# Project report for OVO

Corentin Davion*

February 2025

**Abstract**

This report focus on the article Pseudo-Boolean Polynomials Approach to Edge Detection and Image Segmentation by Tendai Mapungwana Chikake, Boris Goldengorin and Alexey Samosyuk [4]. In the article, the authors suggest a novel method to represent image patch and construct a criterion with said representation to affirm if the patch contains an edge or not. In this report, we will explain their contribution with pseudo-boolean polynomials and situate their approach in the context of computer vision,recreated their experiment and ensure it's generalization.

## 1 Introduction

Edge detection consists of identifying the boundaries between two different component of the image. It's a fundamental task essential for more sophisticated image processing such as segmentation or object recognition [5]. Traditional method, starting in the 1960's mainly used gradient technique such as the Canny filter [3] whereas more recent technique used the laplacien coupled with deep learning models [6] to improved the accuracy. However, these methods remains computationally expensive, which is why simple algebraic methods are still widely used. In the paper [4], the author introduce the usage of pseudo Boolean polynomials to analyze image patches in order to detected eventual edge in said image. This approach is deterministic, can be adapted easily and especially does not requires much compute.

## 2 Methodology

In this section we will present the methodology used by the authors,explain the motivation behind it and provide custom example to a better understanding.

### 2.1 Motivation

An image is usually represented by a $m \times n \times c$ tensor, where $c$ is the number of channel. For illustration purposes, we will consider that our images are gray-scales, so $c = 1$ and every entry is a integer ranging from 0 to 255. The core idea behind edge detection is that edges are variations in the image, so the goal is to find a method that can detect if a image patch has a variations or not, hence the early gradient technique such as [3]. We will see down below that a image patch that present a strong variations will generate a pseudo Boolean polynomial of higher degree, which will allow us to assert if said image patch contains an edge.

### 2.2 Construction of the mathematical objects

A pseudo Boolean polynomials is a function $f \quad \{0,1\}^n \mapsto \mathbb{R}$. It can be uniquely represented as a (see [1] and reference within) as multi-linear polynomials of the form

$$f(y) = \sum_{S \subset [1 \dots n]} c_S \prod_{i \in S} y_i$$

*Student at CentraleSupelec

Given a image patch $C = [c_{ij}]$ for $i \leq m$ and $j \leq n$, we build for $C[:\, j]$ the j-th column of $C$ = a permutation $\Pi^j = (\pi_1^j ... \pi_n^j)$ a permutation that sorts $C[:\, j]$ in ascending order, when read from top to bottom ($c_{\pi_l^j j} < c_{\pi_p^j j}$) for $l \leq p$, ties broken arbitrarily. We then compute the difference column $\delta^j$ where $\delta_1^j = c_{\pi_1^j j}$ and $\delta_r^j = c_{\pi_r^j j} - c_{\pi_{r-1}^j j}$, we subtracted the next row from the previous, leaving the first row untouched. The final pseudo Boolean polynomial is the following for this column in variables $y_1 ... y_m$

$$f_C^j(y) = +\sum_{k=1}^{m} \delta_k^j \prod_{r=1}^{k-1} y_{\pi_r^j}$$

with the usual convention that the empty product yield the unity. Below is a example of the construction process.
Consider the $3 \times 3$ image patch

$$C = \begin{bmatrix} 7 & 7 & 3 \\ 2 & 4 & 5 \\ 8 & 3 & 6 \end{bmatrix}$$

The index ordering will be the matrix

$$\Pi = \begin{bmatrix} 2 & 3 & 1 \\ 1 & 2 & 2 \\ 3 & 1 & 3 \end{bmatrix}$$

So after sorting, the difference matrix is

$$\Delta C = \begin{bmatrix} 2 & 3 & 3 \\ 5 & 1 & 2 \\ 1 & 3 & 1 \end{bmatrix}$$

The term matrix will be

$$\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} \\ y_2 & y_3 & y_1 \\ y_1 y_2 & y_2 y_3 & y_2 y_1 \end{bmatrix}$$

Taking the Hadamard product and summing everything yield the final polynomial

$$f_C = 8 + 2y_2 + 1y_3 + 3y_1 + 2y_1 y_2 + 3y_2 y_3$$

Which is a more compact form because we only used 6 'variable' instead of 9.

## 2.3 Discrimination criteria

In the previous section, we a compact way to represented an image patch. From it, we need to build a easily computable classification function that can be used determined if an image patch contains an edge or it's a blob. We noticed that if a image patch is of constant value $a$, then the resulting polynomial is $f_C = n \times a$ of degree 0 where $n$ is the size of the patch. Hence the following classifier by truncation where $d$ is the degree of the polynomial computed.

$$f(d, p) = \begin{cases} \text{edge if } d > p \\ \text{blob otherwise} \end{cases}$$

Where $p$ is parameter that can be tuned depending the wished sensitivity: a higher value of $p$ will make the detection less prone to flag nonsignificant edge, whereas a small value will flag the smallest contour. Notice that the construction of the pseudo-Boolean polynomials depicted only tackle edges in one direction, so we simply compute the pseudo-boolean polynomials on the transpose of the patch. For illustration, consider

$$C = \begin{bmatrix} 7 & 4 & 3 \\ 7 & 4 & 3 \\ 7 & 4 & 3 \end{bmatrix}, \text{the corresponding polynomials is } f_C = 14$$

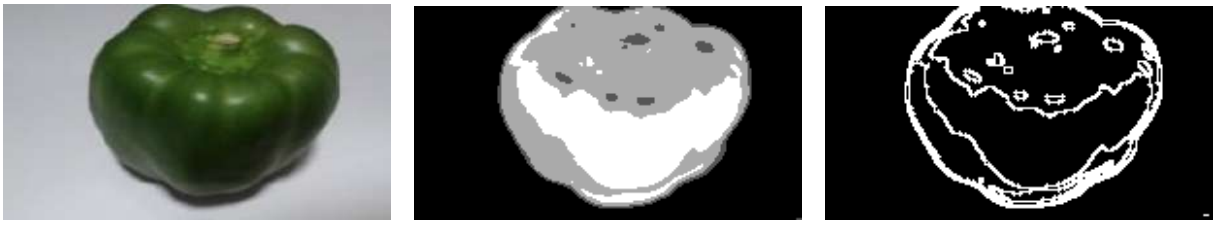
Which means no edge are form in the vertical sense, However, if we transpose it

$$C^T = \begin{bmatrix} 7 & 7 & 7 \\ 4 & 4 & 4 \\ 3 & 3 & 3 \end{bmatrix} \text{ the corresponding polynomials is } f_{C^T} = 9 + 3y_3 + 9y_2y_3$$

which means that there is edge in the horizontal direction.

# 3 Numerical simulation

The images first gets resized before undergoing a pre-processing step. Natural images tends to have a smooth transition of pixel due to anti-aliasing. For this, the images are first resized and undergoes a Gaussian filter to blend the colors a bit followed by a color aggregation step, which essentially is a quantization process ([2] suggest some algorithms, but in the current implementation, a simple formula was used), then the segmentation techniques is applied. Below is a few example, with the original images,the results of the pre-processing step and the output of the algorithm. The time is measured for the whole pipeline, on a Intel i9-13900-HX. No CUDA implementation was used.



(a) Original image (b) After pre-processing (c) Suggested output

Figure 1: Fruit (image implementation [4]) time taken: 1.55s

The figure below compares the results of the pseudo-Boolean-approach with a Sobel filter and the Laplacien method. Those images were generate at `https://pinetools.com/image-edge-detection`. We see that the pseudo Boolean approach does not necessarily flag more edges, and shape of the object with the figures is quite distinguishable. Refer to the appendix for more example. For images with rather simple

(a) Suggested output (b) Sobel operator (c) Laplacien

Figure 2: Comparison with different method

composition, the methods performs quite well, and even in video game screenshots, it's able to capture a significant amount of edges. Comparing the output with the pre-processing stages suggests that the amount of details captures in the pre-processing steps is also captured in the output. We see that this methods fails to produce a qualitative output when the images consist of granulated texture, but this is also captured with methods such as the Sobel filter.

# 4 Conclusion

The article [4] by Chikake et al suggested a new method for edge detection by computing the pseudo-Boolean polynomials over the image patches. This approach is both fast and accurate, as showcased by the numerous example, and is also very interpretable as the computation of the polynomials is explicit. The method is also flexible, depending on the value of the $p$ parameters. The method is also highly parallelizable, which can further increase the speed without losing accuracy as each polynomials can be computed independently over the image patches. Despite it's efficiency, it is still a edge detection method, meaning that it takes place in a broader pipeline of a computer vision task. Nevertheless, the approach consisting of using boolean polynomials is useful to explore and expend in further categories such as images recognition.

# References


[1] Bader AlBdaiwi, Diptesh Ghosh, and Boris Goldengorin. Data aggregation for p-median problems. *Journal of Combinatorial Optimization*, 21:348–363, 04 2009.

[2] S. Arora, J. Acharya, A. Verma, and Prasanta K. Panigrahi. Multilevel thresholding for image segmentation through a fast statistical recursive algorithm. *Pattern Recognition Letters*, 29(2):119–125, 2008.

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

[4] Tendai Mapungwana Chikake, Boris Goldengorin, and Alexey Samosyuk. *Pseudo-Boolean Polynomials Approach to Edge Detection and Image Segmentation*, pages 73–87. Springer Nature Switzerland, Cham, 2023.

[5] Paul King. Digital image processing and analysis: Human and computer applications with cviptools, 2nd edition [book reviews]. *Pulse, IEEE*, 3:84–85, 07 2012.

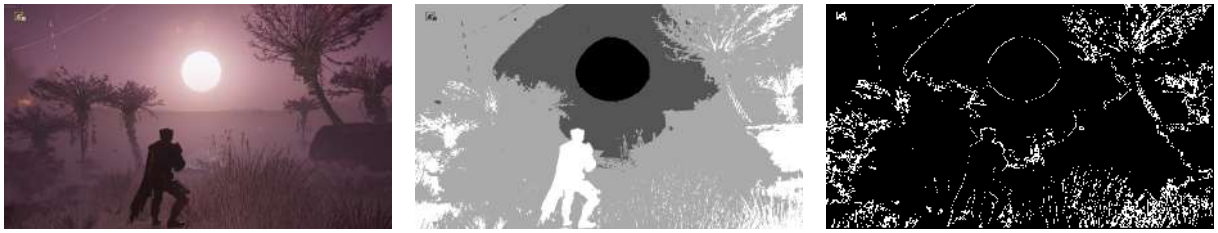
[6] Kanija Muntarina, Rafid Mostafiz, Sumaita Binte Shorif, and Mohammad Shorif Uddin. Deep learning-based edge detection for random natural images. *Neuroscience Informatics*, 5(1):100183, 2025.

# Appendix

(a) Original image (b) After pre-processing (c) Suggested output

Figure 3: Flower (image implementation [4]) time taken: 2.75s

We see that the shape is rather well captured, but also some background details



(a) Original image (b) After pre-processing (c) Suggested output

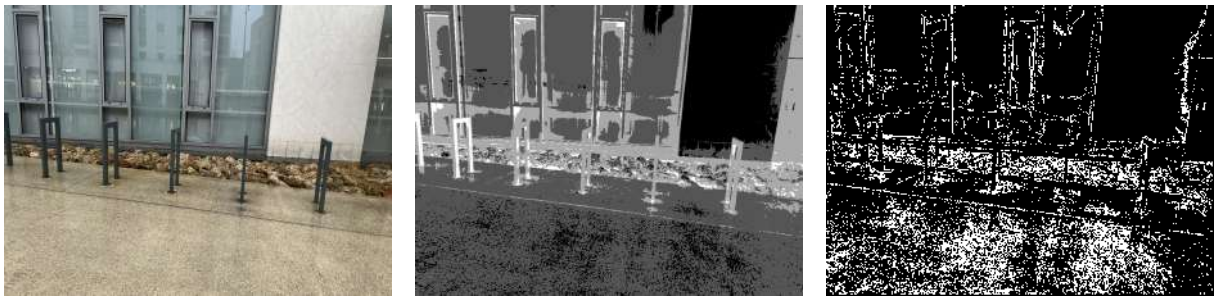
Figure 4: Screenshot of Helldivers 2[1]. Time taken: 39.75s

(a) Original image (b) After pre-processing (c) Suggested output

Figure 5: Screenshot of Wuthering Waves[2]. Time taken: 40.05s

We see that in when dealing with game screenshots, the lightning and other rendering effects greatly affects the pre-processing steps, hence the edges are objects in the scene are less refineds: the most noticeable is the sun 'halo effect'.

[1]HELLDIVERS is trademark of SONY INTERACTIVE ENTERTAINMENT LLC
[2]Wuthering Waves is a product of Kurogame Technology (Guangzhou) Co., Ltd.



(a) Original image (b) After pre-processing (c) Suggested output

Figure 6: Photo taken on a rainy day[3]. Time taken: 31.75s

We see that a lot of artifact on the ground, due to the pre-processing steps highlighting the texture of the ground.

(a) Sobel operator (b) Laplacien method (c) Suggested output

Figure 7: Photo taken on a rainy day, comparing with different methods