Eigencontours: Novel Contour Descriptors Based on Low-Rank Approximation

Wonhui Park
Korea University
whpark@mcl.korea.ac.kr

Dongkwon Jin
Korea University
dongkwonjin@mcl.korea.ac.kr

Chang-Su Kim
Korea University
changsukim@korea.ac.kr

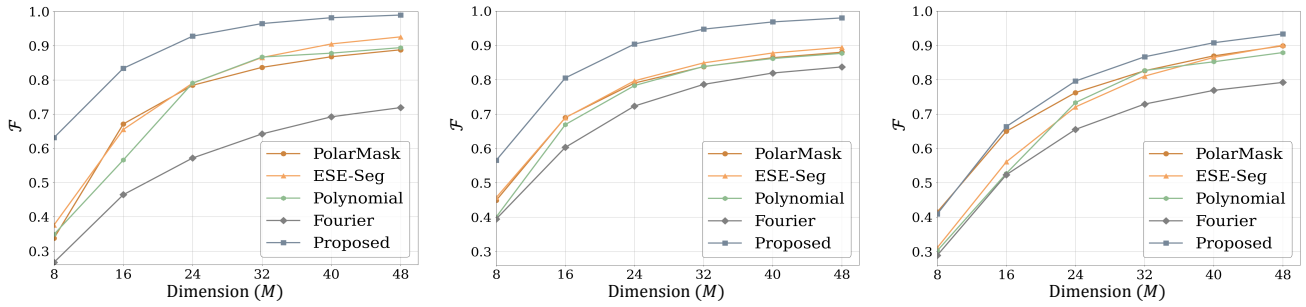
A. Contour Description

A.1. Comparison with more conventional contour descriptors

In Section 4.2 in the paper, we compare the proposed eigencontours with PolarMask [27] and ESE-seg [28] on the KINS, SBD, and COCO2017 datasets. In this supplemental document, for more comparison, we also test traditional contour descriptors: Polynomial [5] and Fourier [31]. In Polynomial [5], cubic polynomials are employed to represent segments of an object contour, while Fourier descriptors are determined from the Fourier series of centroidal profiles in Fourier [31].

Figure 1 compares the $\mathcal{F}$ curves according to the dimension M of the descriptors. In Polynomial, an object contour is divided into $M/4$ segments, and then each segment is represented by 4 cubic polynomial coefficients. In Fourier, M Fourier descriptors are used. For all three datasets, the proposed algorithm is superior to both Polynomial and Fourier, as well as to PolarMask and ESE-seg, at every M. Table 1 compares the area under curve performances of the $\mathcal{F}$ curves (AUC-$\mathcal{F}$) in Figure 1. The proposed algorithm outperforms the conventional algorithms by significant margins on all datasets. In other words, the proposed algorithm represents object boundaries more faithfully than the conventional algorithms do, when the same number of coefficients are used for the contour description.

Figure 2 compares object boundaries approximated by the contour descriptors at $M = 16$. The conventional algorithms fail to reconstruct detailed parts, such as legs or wheel boundaries. In contrast, the proposed eigencontour descriptors represent the object boundaries more accurately and more reliably.



(a) KINS (b) SBD (c) COCO2017

Figure 1. The $\mathcal{F}$ score curves of the proposed eigencontours and the conventional contour descriptors in PolarMask [27], ESE-Seg [28], Polynomial [5], and Fourier [31], according to the dimension M of the descriptors.

Table 1. AUC-$\mathcal{F}$ performances on KINS, SBD, and COCO2017.

	KINS	SBD	COCO2017
PolarMask [27]	75.47	76.23	74.05
ESE-Seg [28]	77.37	76.86	70.21
Polynomial [5]	74.49	75.15	69.64
Fourier [31]	57.29	70.23	63.67
Proposed	**89.17**	**86.51**	**76.92**

Figure 2. Qualitative comparison of boundary representations at $M = 16$. The left four images are from KINS, the middle three from SBD, and the remaining three from COCO2017.

A.2. Clustering in low-dimensional space

Using the proposed eigencontours, we can cluster object contours in a lower-dimensional descriptor space and obtain contour centroids there. As shown in Table 2 in the main paper, the proposed algorithm processes object contours more reliably in a low-dimensional space than PolarMask [27] and ESE-Seg [28] do. Figure 3 compares some centroids on the COCO2017 dataset at $M = 16$ and $K = 500$. In COCO2017, about 50% of objects are classified into 'person' among 80 categories. Thus, most centroids represent rough shapes of a human, such as an upper body or a standing one with a pair of legs. Note that PolarMask fails to reconstruct curved parts. ESE-Seg provides curved shapes, but it blurs complicated parts, especially leg boundaries. In contrast, the proposed eigencontour descriptors represent object boundaries more faithfully.

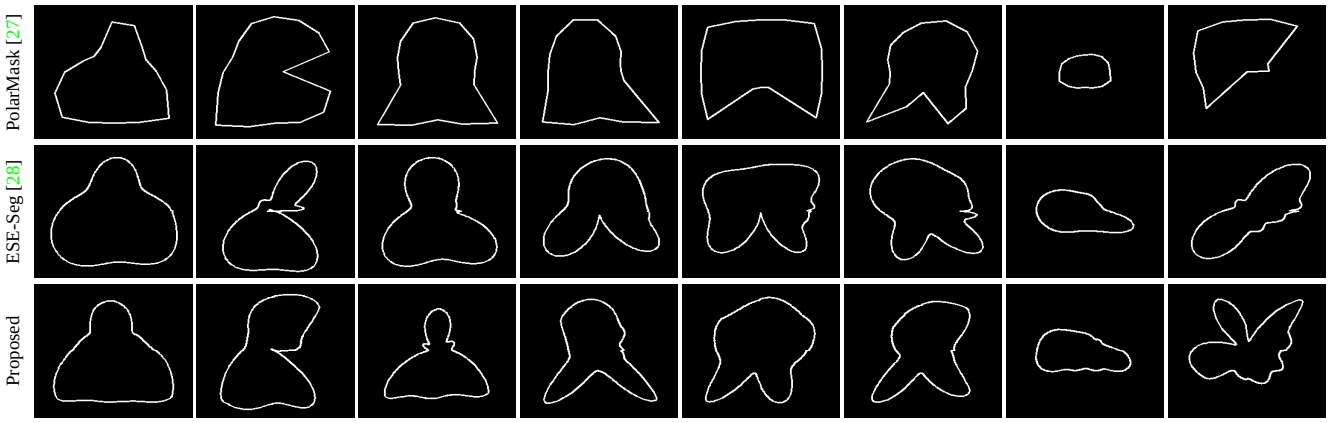


Figure 3. Comparison of clustering results on the COCO2017 dataset at $M = 16$ and $K = 500$.

B. Instance Segmentation

B.1. Network details

We incorporate the proposed eigencontours into an instance segmentation framework. To this end, as done in ESE-Seg [28], we adopt YOLOv3 [23] as an object detector and modify its components. Figure 4 illustrates the structure of the modified YOLOv3 network for the SBD validation dataset. We implement the encoder based on Darknet-53, which extracts a convolutional feature map of size $H \times W \times 512$. Then, using 2D convolutional layers, we yield an output map of size $H \times W \times (25 + M + 2)$. In the output map, each element contains an M-dimensional coefficient vector for object contour regression and a 2-dimensional position vector for object centroid regression, as well as a 25-dimensional YOLOv3 vector for bounding box regression and object classification.

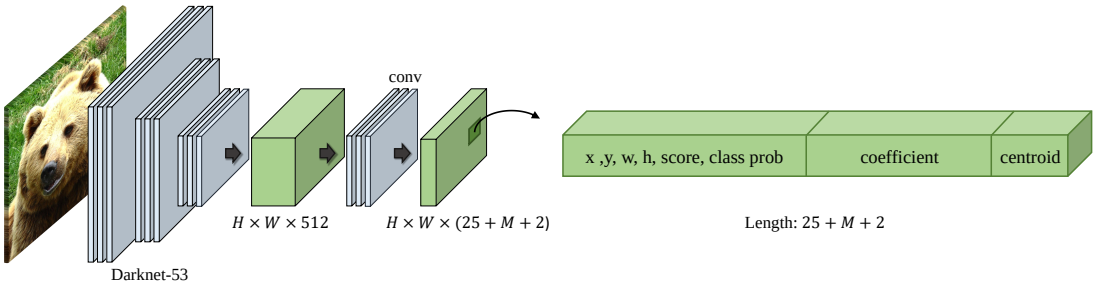


Figure 4. Modified YOLOv3 network for the instance segmentation.

B.2. Training details

We define the loss for training the network as

$$\ell_{\text{total}} = \lambda_{\text{cls}}\ell_{\text{cls}} + \lambda_{\text{reg}}\ell_{\text{reg}} + \lambda_{\text{coeff}}\ell_{\text{coeff}} + \lambda_{\text{cen}}\ell_{\text{cen}}, \tag{1}$$

where ℓ_{cls} is the cross-entropy loss over the classes, ℓ_{reg} is the mean squared error (MSE) loss for the bounding box regression, and ℓ_{coeff} and ℓ_{cen} are also the MSE losses for the coefficient vector and the position vector, respectively. Also, $\lambda_{\text{cls}} = \lambda_{\text{reg}} = 1$, and $\lambda_{\text{coeff}} = \lambda_{\text{cen}} = 1000$.

As done in ESE-Seg [28], we use the stochastic gradient descent (SGD) optimizer with an initial learning rate of 10^{-4} and halve it after every 50 epochs three times. Also, we use a batch size of 16 for 600,000 iterations and augment training images by randomly flipping them horizontally. For the SBD dataset, we resize training images to 416×416.

B.3. More instance segmentation results on SBD dataset

Figure 5 compares the proposed algorithm with the conventional algorithms [27,28] on the SBD dataset.

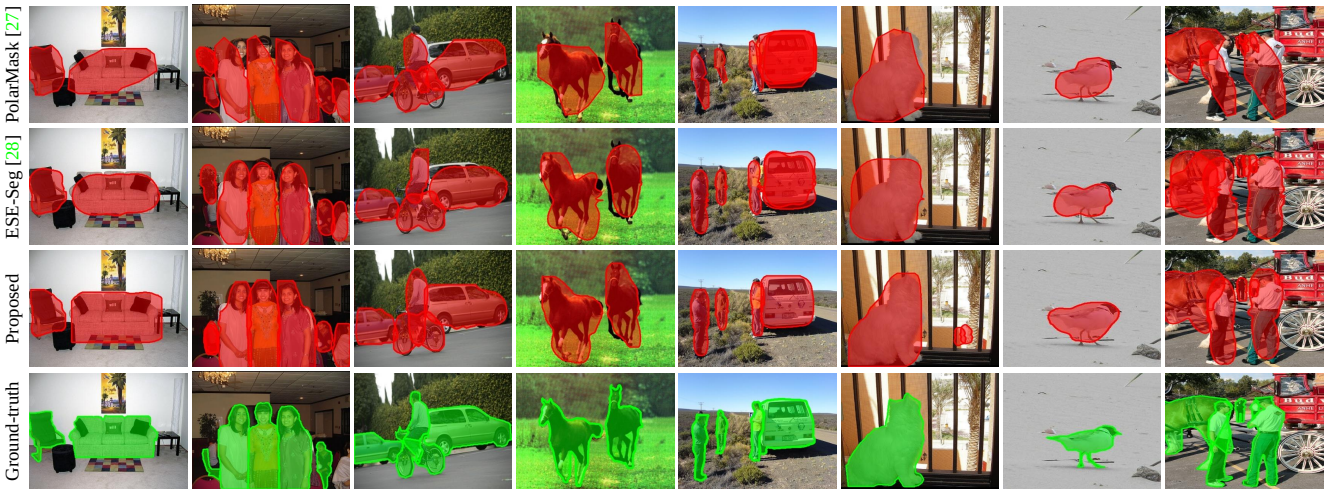
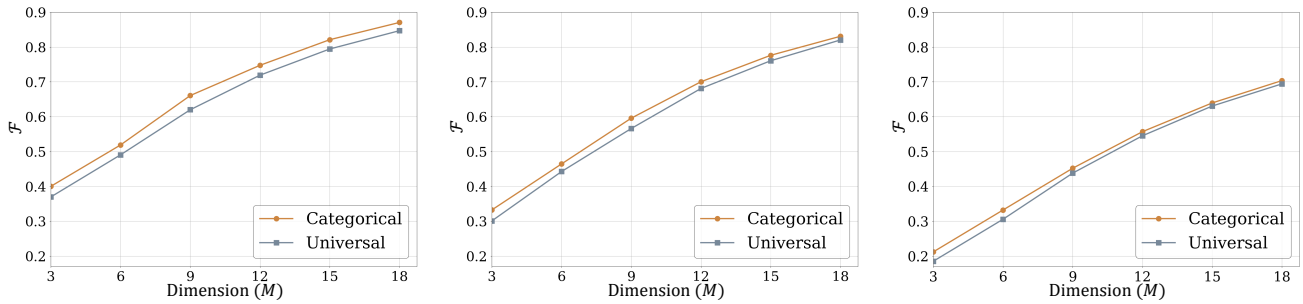


Figure 5. Comparison of instance segmentation results on the SBD dataset.

C. Analysis

C.1. Categorical eigencontour space

As described in Section 4.3, we obtain eigencontour descriptors for two options: categorical construction and universal construction. Figure 6 compares $\mathcal{F}$ curves for the two options, according to the dimension $M \in [3, 18]$. Note that the AUC-$\mathcal{F}$ performances of these curves are summarized in Table 5 in the main paper. We see that the categorical construction provides better performances than the universal construction, since it considers similar shapes in the same category only.



(a) KINS (b) SBD (c) COCO 2017

Figure 6. The $\mathcal{F}$ score curves of categorical and universal eigencontours, according to the dimension $M \in [3, 18]$.

C.2. Cross-validation test

As mentioned in Section 4.3, the proposed eigencontours for a dataset may be effective for that particular dataset only. To assess the dependency on a dataset, we conduct cross-validation tests between the three datasets: KINS, SBD, and COCO2017. First, for each dataset, we determine eigencontour descriptors. Second, we use these descriptors to represent object boundaries in the other two datasets. Table 2 shows the AUC-$\mathcal{F}$ performances of these cross-validation tests. Compared to the diagonal scores in Table 2, the cross-validation scores are decreased. However, the amounts of degradation are less than about 2, indicating that the dependency is not a severe problem in practice. It is also observed that the eigencontours for COCO2017 are more reliable than those for KINS and SBD. This is because COCO2017 contains more diverse objects than KINS and SBD do.

Table 2. AUC-$\mathcal{F}$ performances of cross-validation on KINS, SBD, and COCO2017.

Training set \ Test set	KINS	SBD	COCO2017
KINS	89.17	85.16	75.79
SBD	86.07	86.51	76.15
COCO2017	87.16	85.66	76.92

D. More Examples

D.1. Eigencontours

Figure 7 shows the first ten *universal* eigencontours for the SBD and COCO2017 datasets, respectively. On the other hand, Figure 8 shows the first five *categorical* eigencontours for separate object categories.

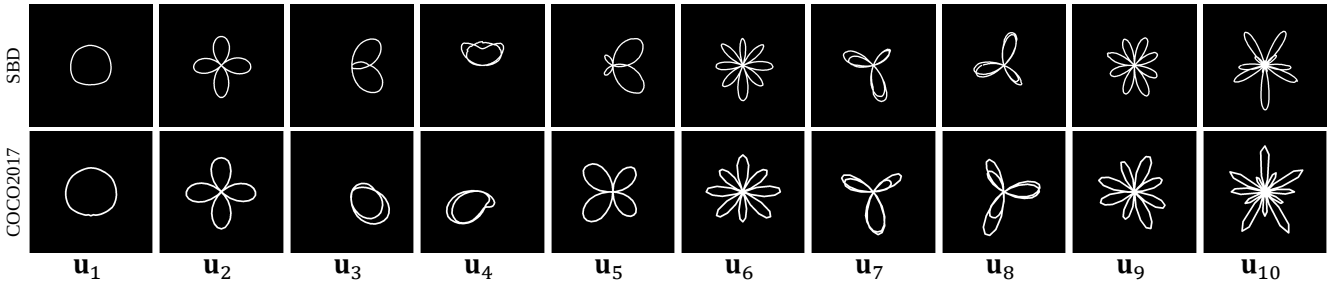


Figure 7. The first ten eigencontours for the SBD and COCO2017 datasets (universal construction).

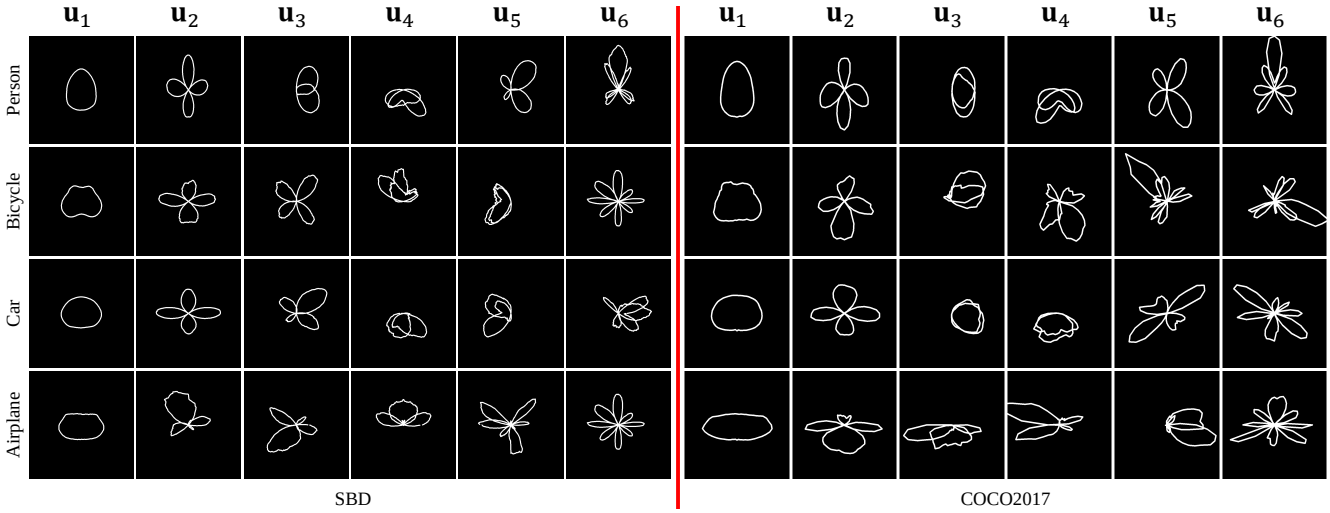


Figure 8. The first five eigencontours for separate categories in the SBD and COCO2017 datasets (categorical construction).

D.2. Rank-M approximation

Figure 9 shows more examples of original boundaries and their rank-M approximations.

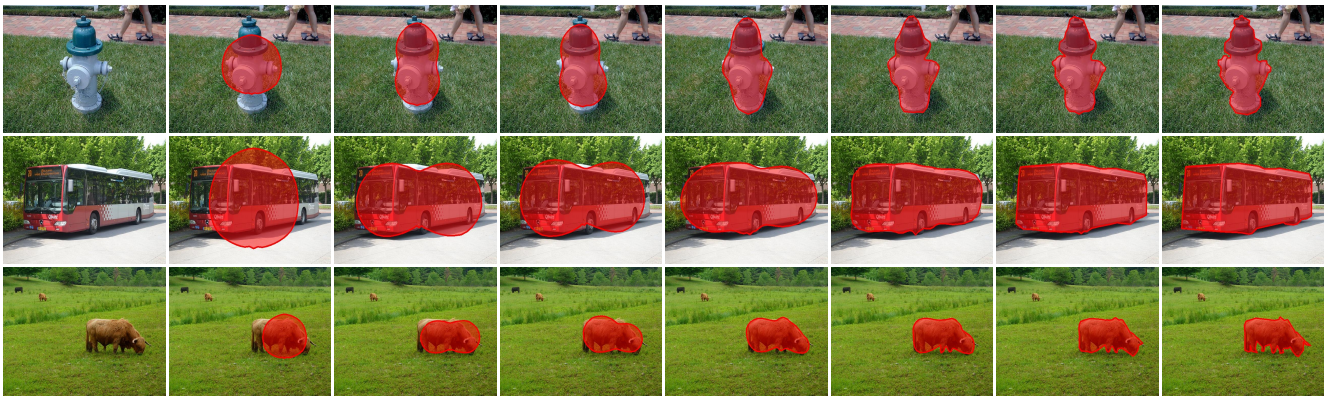


Figure 9. Rank-M approximations and the star-convex conversions of original contours.

D.3. Clustering in eigencontour space

Figure 10 shows more examples of contour centroids for the SBD and COCO2017 datasets at $M = 16$ and $K = 100$.

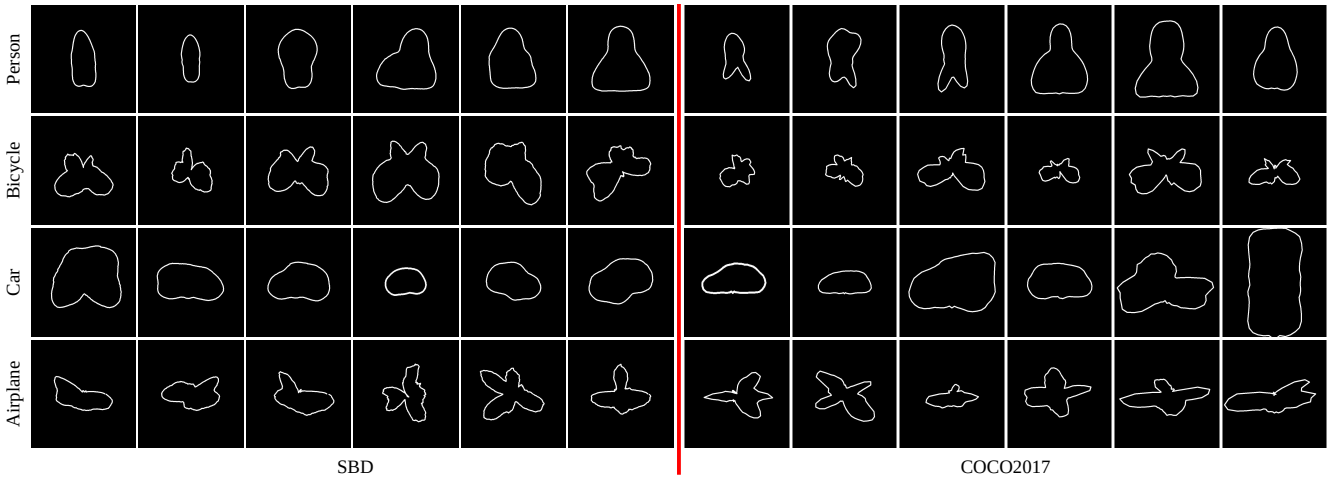


Figure 10. Visualization of contour centroids on the SBD and COCO2017 datasets.