2022 I-RIM Conference
October 7-9, Rome, Italy
ISBN: 9788894580532
DOI:10.5281/zenodo.7531208

FASTDLO: Towards Real-Time Perception of Deformable Linear Objects

Kevin Galassi, Alessio Caporali, Riccardo Zanella, Gianluca Palli

***Abstract*—In this paper is presented an approach for fast and accurate segmentation of Deformable Linear Objects (DLOs) named *FASTDLO*. The perception is obtained from the combination of a deep convolutional neural network for the background segmentation and a pipeline for the dlo identification. The pipeline is based on skeletonization algorithm to highlights the structure of the DLO and a similarity-based network to solve the intersection. *FASTDLO*is trained only on synthetically generated data, leaving real-data only for evaluation purpose. *FASTDLO* is experimentally compared against DLO-specific approach achieving better overall performances and a processing rate higher than 20 FPS.**

***Index Terms*—Deformable Linear Objects, Industrial Manufacturing, Computer Vision**

I. Introduction

Deformable Linear Objects (DLOs) are a special subgroup of materials composed mainly by wires, electrical cable and ropes. These materials are important for a large variety of industrial application such as Aerospace or Automotive. Given the difficulties on the perception and manipulation, they still strongly rely on human operator for various assembly task. Furthermore, the development of more intelligent machine are impeded by the solutions available for the perception of such materials. In particular, most research only consider a simplification of a real working environments supposing to have a controlled background (green or white) from which is easier to detect the cable using simple threshold based method [1], [2]. Other research limit the number of DLO to perceive to just one [1]. In the spirit to overcome such limitation, previous attempt were made by *Ariadne+* [3], an algorithm based on the segmentation and modeling of DLO based on its predecessor *Ariadne* [4]. Differently from the other approaches, it is capable to achieve better accuracy and efficiency and it has the ability to consider even more complex scenarios in which the endpoints of the cables were not present in the image. Despite being the state of the art of DLO-sensing, the major drawback of the algorithm was the high time of execution that was limiting his application for more complex task such as cable tracking. In this paper, is presented *FASTDLO* (FAst SegmenTation of Deformable Linear Objects) [5], an algorithm for the precise instance segmentation of DLO based on no-assumption of the environments with a faster execution time. The algorithm takes as input an image, by using a DCNN it generates a binary mask with all the DLO in the scene and after that it and provides as output the colored mask associated to the DLOs in the scene (if any) with distinct color, all the non-dlo pixel are depicted as black. The DLO instances are modeled by means of a sequence of key-points to be interpolated as spline. The learning part was trained using only synthetic dataset, leaving the real dataset only for validation. As processing speed, *FASTDLO* achieves an overall rate higher than 20 Frames-Per-Second (FPS) with an image size of 640×360 pixels, employing, as hardware, a workstation with an Intel Core i9-9900K CPU clocked at 3.60GHz and an NVIDIA GeForce GTX 2080 Ti. PyTorch 1.4 is used for the software implementation. To summarize, the main characteristic of *FASTDLO*:

- Reliable and efficient method for the instance segmentation of DLOs
- Fast deployment with the use of synthetic data for all the data-driven approaches;
- use of appearance-based and topological features for DLO identification;
- Better performances in terms of speed and accuracy.

The source code implementing *FASTDLO* and the associated data is available at https://github.com/lar-unibo/fastdlo.

II. The *FASTDLO* Algorithm

The *FASTDLO* pipeline, schematized in Fig. 1, consists of the following main steps: **Background Segmentation**, **Skeleton Pixels Classification**, **Segments Generation**, **Intersections Processing**, **Informed Merging**, **Intersections Layout** The aforementioned steps are discussed, a deeper analysis is available in [5].

A. Background Segmentation

The generic input image I_s is processed by means of a DCNN using a semantic segmentation network using DeeplabV3+ [6] as architecture, producing as output a binary mask M_b, with DLOs pixels labeled in white and as black the remaining pixels. The training of the DCNN were done by using only synthetic data generated using a Blender based pipeline [7]. In total, $32,000$ images were rendered with various DLOs' color, sizes and lengths, light condition and backs. An example of the segmentation can be seen in a real sample in Fig. 2, the shadows and the complex background doesn't affect the final result. For clarity, the intersection area is zoomed.

Alessio Caporali, Kevin Galassi, Riccardo Zanella and Gianluca Palli are with DEI - Department of Electrical, Electronic and Information Engineering, University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy.

This work was supported by the European Commission's Horizon 2020 Framework Programme with the project REMODEL - Robotic technologies for the manipulation of complex deformable linear objects - under grant agreement No 870133.

Corresponding author: *kevin.galassi@unibo.it*

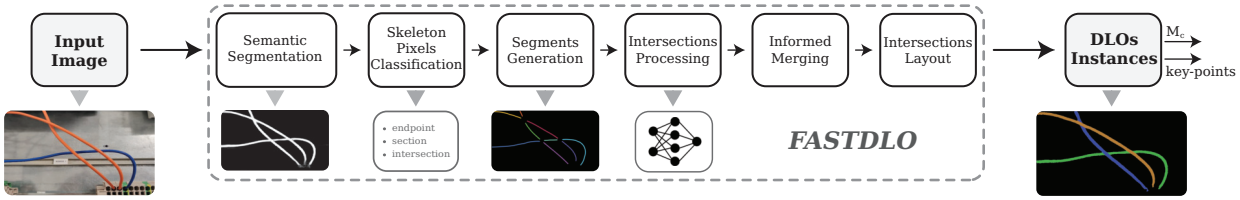
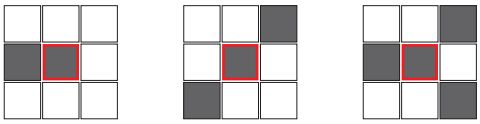


Fig. 1: The *FASTDLO* algorithm.

Fig. 2: Input image with background segmentation, result and generated skeleton zoomed at the intersections area.



(a) endpoint (b) section (c) intersection

Fig. 3: Local neighbors possibilities of a skeleton pixel given a 3×3 kernel. To clarify the representation, the skeleton is in dark while the background is in white.

B. Skeleton Pixels Classification

The segmentation mask M_b is processed with a skeletonization algorithm consisting of a thinning iterative approach which erodes the input mask obtaining a new mask M_s. For each pixel of the skeleton M_s, is defined a small (3×3) kernel, three types of local neighbors distintct by the pixel distribution around the central one, the cases are named as *endpoint*, *section* and *intersection* (shown in Fig. 3).

C. Segments Generation

The skeleton's pixels are merged based on their label forming continuous segment, in this process the intersections' segments are not considered forming a list of distintc linear segments. This pixel are introduced later, after the intersection connection are evaluated.

D. Intersections Processing

The intersections among the DLOs are solved by comparing the feature vectors of the endpoints of two candidate segments via a shallow neural network, i.e. similarity network, predicting the probability of their connection. As features embedded are used the RGB color of the endpoints, the thickness of the mask and the direction of the nodes. The loss is computed between an anchor, a positive and a negative sample. The distance in the embedding space between anchor and positive is minimized, while the one between anchor and negative is maximized.The prediction is based on the distance of the embedding vectors. To obtain a probability-like value in the $[0, 1]$ range describing the likelihood of the connection is used a Gaussian activation function.

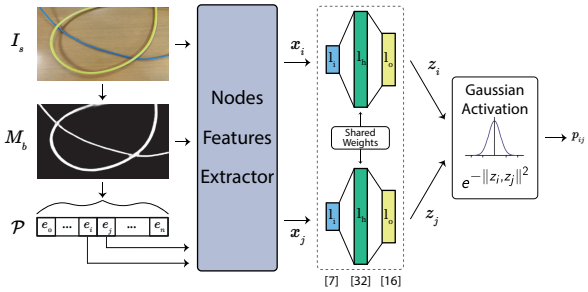
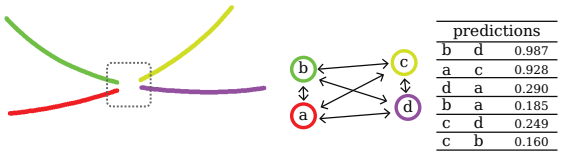
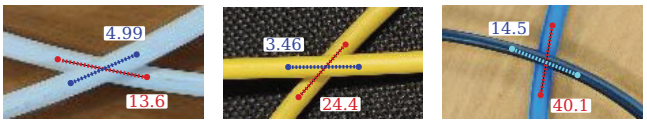


Fig. 4: Endpoint-pair probability computation with a similarity network using the node features embedding vector.



predictions		
b	d	0.987
a	c	0.928
d	a	0.290
b	a	0.185
c	d	0.249
c	b	0.160

(a) different *segments* (b) intersection processing with predictions of the endpoints pairs



(c) example 1 (d) example 2 (e) example 3

Fig. 5: (a) *segments* generated; (b) example of intersection processing of the region highlighted area in (a). Example of the intersection layout estimation with DLOs having identical colors, in (c) and (d), and different colors (e)

E. Informed Merging

Exploiting the endpoint-pairs connection probabilities computed in Sec. II-D it is possible to concatenate *segments* obtaining the full description of each DLO in the image. This concatenation process is addressed as informed merging. The probability scores are sorted in descending sorted. For each probability, the relative segments obtained in II-C are merged together into new segments with a new pair of endpoints. Consequently, the endpoint-pairs having lower scores and with one of the two endpoint elements already associated are not considered and their merging avoided.

F. Intersections Layout

As final step, the DLOs are ordered by obtaining the one at the top of each intersection by comparing the standard deviation of the RGB colors along the line connecting the endpoint-pair previously solved. This solution work with both

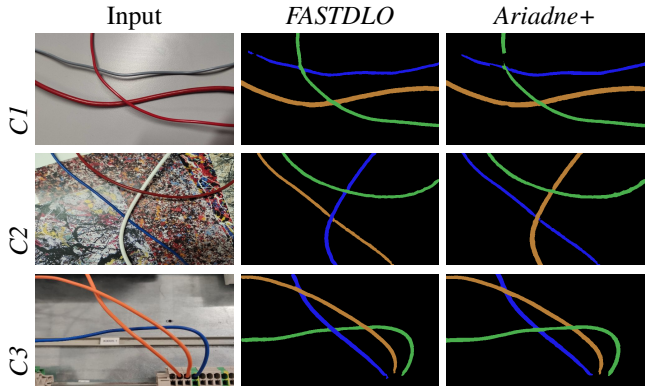


Fig. 6: Qualitative evaluation of *FASTDLO* and the best performing baseline using a sample for each category.

Method	Backbone	Key-points	FPS	Time [ms]	IoU [%]
Ariadne+	ResNet-50	✓	3	354	73.96
Ariadne+	ResNet-101	✓	3	360	76.87
FASTDLO	**ResNet-50**	✓	**23**	**44**	**73.89**
FASTDLO	**ResNet-101**	✓	**22**	**46**	**77.77**

TABLE I: Comparison of *FASTDLO* with *Ariadne+* [3]. Other method test (Not shown in the table) involved [8], [9], [10].

colored and b/w images, in the first case because it is helped by the different color, while in the latter since it exploit the shadow produced by the DLO at the top. In Fig. 5 some example intersections are displayed with the computed values.

III. Experimental Validation

The evaluation of *FASTDLO* is performed on real data, a *test set* of 135 manually labeled real images of electrical wires organized in categories based on the background characteristic and the numbers of intersections in the scene.

C1: Plain background with high contrast.
C2: Highly featured background (e.g. Pollock).
C3: Real-case scenario (e.g. Electrical Switchgear).

FASTDLO achieves better overall scores, showing a large advantage over the general purpose approaches and performing slightly better compared to *Ariadne+*, where both methods employ the same weights in the segmentation network. From the processing time perspective, *FASTDLO* is competitive with respect to the general purpose methods while being almost one order of magnitude faster than *Ariande+*. In Fig. 6 some samples for each *test set* category are shown with the corresponding output predictions obtained with *FASTDLO* and with *Ariande+*.

The prediction performances of the intersections layouts, i.e. Sec. II-F, are also evaluated on the *test set*. Considering only the correct endpoint-pairs predictions, the approach discussed in Sec. II-F is able to provide a correct result in 226 of the totals of 232 intersections, achieving an overall accuracy of 97.4% compared to $78.3\%\,(177/226)$ of *Ariadne+*. Thus, it is clear the validity of the proposed method that is executed without noticeable overhead in terms of processing time. In

Procedure	ResNet-101			ResNet-50		
	1	2	3	1	2	3
Binary Segmentation	19.72	19.49	19.49	15.73	15.67	15.67
Skeleton Generation	12.27	13.26	14.25	12.86	14.33	14.78
Endpoint-pairs Predictions	0.80	1.04	1.21	0.86	1.01	1.16
Informed Merging	15.13	17.09	18.81	15.93	18.57	19.58
Total	41.91	45.62	50.53	39.26	45.43	49.07

TABLE II: Average execution times of *FASTDLO* with respect to the number of intersections in the image, i.e. 1, 2, and 3, and the backbone. Values expressed in milliseconds

Tab. II a characterization of the average timing for the different stages of the method is provided by analyzing the effects of the ResNet backbones and the number of intersections in the image. As the number of intersections increases, both the segmentation and endpoint-pairs predictions times stay relatively constant batch-inference. The skeleton generation time increases of about 15% from 1 to 3 intersections. Also, the additional processing time, mostly due to the informed merging approach, increases with the number of intersections, as expected. Overall, the total processing time is in the range of 40 to 50 ms in all the conditions. *FASTDLO* can be extended to work with a large variety of DLOs, in particular it was tested on semi-transparent plastic medical hoses. For other DLOs such as ropes or string, it may be required to re-train the segmentation with a more specific dataset.

IV. Conclusions and Future Work

In this paper, a DLOs instance segmentation algorithm is presented capable to reach processing rate higher than 20 FPS with reliable and accurate prediction. As future work, *FASTDLO* will be integrated into application where the information provides by the algorithm may be required, for instance the cable-tracking.

References

[1] A. Keipour, M. Bandari, and S. Schaal, "Deformable one-dimensional object detection for routing and manipulation," *IEEE Robotics and Automation Letters*, 2022.
[2] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE robotics and automation letters*, 2020.
[3] A. Caporali, R. Zanella, D. De Gregorio, and G. Palli, "Ariadne+: Deep learning-based augmented framework for the instance segmentation of wires," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2022.
[4] D. D. Gregorio, G. Palli, and L. D. Stefano, "Let's take a walk on superpixels graphs: Deformable linear objects segmentation and model estimation," in *Asian Conference on Computer Vision*. Springer, 2018.
[5] A. Caporali, K. Galassi, R. Zanella, and G. Palli, "Fastdlo: Fast deformable linear objects instance segmentation," *IEEE Robotics and Automation Letters*, 2022.
[6] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. of the ECCV*, 2018.
[7] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam, "Blenderproc," *arXiv preprint arXiv:1911.01911*, 2019.
[8] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," in *Proc. of the ICCV*, 2019.
[9] ——, "Yolact++: Better real-time instance segmentation," *IEEE transactions on pattern analysis and machine intelligence*, 2020.
[10] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, and Y. Yan, "Blendmask: Top-down meets bottom-up for instance segmentation," in *Proc. of the CVPR*, 2020.