Journal of Telecommunication, Electronic and Computer Engineering
ISSN: 2180 – 1843 e-ISSN: 2289-8131 Vol. 16 No. 3 (2024) 35-39
jtec.utem.edu.my
DOI: https://doi.org/10.54554/jtec.2024.16.03.005

Enhanced Malaysian License Plate Recognition System using an Improved YOLOv2 Model

A.R. Syafeeza[1*], P. Marzuki[1], Asar Khan[1], Norihan Abdul Hamid[1], Wira Hidayat Mohd Saad[1], and Airuz Sazura A. Samad[2]
[1]Machine Learning and Signal Processing (MLSP) Research Group, Fakulti Teknologi dan Kejuruteraan Elektronik dan Komputer (FTKEK), Universiti Teknikal Malaysia Melaka, 76100, Durian Tunggal, Melaka, Malaysia.
[2]Venture GES Manufacturing Services (M) Sdn. Bhd, 81400, Kulai Jaya, Johor, Malaysia.

Article Info

Article history:
Received Feb 23rd, 2024
Revised June 13th, 2024
Accepted Sep 2nd, 2024
Published Sep 30th, 2024

Index Terms:
License Plate
Detection
Recognition
Preprocessing
Convolutional Neural Network
Malaysian License Plate Recognition

Abstract

License Plate Recognition (LPR) has gained popularity among researchers due to its wide range of applications, including law enforcement, monitoring, and toll gate systems. However, existing LPR systems still require improvements to achieve optimum accuracy and speed. The advancements in Convolutional Neural Network (CNN) variants offer potential solutions for these challenges. This primary aim of this system is to ensure accurate and efficient recognition of the vehicle plate characters using CNN techniques. This research utilizes two CNN network architectures for deep object detection to address the Malaysian License Plate Recognition (MLPR) task. The first network is designed to detect the license plate, while the second is responsible for recognizing the characters on the plate. Both networks are cascaded from the architecture of two-stage YOLOv2, providing promising speed and accuracy. The MLPR system achieved an accuracy of 98.75% and a processing speed of 0.0104 seconds, using a total of 2,200 license plate images. In conclusion, the system adapted from deep object detection techniques presents a promising solution for the MLPR problem, based on the achieved accuracy and speed.

*Corresponding Author: syafeeza@utem.edu.my

I. INTRODUCTION

Automatic License Plate Recognition (ALPR), also known as automobile or vehicle license plate recognition, is a system that uses image processing techniques to extract and recognize license plate characters from images or videos. ALPR has numerous applications, including electronic payment gateway systems, parking charge payment systems, road monitoring systems, and traffic control systems [1]. License plate characters come in various typefaces, sizes, and colors and can be positioned differently on vehicles. Hence, a robust algorithm is essential for the ALPR system to accurately detect and recognize these characters in real-world applications. The capability of the ALPR system to recognize the characters can be further complicated by environmental conditions such as brightness, weather, and objects [2]. Furthermore, the recognition rate may vary depending on the surrounding conditions, such as lighting and the background of the license plate [3].

ALPR typically involves four major stages, as shown in Figure 1. The first stage involves preprocessing the input, which could be either a still image or a video frame, to enhance the features in the image. In the second stage, the license plate area is identified, creating a sub-image that solely contains the plate. The third stage is the extraction phase, where each character on the license plate is extracted through image segmentation. These segmented characters are then normalized and passed to a character recognition algorithm. The final stage focuses on the production of the text form of the license plate characters. The characters are recognized using a specific algorithm for recognition purposes. The final output must be in the form of a string of characters.

High-quality images are essential for accurate license plate recognition. Factors such as the camera's make and model, resolution, lighting conditions, and image orientation during capture affect the image quality. In conventional ALPR systems, each of the four stages relies on different handcrafted algorithms. Typically, the final stage, character recognition, often employs traditional machine learning techniques like neural networks (NN) or Support Vector Machine (SVM). However, these classical systems often struggle with the diverse challenges posed by license plates. To overcome these limitations, deep learning approaches, particularly object detection methods like YOLO and SSD, have shown a promise. These methods integrate feature extraction and classification into a single trainable module, eliminating the need for handcrafted algorithms.

Consequently, networks employing deep learning techniques exhibit robustness in handling variations in input samples [4].

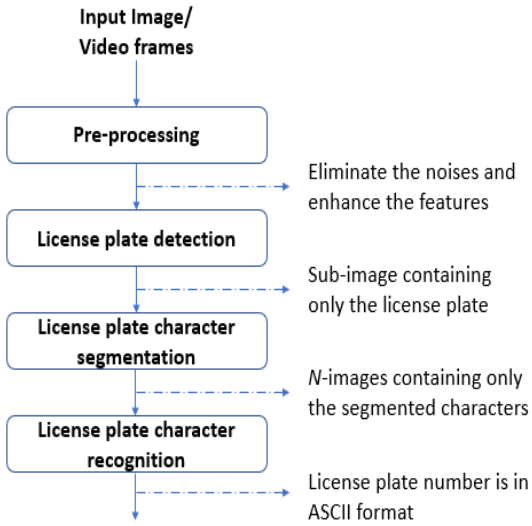

Figure 1. ALPR general flowchart

II. THEORY AND RELATED WORKS

A.YOLOv2

Deep convolutional neural networks have been successfully applied to object detection. YOLOv2 [5], also known as YOLO9000 due to its ability to detect more than 9,000 object categories, is an improved version of YOLO [6], a state-of-the-art, real-time object detection method based on deep learning. YOLOv2 achieves a mean average precision (mAP) of 76.8 mAP at 67 frames per second (FPS) on the PASCAL VOC 2007 dataset. It uses a fully convolutional network (FCN) consisting of 22 convolution layers and 5 pooling layers. This FCN-based feature extraction approach results in higher detection accuracy compared to other deep learning-based detection methods. The architecture of the YOLOv2 network is illustrated in Figure 2.

Type	Filters	Size	Output
Convolutional	32	3 × 3	
MaxPooling		1/2	128 × 128
Convolutional	64	3 × 3	
MaxPooling		1/2	64 × 64
Convolutional	128	3 × 3	
Convolutional	64	1 × 1	
Convolutional	128	3 × 3	
MaxPooling		1/2	32 × 32
Convolutional	256	3 × 3	
Convolutional	128	1 × 1	
Convolutional	256	3 × 3	
MaxPooling		1/2	16 × 16
Convolutional	512	3 × 3	
Convolutional	256	1 × 1	
Convolutional	512	3 × 3	
Convolutional	256	1 × 1	
Convolutional	512	3 × 3	
MaxPooling		1/2	8 × 8
Convolutional	1024	3 × 3	
Convolutional	512	1 × 1	
Convolutional	1024	3 × 3	
Convolutional	512	1 × 1	
Convolutional	1024	3 × 3	
——			
Convolutional	1024	3 × 3	
Convolutional	1024	3 × 3	
Concatonation	3072		
Convolutional	1024	3 × 3	S × S × (B × 5+C)

Figure 2. Network of YOLOv2

B.Related works of Malaysian plate recognition

Various approaches to Malaysian License Plate Recognition (MLPR) have been conducted, as shown in Table 1. The work in [7] utilized a Convolutional Neutral (CNN) for their MLPR design, focusing on developing a mobile application; however, specific details about the CNN model used were not provided. In [8], a two-stage approach combining YOLOv2 and ResNet-50 was employed, but the small size of the test images led to a recognition rate of less than 90%. The study in [9] achieved 92% accuracy, though is used a relatively small dataset of only 119 images. This highlights the rationale for using a pretrained ResNet18 model for their MLPR system.

Table 1
Comparison of the Deep Learning Method in MLPR Techniques

Ref.	Methods	Comments
[7]	Convolutional Neural Networks	Computational time of 0.635 seconds focusing on mobile app development
[8]	Two-stage YOLOv2-ResNet-50	404 Iraqi images and 681 Malaysian test images achieved average recognition rate of 85.56% and 88.86% on Iraqi and Malaysian datasets, respectively
[9]	Modified pretrained ResNet18	92% accuracy. A total of 119 images is used as the vanity license plate dataset images. There are a total of 38 images for vanity license plate type MALAYSIA, a total of 21 images for vanity license plate type PUTRAJAYA and a total of 60 images for normal license plates.

Table 2
Comparison of the Deep Learning Method in LPR Technique

Ref.	Methods	Comments
[10]	Two-stage YOLOv2	LP detection for clear weather 99% but 74% for night scene.
[11]	YOLOv4	Limited amount of training dataset (1000 images). Achieved 94.6% accuracy on the detection rate but recognition rate was not reported.
[12]	YOLOv6	The F1-score is 0.95. YOLOv6 is more suitable for industrial applications where speed is more important than accuracy.
[13]	SSD-MobileNetv1	SSD-based detector works efficiently with an accuracy of 94.87 %.
[14]	RPNet (Deep CNN)	Large China License plate dataset, CCPD (250k unique car images) with 95.5% precision and 61 fps.
[15]	Fast-YOLO	They introduced Brazilian license plates (FPR-ALP dataset). The system surpasses commercial solutions like Sighthound and OpenALPR by 93.53% accuracy and 47 fps.
[16]	Deep CNN	The result obtained is 0.993377 accuracy for 302 test images.
[17]	Deep CNN	LPDR was applied on country-specific plates, such as American or European, Chinese, Indian and Korean license plate. They achieved 99% detection and 93% recognition accuracy.

Table 2 presents a comparison of various deep learning approaches for license plate recognition (LPR) across different applications in simulated environments. The findings from these studies indicate that YOLO-based models

are commonly used. Although newer YOLO versions have introduced various enhancements, YOLOv2 was selected for its balance between simplicity, speed, and accuracy. All the works reported in Table 1 and Table 2 used self-collected images.

III. METHODOLOGY

MLPR is divided into two stages: detection and recognition. The detection phase consist of into three subphases: ground truth or bounding box annotation for the license plate region within the image, and two detection phases. Once the images in the dataset have been annotated, the network is trained using the CNN algorithm to learn the essential features. The optimal weights are obtained after the training phase is completed. These weights are crucial since they are used to analyze the detection accuracy and processing speed. Several parameters need to be considered before the training process to achieve the optimum result. In this case, the chosen parameter values for momentum and weight decay are 0.9 and 0.0005, respectively. In the second phase, the character recognition network takes the cropped image of the license plate region as input. In this phase, the 36 possible alphanumerical characters are labeled according to the character sequence in the image. The key difference between this work and [10] lies in the parameter settings and the dataset used. Figure 3 shows the flowchart of the MLPR system.

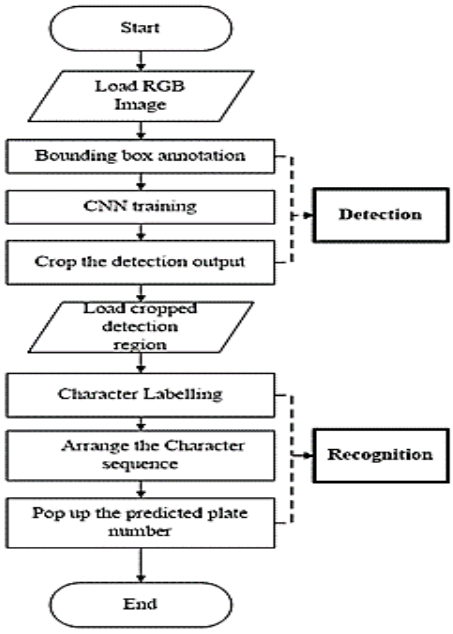

Figure 3. MLPR system flowchart.

System architecture

Figure 4 shows the overall system flow of the MLPR. The system consists of two main phases: "detect plates" (detection) and "recognize characters" (recognition). These two phases operate independently. In the first detection phase, the raw image input is annotated as a pre-trained dataset for the training process. The ground truth is then created in the plate region location within the image dataset. Once annotated, the file is converted into a darknet format, which is subsequently transformed into coordinates for further processing. After these steps, the dataset is ready for training. The CNN training process takes about one week to achieve the optimal weights required to detect the plate region. Once the plate region is successfully detected, the region is cropped based on their bounding boxes. The cropped image, with an additional margin to ensure no characters are missed, is then passed to the second phase, recognition.

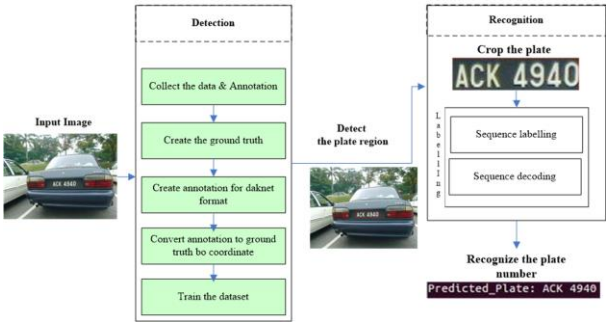

Figure 4. The overall system flow

In the second phase, the character recognition network takes the cropped license plate region image as input. In this phase, 36 possible alphanumerical characters are recognized. The training data comprises characters utilized on the plates but cannot recognize other characters found outside the license plate region. After the characters are detected, they are arranged according to the sequence of the alphabet within the image, a process called sequence labelling. The following step is sequence decoding. The predicted plate popped out on the terminal as the final result. A total number of 600 license plate images and around 11,200 characters were used to train both plate and character networks.

Architecture design

Both the plate and character networks are adaptations of the modified YOLOv2 architecture. The parameters differs from the original design to adapt to new datasets. The architecture design has been modified to ensure compatibility using different output classes. Originally, the YOLOv2 architecture was designed to predict 1,000 classes from ImageNet dataset. However, in this case, the plate and character networks need to predict only one and 36 classes, respectively. The final output number of nodes O is defined by the following equation:

$$O = S \times S * (B * 5) * C \quad (1)$$

where $S \times S$ represents the number of grid cells in the image, B is the number of bounding boxes predicted for each grid cell, and C is the number of possible classes. The factor B multiplied by 5 accounts for predicting the values of x,y,w,h, and the confidence score for each bounding box. The (x,y) is the coordinate's center of the bounding box. Meanwhile, w and h are the width and height of the bounding box, respectively. Finally, the confidence score, ranging from 0% to 100%, indicates how confident the bounding box model is, which contains the object and the accuracy of the box drawn. This confidence score is crucial to control the rate of false positives, compared to the overall prediction accuracy.

Different networks can be trained with different values of S to determine the number of grid cells ($S \times S$). Since training a single network needs significant time, S is chosen to optimize learning for both the training and test data. In the plate network, only the license plate needs to be detected, so even though multiple license plates might be found in the

image, only the closest one is detected, and $S = 1$ is set for this purpose. Since it is acceptable to have a higher false positives rate of plates, the default value of $S = 13$, which is a larger S, helps the network detect smaller plate images and reduces the chances of missed plates.

IV. RESULTS AND DISCUSSION

A. Evaluation

The comparison of the MLPR performance is shown in Table 3. The image conditions and the number of test images were recorded to measure the difficulty of the test set. Although accuracy rates are highly dependable on the difficulty of the test set, achieving a high overall accuracy does not indicate that the technique is superior. The performance comparisons also highlight the limitations and achievements of this technique.

The samples were collected with varying orientations, lighting conditins, and multiple viewpoints. Consequently, the prediction rates of the two methods cannot be directly compared. However, the performance speed remains comparable since the MLPR operates at a consistent speed. The proposed MLPR performs at a similar level to other methods. The inference speed is promising for both GPU and CPU execution.

Table 3
General comparison of MLPR performance

Ref.	Methods	Achievement
[7]	Convolutional Neural Networks	Accuracy: Unknown Speed: 0.635 seconds
[8]	Two-stage YOLOv2-ResNet-50	Accuracy: 88.86%
[9]	Modified pretrained ResNet18	Accuracy: 92%
MLPR	Two-stage YOLOv2 with 2200 Malaysian plates of 640×480 resolution captured in the parking area	Accuracy: 99.75% Speed: 0.0100 seconds

V. SPEED EVALUATION

A. GPU performance

As shown in Table 4, the average processing times for the plate and character networks using the Tesla k40c GPU were 0.0100 seconds and 0.0104 seconds, respectively. The processing time of the plate network varied very little, as evidenced by the minimal difference between the slowest and average times.

Table 4
The time taken when using Tesla K40 12GB GPU

Network	Time		
	Fastest time	Slowest time	Average time
Plate network	0.0102s	0.0127s	0.0100s
Character network	0.0099s	0.0109s	0.0104s

B. CPU performance

The CPU used in this project is not among the fastest, and the current CPUs may be capable of improving processing spee. The processing times for the plate and character networks were 7.985 seconds and 7.845 seconds, respectively, when running on Intel Xeon(R) 2.4GHz, as shown in Table 5. This speed is approximately 700 times slower than the running on GPU. Furthermore, most end-users experience significantly slower processing times on average.

Table 5
The time taken when using Intel Xeon(R) 2.4GHz CPU2

Network	Time		
	Fastest time	Slowest time	Average time
Plate network	7.78s	8.19s	7.985s
Character network	7.63s	8.06s	7.845s

Table 6 displays the experimental results for the plate network. The best results were achieved using a network trained on the extended training set. This network predicted each license plate correctly, with only one false positive, resulting in an overall accuracy of 99.75%. While it is difficult to eliminate errors completely, maintaining a false positive rate below 10% is necessary to prevent incorrect detections.

Table 6
Plate Detection result summary

Iteration	Training set	Threshold	Misclassified	False positive	Accuracy
1	800	75%	5	3	97.50%
2	1600	75%	0	1	99.75%

Table 7 displays the experimental results for the character network. Using the extended training set with a 75% threshold (within the 70-79% range) yielded a good accuracy. With a prediction rate of 97.8%, only 5 out of 400 plates were misclassified. The false positive rate was 0.24%, due to nine errors and one false positive. To avoid false positives, a threshold of over 75% is needed, though this increases the number of misclassified samples to 12. Figure 5 depicts successful plate detection and character recognition using MLPR.

Table 7
Character Recognition result summary

Iteration	Training set	Threshold	Misclassified samples	Accuracy
1	800	75%	8	97.50%
2	1600	75%	5	99.75%

Figure 5. Successful plate detection and character recognition in challenging conditions

VI. CONCLUSION

The primary objective of this research is to design a CNN-based object detection model utilizing a two-stage YOLOv2 approach for license plate recognition of 26 alphanumeric characters, with the goal of optimizing the performance and accuracy of MLPR. YOLOv2 was selected for its balance of simplicity, speed, and accuracy. The evaluation of the system's performance encompasses three crucial areas: plate detection accuracy, character recognition accuracy and processing speed on both CPU and GPU. The accuracy rates of plate detection and character recognition were 97.49% and 95.9%, respectively. Furthermore, the CPU and GPU were tested to measure the processing speed of the overall operation. The average processing times for plate and character recognition on the Tesla K40c GPU were 0.0100 seconds and 0.0104 seconds, respectively. Conversely, the processing times on an Intel Xeon(R) 2.4GHz processor for the plate and character networks were 7.985 seconds and 7.845 seconds, respectively. Based on the results obtained, this research achieves performance comparable to the best existing methods. In conclusion, the CNN algorithm adapted from the YOLOv2 approach has successfully enhanced MLPR performance and is suitable for real-world applications.

ACKNOWLEDGMENT

The authors would like to acknowledge the support from the Machine Learning and Signal Processing (MLSP) research group under the Centre for Telecommunication Research & Innovation (CeTRI), Fakulti Teknologi dan Kejuruteraan Elektronik dan Komputer (FTKEK), Universiti Teknikal Malaysia Melaka (UTeM).

CONFLICT OF INTEREST

Authors declare that there is no conflict of interests regarding the publication of the paper.

AUTHOR CONTRIBUTION

The authors confirm contribution to the paper as follows: Data collection and preparation: Norihan Abdul Hamid; Literature survey and review: Asar Khan; Training and model design: Marzuki P. Ramli; Parameter hypertuning: Wira Hidayat Mohd Saad; Draft manuscript preparation: Syafeeza Ahmad Radzi and Airuz Sazura A. Samad. All authors reviewed the results and approved the final version of the manuscript.

REFERENCES

[1] W. Zhou, H. Li, Y. Lu, and Q. Tian, "Principal visual word discovery for automatic license plate detection," IEEE Trans. Image Process., vol. 21, no. 9, pp. 4269–4279, 2012, doi: 10.1109/TIP.2012.2199506.

[2] S. A. Radzi and M. Khalil-Hani, "Character recognition of license plate number using convolutional neural network," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 7066, pp. 45–55, 2011, doi: 10.1007/978-3-642-25191-7_6.

[3] Y. Wen, Y. Lu, J. Yan, Z. Zhou, K. M. Von Deneen, and P. Shi, "An algorithm for license plate recognition applied to intelligent transportation system," IEEE Trans. Intell. Transp. Syst., vol. 12, no. 3, pp. 830–845, 2011, doi: 10.1109/TITS.2011.2114346.

[4] V. Jain, Z. Sasindran, A. Rajagopal, S. Biswas, H. S. Bharadwaj, and K. R. Ramakrishnan, "Deep automatic licence plate recognition system," ACM Int. Conf. Proceeding Ser., 2016, doi: 10.1145/3009977.3010052.

[5] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in Proc. of 30th IEEE Conf. Comput. Vis. Pattern Recognition (CVPR 2017), 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in Proc. of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[7] A. S. F. Gani et al., "A live-video automatic number plate recognition (ANPR) system using convolutional neural network (CNN) with data labelling on an android smartphone," Int. J. Emerg. Technol. Adv. Eng., vol. 11, no. 10, pp. 88–95, 2021, doi: 10.46338/ijetae1021_11.

[8] D. Habeeb et al., "Deep-learning-based approach for Iraqi and Malaysian vehicle license plate recognition," Comput. Intell. Neurosci., vol. 2021, pp. 1–14, 2021, doi: 10.1155/2021/3971834.

[9] H. H. Tan, R. Shahid, M. Mishra, and S. L. Lim, "Malaysian vanity license plate recognition using convolutional neural network," Int. J. Technol., vol. 13, no. 6, pp. 1271–1281, 2022, doi: 10.14716/ijtech.v13i6.5868.

[10] S. Yonetsu, Y. Iwamoto, and Y. W. Chen, "Two-stage YOLOv2 for accurate license-plate detection in complex scenes," in Proc. of 2019 IEEE Int. Conf. Consum. Electron. (ICCE 2019), 2019, pp. 1–4, doi: 10.1109/ICCE.2019.8661944.

[11] C. L. Chang, P. J. Chen, and C. Y. Chen, "Semi-supervised learning for YOLOv4 object detection in license plate recognition system," J. Imaging Sci. Technol., vol. 66, no. 4, pp. 1–9, 2022, doi: 10.2352/J.ImagingSci.Technol.2022.66.4.040404.

[12] M. Li and L. Zhang, "Deep learning-based license plate recognition in IoT smart parking systems using YOLOv6 algorithm," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 12, pp. 222–231, 2023, doi: 10.14569/ijacsa.2023.0141223.

[13] N. Awalgaonkar, P. Bartakke, and R. Chaugule, "Automatic license plate recognition system using SSD," in Proc. of 2021 Int. Symp. Asian Control Assoc. Intell. Robot. Ind. Autom. (IRIA 2021), 2021, pp. 394–399, doi: 10.1109/IRIA53009.2021.9588707.

[14] Z. Xu et al., "Towards end-to-end license plate detection and recognition: a large dataset and baseline," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 11217, pp. 261–277, 2018, doi: 10.1007/978-3-030-01261-8_16.

[15] R. Laroca et al., "A robust real-time automatic license plate recognition based on the YOLO detector," Int. Jt. Conf. Neural Networks, vol. 2018-July, 2018, doi: 10.1109/IJCNN.2018.8489629.

[16] H. H. Kim, J. K. Park, J. H. Oh, and D. J. Kang, "Multi-task convolutional neural network system for license plate recognition," Int. J. Control. Autom. Syst., vol. 15, no. 6, pp. 2942–2949, 2017, doi: 10.1007/s12555-016-0332-z.

[17] S. M. Silva and C. R. Jung, "Real-time Brazilian license plate detection and recognition using deep convolutional neural networks," in Proc. of 30th Conf. Graph. Patterns Images (SIBGRAPI 2017), 2017, pp. 55–62, doi: 10.1109/SIBGRAPI.2017.14.