+	Random	Smartphone	Prasad	Prasad+	Random	Smartphone
[14]	0.5092	0.6540	0.6009	0.6403	162.70	550.49	640.23	1118.08
[8]	0.4293	0.5539	0.4997	0.5510	3.75	7.78	9.71	14.66
[9]	0.4265	0.5713	0.5838	0.6424	7.96	14.18	17.48	25.20
[10]	0.3552	0.4851	0.6022	0.6825	6.60	10.15	15.97	24.53
[15]	0.3866	0.4648	0.5559	0.5246	56.47	55.65	54.48	55.35
[12]	0.3425	0.5198	0.5144	0.5000	3.95	6.94	9.32	12.41
[31]	0.4332	0.5618	0.5104	0.5629	4.07	7.96	10.75	17.73
Ours	0.4632	0.6012	0.6106	0.7006	4.09	6.61	7.76	11.81
*Ours	0.5126	0.6589	0.5898	0.6108	5.95	10.01	13.49	24.10
Ours+CM	0.4381	0.5742	0.5815	0.6689	3.82	6.11	7.37	11.12

include the Occlusion/Overlapping dataset [5] and Concentric/Concurrent dataset [8]. The tested four real-world datasets are Prasad/Prasad+ dataset [5], and the Random/Smartphone dataset [6].

Evaluation metrics. Here, we use Precision, Recall and F-measure to evaluate the performance of an ellipse detector over a specific dataset.IoU is used to evaluate the similarity between a detected ellipse with an ellipse of ground truth.

5.1 Accuracy

We made tests on synthetic and real-world datasets, as discussed below. We also made tests on generated ellipses with various shapes, orientations and sizes in the supplementary materials, showing our superiority over existing methods.

Tests on the synthetic datasets. The results of the compared methods for the four synthetic datasets are shown in Table 2. Clearly, our method can always achieve the best results except the one for Precision metric on the Overlapping dataset, where Jiang *et al.*[10] has the best Precision result. This is because it is very rigorous in selecting ellipse candidates and reduces the ellipses to be generated in the cases when there are many overlapped ellipses, by which its Precision value is high. However, it would miss many true ellipses so that its Recall value is low. Some visualization are provided in the appendices.

Tests on the real-world datasets. The statistics on the real-world datasets are given in Table 3, where we use IoU $= 0.8$ as the threshold to validate ellipses, as suggested by [7]. For the details about comparison on detection performance with various setting of IoU, please see Appendix B in the supplementary materials. From the statistics in Table 3, it is known that our method always achieve the highest F-measure values than existing methods except for that Lu *et al.*[14] achieves better F-measure values than ours on Prasad and Prasad+. This is because the images of these two datasets are of low pixel resolutions, so that our arc generation with Gaussian filtering may have some elliptical arcs missed. When the images are of high pixel resolutions, as those in the datasets Random

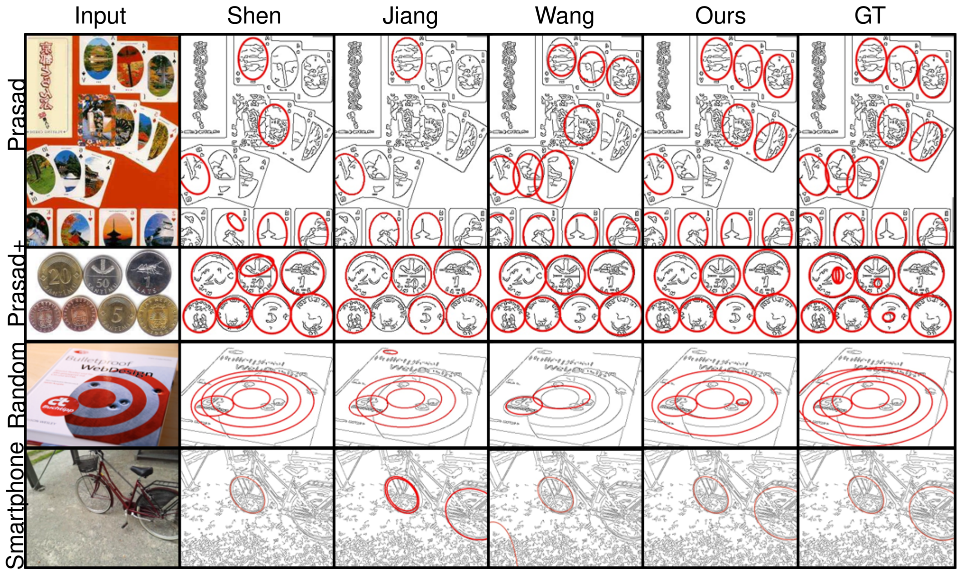


Figure 5: Some detection results of the methods in comparison on real-world datasets. We can detect more ellipses than the others, and in a higher quality.

and Smartphone, our method can obtain better results than Lu's method. For further verification, we made tests with relaxing constraints on arc generation (referred to as *Ours in Table 3), where we have better F-measure values than [14] on Prasad and Prasad+ datasets. Overall, we can always obtain many more accurate ellipses while producing fewer wrong ellipses than existing methods, as illustrated in Fig. 1 and Fig. 5.

5.2 Efficiency

We made tests on the four real-world datasets to check our efficiency on ellipse detection. By the statistic data in Table 3, it is known that ours can be faster than existing methods except in handling the Prasad dataset, in which each image has very few ellipses, leading the generated matrices for the method of [8] to detect ellipses very small, so that [8] is the fastest in handling this dataset. This also makes [12] faster than ours. As for the other cases that have many ellipses in the image, ours is faster than them, especially ours+CM, which is by combining ours with the characteristics mapping constraints [12]. This shows our higher performance than existing methods.

6 Conclusions

Edge-link methods are prominent for ellipse detection. In this paper, we presented two novel measures to improve edge-linking methods, one for generating arcs more consistently and the other for saving a large amount of computation by implicitly excluding most useless arc groups. Meanwhile, we develop an algorithm to detect ellipses as many as possible by checking whether an arc or any a group of arcs can form an ellipse. Experimental results show that we can more efficiently detect ellipses, while obtaining many more ellipses and in a higher quality, than existing methods.

Limitation. Our method is dependent on arc extraction. When arcs are sufficiently detected, their corresponding ellipses can be almost detected by our method. Though we improve arc extraction, some arcs may be still missed to prevent ellipse detection, especially in handling the overlapped ellipses. As a future issue, arc extraction would be seriously investigated.

References

[1] Zhen Wu, Fan Chen, Guoyuan Liang, Yimin Zhou, Xinyu Wu, and Wei Feng. Accurate localization of defective circular pcb mark based on sub-pixel edge detection and least square fitting. In *IEEE Data Driven Control and Learning Systems Conference (DDCLS)*, pages 465–470, 2019.

[2] Jiancong Chen, Yingying Zhang, Jingyi Wang, Xiaoxue Zhou, Yihua He, and Tong Zhang. Ellipsenet: anchor-free ellipse detection for automatic cardiac biometrics in fetal echocardiography. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part VII 24*, pages 218–227. Springer, 2021.

[3] Manal El Baz, Taher Zaki, and Hassan Douzi. Detection of elliptical traffic signs. In *International Conference on Image and Signal Processing*, pages 254–261. Springer, 2020.

[4] Huixu Dong, Ehsan Asadi, Guangbin Sun, Dilip K Prasad, and I-Ming Chen. Real-time robotic manipulation of cylindrical objects in dynamic scenarios through elliptic shape primitives. *IEEE Transactions on Robotics*, 35(1):95–113, 2018.

[5] Dilip K Prasad, Maylor KH Leung, and Siu-Yeung Cho. Edge curvature and convexity based ellipse detection method. *Pattern Recognition*, 45(9):3204–3221, 2012.

[6] Michele Fornaciari, Andrea Prati, and Rita Cucchiara. A fast and effective ellipse detector for embedded vision applications. *Pattern Recognition*, 47(11):3693–3708, 2014.

[7] Qi Jia, Xin Fan, Zhongxuan Luo, Lianbo Song, and Tie Qiu. A fast ellipse detector using projective invariant pruning. *IEEE Trans. Image Process.*, 26(8):3665–3679, 2017.

[8] Cai Meng, Zhaoxi Li, Xiangzhi Bai, and Fugen Zhou. Arc adjacency matrix-based fast ellipse detection. *IEEE Trans. Image Process.*, 29:4406–4420, 2020.

[9] Zeyu Shen, Mingyang Zhao, Xiaohong Jia, Yuan Liang, Lubin Fan, and Dong-Ming Yan. Combining convex hull and directed graph for fast and accurate ellipse detection. *Graphical Models*, 116:101110, 2021.

[10] Jingen Jiang, Mingyang Zhao, Zeyu Shen, and Dong-Ming Yan. EDSF: Fast and accurate ellipse detection via disjoint-set forest. In *Int. Conf. Multimedia and Expo*, pages 1–6. IEEE, 2022.

[11] Mingyang Zhao, Xiaohong Jia, Lei Ma, Li-Ming Hu, and Dong-Ming Yan. Coherent chord computation and cross ratio for accurate ellipse detection. *Pattern Recognition*, 146:109983, 2024.

[12] Qi Jia, Xin Fan, Yang Yang, Xuxu Liu, Zhongxuan Luo, Qian Wang, Xinchen Zhou, and Longin Jan Latecki. Characteristic mapping for ellipse detection acceleration. *IEEE Trans. Image Process.*, 2023.

[13] Xiujun Fang, Enzheng Zhang, Bingchen Li, and Bin Zhai. A fast and high-precision ellipse detection method based on the candy's theorem. *IEEE Access*, 2023.

[14] Changsheng Lu, Siyu Xia, Ming Shao, and Yun Fu. Arc-support line segments revisited: An efficient high-quality ellipse detection. *IEEE Trans. Image Process.*, 29:768–781, 2019.

[15] Tianhao Wang, Changsheng Lu, Ming Shao, Xiaohui Yuan, and Siyu Xia. ElDet: An anchor-free general ellipse object detector. In *Asian Conf. Comput. Vis.*, pages 2580–2595, December 2022.

[16] Nahum Kiryati, Yuval Eldar, and Alfred M Bruckstein. A probabilistic hough transform. *Pattern Recognition*, 24(4):303–316, 1991.

[17] Robert A McLaughlin. Randomized hough transform: improved ellipse detection with comparison. *Pattern Recognition Letters*, 19(3-4):299–305, 1998.

[18] Wei Lu and Jinglu Tan. Detection of incomplete ellipse in images with strong noise by iterative randomized hough transform (IRHT). *Pattern Recognition*, 41(4):1268–1279, 2008.

[19] Yi Tang and Sargur N Srihari. Ellipse detection using sampling constraints. In *IEEE Int. Conf. Image Process.*, pages 1045–1048. IEEE, 2011.

[20] Mingyang Zhao, Xiaohong Jia, and Dong-Ming Yan. An occlusion-resistant circle detector using inscribed triangles. *Pattern Recognition*, 109:107588, 2021.

[21] Wenbo Dong, Pravakar Roy, Cheng Peng, and Volkan Isler. Ellipse R-CNN: Learning to infer elliptical object from clustering and occlusion. *IEEE Trans. Image Process.*, 30:2193–2206, 2021.

[22] Ziteng Cui, Weiwei Guo, Zenghui Zhang, Huiyuan Chen, and Wenxian Yu. Ellipse-FCN: Oil tanks detection from remote sensing images with fully convolution network. In *IEEE International Geoscience and Remote Sensing Symposium*, pages 2855–2858. IEEE, 2020.

[23] Chicheng Liu, Rui Chen, Ken Chen, and Jing Xu. Ellipse detection using the edges extracted by deep learning. *Machine Vision and Applications*, 33(4):1–17, 2022.

[24] Feng Li, Bin He, Gang Li, Zhipeng Wang, and Rong Jiang. Shape-biased ellipse detection network with auxiliary task. *IEEE Transactions on Instrumentation and Measurement*, 71:1–13, 2022.

[25] Zhuo Su, Wenzhe Liu, Zitong Yu, Dewen Hu, Qing Liao, Qi Tian, Matti Pietikäinen, and Li Liu. Pixel difference networks for efficient edge detection. In *Int. Conf. Comput. Vis.*, pages 5117–5127, 2021.

[26] Yang Su, Baojiang Zhong, Zikai Wang, and Kai-Kuang Ma. Ellipse detection based on structure-preserving anisotropic edge extraction. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3900–3904. IEEE, 2024.

[27] Euijin Kim, Miki Haseyama, and Hideo Kitajima. Fast and robust ellipse extraction from complicated images. In *Proceedings of IEEE Information Technology and Applications*, pages 1–6. IEEE, 2002.

[28] Dilip K Prasad, Chai Quek, Maylor KH Leung, and Siu-Yeung Cho. A parameter independent line fitting method. In *The First Asian Conference on Pattern Recognition*, pages 441–445. IEEE, 2011.

[29] Dilip K Prasad, Maylor KH Leung, and Chai Quek. Ellifit: an unconstrained, non-iterative, least squares based geometric ellipse fitting method. *Pattern Recognition*, 46(5):1449–1465, 2013.

[30] Zepeng Wang, Derong Chen, Jiulu Gong, and Changyuan Wang. Fast high-precision ellipse detection method. *Pattern Recognition*, 111:107741, 2021.

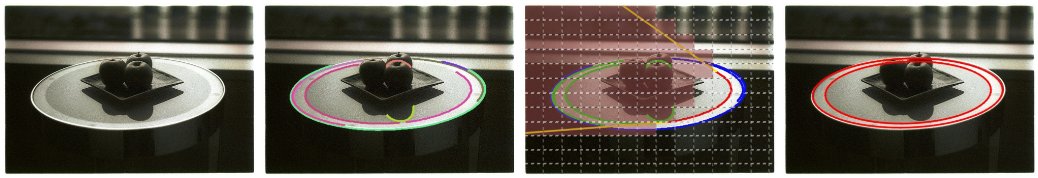
[31] Zikai Wang, Baojiang Zhong, and Kai-Kuang Ma. Ellipse detection based on contrast-guided arc enhancement. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9376–9380. IEEE, 2024.

[32] John Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

[33] G. Griffin, AD. Holub, and P. Perona. The Caltech 256. Technical report, Caltech Technical Report, 2006.

A Some details for implementation of our method

Pipeline of ellipse detection. The pipeline of our method for ellipse detection follows the standard workflow of all edge-linking methods, as illustrated in Fig. 6. For an input image, we first extract the elliptic arcs, which involves edge detection, contour extraction and splitting. Then, we group the arcs that are likely from a same ellipse, where we use our grid-base method to implicitly eliminate most useless groups. Finally, we generate a candidate ellipse for each group, and apply further checks to remove redundant ones.



(a) (b) (c) (d)

Figure 6: The pipeline for edge-link based ellipse detection. (a) The input image. (b) Extracting arcs. (c) Grouping arcs for ellipse checks. (d) Generating ellipses for co-elliptical arcs.

Contour segmentation. After extraction of curves, we subdivide the curves into elliptic arcs using corner points. The corner point can have an abrupt change in the magnitude of curvature, or be related to direction bending. Here, we provide an explicit example of this procedure. As shown in Fig. 7, we calculate the angle θ_i between the connected straight lines, and when θ_i is greater than a threshold θ_{arc} or its sign is different from θ_{i-1}, we mark the point as a corner point. We have $\theta_{arc} = 46°$ in our experiments as suggested by [9].

Removing redundant ellipses. For the evaluation of redundant ellipses, we use the measure by [9] to compute the difference between two candidate ellipses, say e_i and e_j, $\mathit{Diff}\,(e_i, e_j)$, in the following formula,

$$\begin{aligned}\mathit{Diff}\;(e_i, e_j) = & |x_i - x_j|^2 + |y_i - y_j|^2 \\ & + |a_i - a_j|^2 + |b_i - b_j|^2 + k_\theta \cdot \delta\theta\end{aligned} \tag{3}$$

where (x_*, y_*) are the center coordinates of these two ellipses, a_* and b_* are the lengths of the semi-major axis and semi-minor axis, respectively, $\delta\theta$ is the angle between the two semi-major axes of these two ellipses, and $k_\theta = \min\left\{\frac{a_i - b_i}{a_i + b_i}, \frac{a_j - b_j}{a_j + b_j}\right\}$ is used to attenuate the effect of $\delta\theta$ on $\mathit{Diff}\,(e_i, e_j)$ when one of the two ellipses is close to the circle. When two ellipses have a very low $\mathit{Diff}\,(e_i, e_j)$, meaning they are very similar and regarded as redundant ones. Here, we use a threshold $Th_d = 9.8$ for determining redundant ellipses, as suggested by [9].

B Additional experimental results

Details of used datasets. We test on four synthetic datasets, the Occlusion dataset and the Overlapping dataset proposed in [5], and the Concentric dataset and the Concurrent dataset constructed by Meng *et al.*[8]. The Occlusion dataset and the Overlapping dataset each contain 300 images, containing many incomplete ellipses and broken arcs respectively. The Concentric dataset and the Concurrent dataset have 720 images and 1200 images respectively, whose contained concentric arcs and concurrent arcs are generally difficult to handle. The tested four real-world datasets are Prasad Dataset, Prasad+ Dataset, Random Dataset, and Smartphone Dataset. Prasad Dataset and Prasad+ Dataset [5] totally contain 400 images from Caltech256 dataset [33] with low resolutions, whose ellipses and occlusion cases are fewer than the other datasets. Random Dataset [6] has 400 images, whose quality is better than Prasad Dataset, containing overlapping ellipses and complicated backgrounds. Smartphone Dataset has 629 images from 6 video shots [6], which are mainly from traffic signs and bicycle wheels of various perspectives, generally used to test the performance of the methods in practical application scenarios.

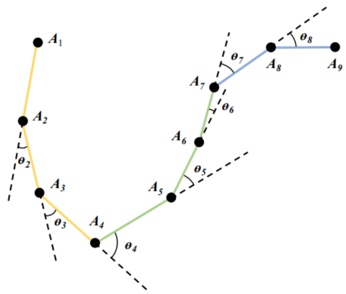
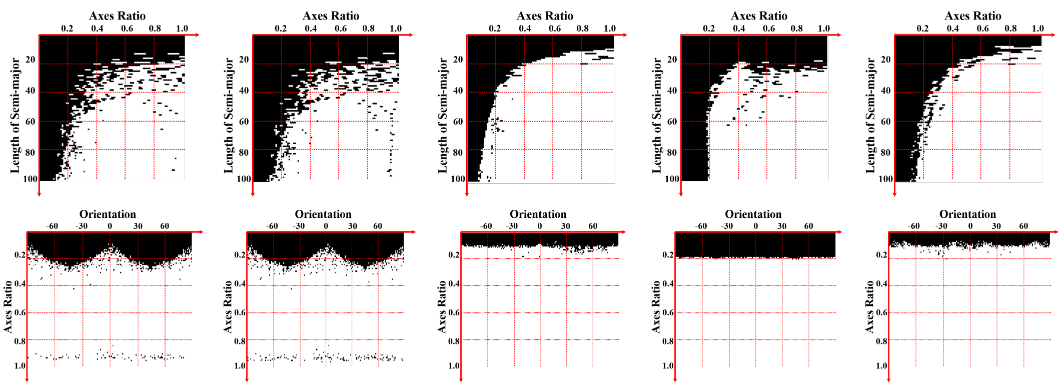


Figure 7: Contour segmentation by corner points. Here, A_4 is a corner point with an abrupt change in the magnitude of curvature, A_7 is a corner related to direction bending along the contour, and these two corner points segment the contour into three parts, marked in different colors.

Table 4: Recall rates for the methods on detecting various shape of ellipses.

Method	Recall(%)	
	First set	Second set
[6]	57.11	79.14
[7]	54.78	78.81
[8]	70.72	89.56
[10]	60.34	71.51
Ours	66.43	90.38



(a) [6] (b) [7] (c) [8] (d) [10] (e) Ours

Figure 8: Performance comparison for detecting ellipses in various shapes, orientations, and sizes. The left shows the relationship between the ratio of the minor axis to the major axis and the length of the major axis. The right shows the relationship between the ratio of the minor axis to the major axis and the orientation. Here, the white area indicates the set of ellipses that can be correctly detected, while the black area indicates the failed ones. Wang's method [15] fails to work in these tests.

Evaluation on variant axes ratio and orientation of ellipse. We generated two sets of ellipses in various shapes, orientations, and sizes to investigate the potential of our method. In the first set, 10,000 ellipses are each generated in images respectively, whose centers and orientations are fixed, but their semi-majors have lengths varied from 1 pixel to 100 pixels with an interval of 1 pixel, and their semi-minors have the lengths by the ratios of the minor axis to the major axis that vary from 0.01 to 1.0 with an interval of 0.01. In the second set, 18,000 ellipses are each generated in images respectively, whose orientations vary from -90° to 89° in a step of 1°. Here, with a direction, the semi-majors for the ellipses are fixed in the length of 100 pixels, and their semi-minors have the lengths varied by the ratios of the minor axis to the major axis varying from 0.01 to 1 in a step of 1. The results for the methods in comparison to detect the generated ellipses are shown in Fig. 8. The effectiveness of these methods to detect various ellipses are measured by the recall rates. The statistics in Table 4 show that our method can effectively detect a larger range of ellipses than the compared methods in general, except that in handling the first set, where ours is a little inferior to Meng *et al.*[8] in detecting the ellipses that are too small or too flat.

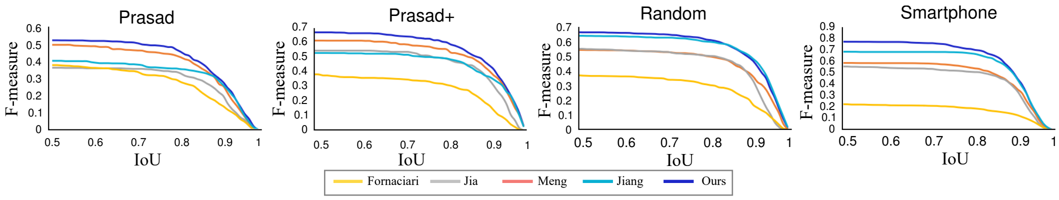
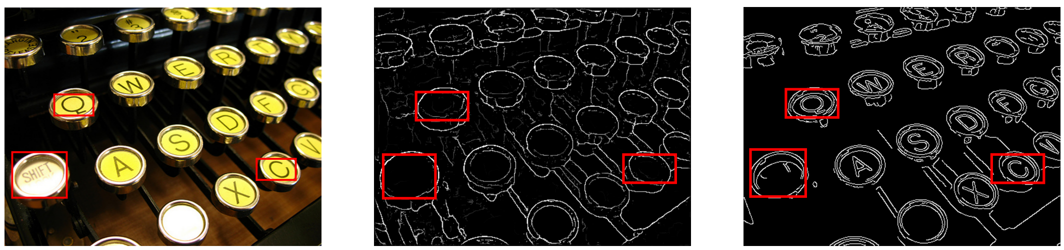


Figure 9: Ellipse detection performance of our method in comparison with state-of-the-art methods by the threshold for IoU varying from 0.5 to 0.99 with the interval of 0.01 on four real-world datasets, Prasad dataset, Prasad+ dataset, Random dataset and Smartphone dataset, respectively.



(a) (b) (c)

Figure 10: Comparison on edge extraction. (a) Input image. (b) Results of the learning method [25]. (c) Ours. Clearly, we can obtain many more arcs than using the learning method, as marked in the red boxes.

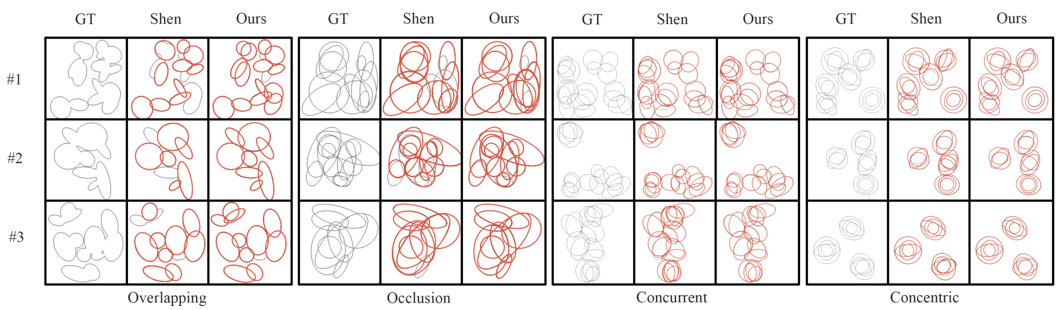


Figure 11: Some visualizations for the detection results of our method on the four synthetic datasets.

Performance over different IoU In general, when a detected ellipse has its IoU bigger than a threshold, it is regarded as correctly detected. In the paper, we use the results by setting the threshold as 0.8 for real-world datasets and 0.9 for the synthetic datasets. To further demonstrate the performance of our method, we performed experiments with the threshold ranging from 0.5 to 0.99. The results for the four real-world datasets are shown in Fig. 9, showing that our method achieves the best performance. When the threshold is lower than 0.75, our F-measure values no longer change significantly, showing our potentials for high-quality ellipse detection.

Comparison on edge extraction. We compare our edge extraction results with that of the learning based method [25]. As shown in Fig. 10, we can obtain many more arcs for detecting ellipses as many as possible, especially those overlapped ones. The method of [25] mainly extracts the arcs on the outer contours of objects, missing many overlapped ellipses, as shown in the red boxes in Fig. 10.

More results on synthetic datasets. We provide some visual comparison between the results of our method and that of [9] in Fig. 11 and Table 5. On the whole, we can always obtain many more correct ellipses for some complicated cases. In this experiment, we have IoU $= 0.9$ for validating ellipses because these images dose not contains noise or texture that may disturb arc extraction, as suggested by [7, 8, 9]. We also list the statistics of comparison on precision, recall and F-measure in Table 6.

Table 5: Statistics of the detection results in Fig. 11. We can detect more ellipses and in higher quality than [9], which can better extract ellipses than the other existing methods on synthetic datasets according to Table 6. TP, FP and FN stand for True Positive, False Positive and False Negative, respectively.

Method	Type	Occlusion#1	Occlusion#2	Occlusion#3	Overlapping#1	Overlapping#2	Overlapping#3
Shen	TP	9	6	9	8	7	7
	FP	1	0	0	3	4	1
	FN	7	2	3	4	5	1
Ours	TP	14	8	11	10	11	8
	FP	0	0	0	1	0	0
	FN	2	0	1	2	1	0

Method	Type	Concentric#1	Concentric#2	Concentric#3	Concurrent#1	Concurrent#2	Concurrent#3
Shen	TP	19	16	15	15	11	11
	FP	0	2	5	1	2	1
	FN	1	0	1	1	1	1
Ours	TP	20	16	16	16	12	12
	FP	2	1	0	0	0	0
	FN	1	0	0	0	0	0

Table 6: The test results of the compared methods on the four synthetic datasets. Here, the values for the metrics are the averaged ones for an image in a dataset, and the best results and the second best results are marked in red and yellow respectively.

Method	Occlusion			Overlapping			Concentric			Concurrent		
	Precision↑	Recall↑	F-measure↑	Precision↑	Recall↑	F-measure↑	Precision↑	Recall↑	F-measure↑	Precision↑	Recall↑	F-measure↑
[6]	0.0904	0.3624	0.1353	0.0881	0.2216	0.1260	0.0542	0.7881	0.1015	0.0684	0.8926	0.1271
[7]	0.4674	0.2688	0.2944	0.3197	0.1659	0.2155	0.4587	0.6426	0.5353	0.4370	0.8079	0.5672
[14]	0.4889	0.4559	0.4685	0.6024	0.5287	0.5231	0.6627	0.8546	0.7465	0.6635	0.8392	0.7411
[8]	0.5558	0.1774	0.2492	0.4910	0.2680	0.3462	0.7428	0.6692	0.7041	0.7727	0.7340	0.7528
[9]	0.5955	0.4587	0.5174	0.6048	0.4267	0.4686	0.8742	0.8435	0.8586	0.8193	0.9135	0.8638
[10]	0.4441	0.1350	0.2009	0.7238	0.3874	0.4498	0.8095	0.8446	0.8267	0.6996	0.9337	0.7999
[15]	0.0863	0.0280	0.0422	0.0934	0.0249	0.0366	0.0310	0.0096	0.0147	0.1386	0.0622	0.0859
Ours	0.7074	0.5558	0.6191	0.6773	0.4827	0.5282	0.9117	0.8860	0.8987	0.8737	0.9430	0.9070

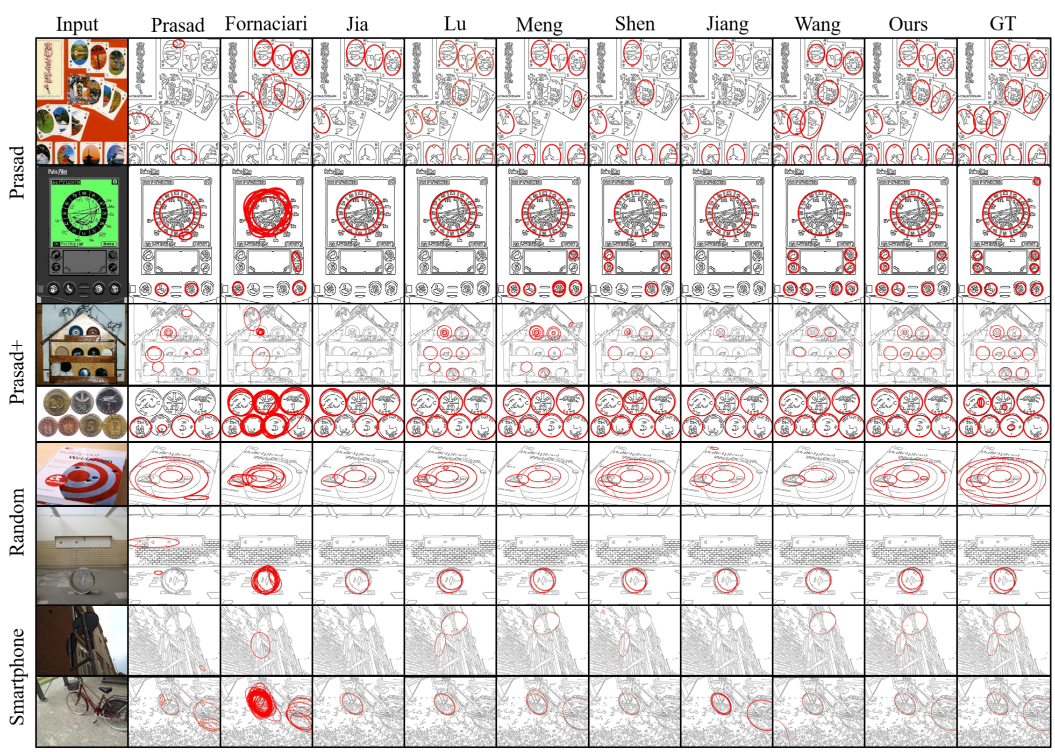


Figure 12: Some detection results of the methods in comparison on real-world datasets.

Table 7: The test results of the compared methods on the four real-world datasets. Here, the values for the metrics are the averaged ones for an image in a dataset, and the best results and the second best results are marked in red and yellow respectively.

Method	Prasad		Prasad+		Random		Smartphone	
	F-measure↑	Time↓	F-measure↑	Time↓	F-measure↑	Time↓	F-measure↑	Time↓
[5]	0.2874	2253.82	0.2108	5697.04	0.3112	6185.56	0.2226	13721.00
[6]	0.2888	4.48	0.2072	12.18	0.3063	13.58	0.1919	18.63
[7]	0.3343	4.10	0.4896	8.32	0.5016	10.79	0.5222	14.58
[14]	0.5092	162.70	0.6540	550.49	0.6009	640.23	0.6403	1118.08
[8]	0.4293	3.75	0.5539	7.78	0.4997	9.71	0.5510	14.66
[9]	0.4265	7.96	0.5713	14.18	0.5838	17.48	0.6424	25.20
[10]	0.3552	6.60	0.4851	10.15	0.6022	15.97	0.6825	24.53
[15]	0.3866	56.47	0.4648	55.65	0.5559	54.48	0.5246	55.35
[12]	0.3425	3.95	0.5198	6.94	0.5144	9.32	0.5000	12.41
[31]	0.4332	4.07	0.5618	7.96	0.5104	10.75	0.5629	17.73
Ours	0.4632	4.09	0.6012	6.61	0.6106	7.76	0.7006	11.81
*Ours	0.5126	5.95	0.6589	10.01	0.5898	13.49	0.6108	24.10
Ours+CM	0.4381	3.82	0.5742	6.11	0.5815	7.37	0.6689	11.12

Notes: 1) [15] runs on GPU, and the other methods run on CPU. Time is in millisecond.
2) "*Ours" refers to using our method without filtering in edge detection and with constraints relaxed in arc determination.
3) [12] replace characteristic number with characteristic mapping(CM) for arc grouping of [7].
4) "Ours+CM" refers to that our arc groups are further filtered by the CM constraints [12] before ellipse generation.

More results on real-world datasets. In the main paper, we only provide the statistics of some recent methods and limited visual results. Here, we provide more quantitative comparison in Table 7, and more visualization of detected ellipses in Fig. 12.

C Promotion to application of autonomous driving

Traffic sign detection is a crucial problem in autonomous driving, where it is very important to detect traffic signs as early and thoroughly as possible. Among all traffic signs, circular signs account for a large proportion, and provide key information about traffic rules and restrictions. Thus, ellipse detection in the captured images of the cameras for autonomous driving are much required.

We made a test by comparing our method and Jia *et al.* [12] on ellipses detection. Here, the used images are from the dataset collected from video frames captured by a mobile phone [12] and a set of complex real images containing circular traffic signs from the public Traffic Sign Detection Dataset (TSDD)[1].

As illustrated by some results in Fig. 13, we can detect more traffic signs than the method of Jia *et al.* [12]. Thus, we can promote the safety of autonomous driving, as discussed in the following. Firstly, we can more effectively detect small-sized ellipses, as shown in Fig. 13(a)(f), which means that traffic signs can be recognized from a greater distance, improving the timely response of autonomous driving systems. Secondly, our method detects more traffic signs, as shown in Fig. 13(c)(e), thereby avoiding the risk of missing critical information. Thirdly, we can effectively identify incomplete signs, as shown in Fig. 13(b), which are quite common in real-world scenarios due to limitations such as camera field of view or obstructions. Clearly, with our method, autonomous driving can be promoted a lot.

[1]It is part of the Chinese Traffic Sign Database (`https://nlpr.ia.ac.cn/pal/trafficdata/index.html`) collected by Huang *et al.*.

Figure 13: Visualization results of our method and Jia-CM [12] on real-world scenes of traffic sign detection.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Please refer to the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to the Conclusions section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper does not include theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please refer to the "Results and discussion" section and Appendix A. We provide detailed information about the experiments. The datasets are publicly available.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
    (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
    (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
    (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
    (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

    Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

    Answer: [No]

    Justification: We currently dose not provide our source code, but will be willing to release on acceptance.

    Guidelines:

    - The answer NA means that paper does not include experiments requiring code.
    - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
    - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
    - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
    - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
    - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
    - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to the "Results and discussion" section and Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We provide experiment statistics in the "Results and discussion" section and Appendix B. However, the statistical significance is usually not part of the results as in related work.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Results and discussion.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: No societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: No such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All assets that are produced by others are properly cited and the license is respected.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: Does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Only used for editing.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.