A Review of Modern Edge Detection Techniques

ISAAC WOODARD[1]

[1] *Knight Campus Graduate Internship Program, University of Oregon, 1585 E 13th Ave, Eugene OR 97403*
[*] *isaac.j.woodard@gmail.com*

Abstract: Edge detection (ED) is one of the fundamental problems of image processing. The detection of edges is a first step towards identifying structures in an image, paving the way for subsequent analysis. Two of the most common and robust edge detectors are the Laplacian of Gaussian edge detector and the Canny edge detector. Both of these detectors utilize both Gaussian smoothing to reduce noise and a multi-scale approach to detect edges on different scales. In the semiconductor industry, ED is used as the basis for contour extraction and can be used for classification of defects through machine learning. While exploration of modern applications is confined in this paper to the semiconductor industry, many other applications for ED exist and a few are briefly mentioned.

1. Introduction

Edge detection (ED) is the identification of features in an image which correspond to physical or visual edges. Practically speaking, edges are identified based on sharp changes in pixel values which can be found by applying local gradient filters to the image. These filters can provide information on not only the magnitude of the gradient but also it's direction. Common names for these quantities are "edge strength" and "edge orientation".

Identifying edges in an image is often a first step towards enabling computer vision–the identification of image structures with digital algorithms. In many cases, edges carry all the information needed to identify objects in an image [1]. Extracting edges removes noise and superfluous data while leaving behind details important for identification of structures such as biological cells and circuit components.

The fields of computer vision and image processing use edges as one of the main types of image features. Other main types are corners and blobs. For visualization purposes, corners can be thought of as points while blobs are two-dimensional regions [2]. Edges are essentially one-dimensional, though they may have some thickness depending on how they are processed. There can be overlap in the identification of these three types of features: a blob might be defined as the region surrounded by an edge and a corner as the junction between multiple edges [3].

Studies on ED have been partly inspired by research on human and other biological vision [3,4]. Indeed, edge definitions originate from our qualitative perception of them. It can be difficult to provide quantitative rather than qualitative definitions for an edge. Rather than use ED algorithms to provide empirical edge definitions, qualitative assessment is usually performed to decided if edges are being properly detected.

Nonetheless, there are several idealized classifications for edges, a few of which are shown in Fig. 1. Perhaps the most common is the step edge. As the name suggests, this type of edge has a profile characterized by a step function. Other types of edges include ridge, roof, and staircase edges [5,6]. A number of factors prevent edges in an image from taking on these ideal forms. Instead, profiles are noisy and show some smoothing inherent in the act of image capture [5].

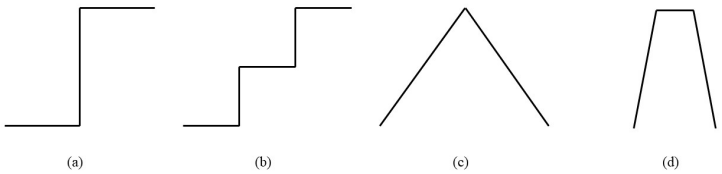

Fig. 1. Idealized edge profiles in one dimension for a step (a), a staircase (b), a roof (c), and a ridge (d). Height represents pixel intensity.

At the core of most ED techniques is the application of filters to an image. A filter is an operation which uses values from multiple pixels in the image to process a new value for each pixel [1]. This is done using a kernel: a matrix which is convolved at every pixel in the image. For example, the result in Eq. 1 is the linear convolution of a 3x3 kernel with a 4x4 8-bit image,

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 0 & 50 & 255 & 100 \\ 150 & 255 & 150 & 50 \\ 100 & 100 & 255 & 50 \\ 0 & 100 & 255 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 50 & 255 & 100 \\ 150 & 705 & 965 & 50 \\ 100 & 810 & 810 & 50 \\ 0 & 100 & 255 & 1 \end{bmatrix}. \tag{1}$$

An important detail is missing in that the kernel cannot be properly convolved at the boundaries of the image matrix. Fortunately, there are a variety of suitable techniques for overcoming this problem–in essence by extending the image matrix [1]. Another issue is that the result has pixel values greater than the 8-bit maximum of 255; this is addressed by normalizing the kernel, as shown in Eq. 2,

$$\begin{bmatrix} 0.0 & 0.2 & 0.0 \\ 0.2 & 0.2 & 0.2 \\ 0.0 & 0.2 & 0.0 \end{bmatrix}. \tag{2}$$

ED algorithms generally begin with a smoothing filter to reduce false edge detection due to noise [5]. Preliminary ED is then performed by applying one or more gradient filters. An edge map is created which can give information on the edge strength and edge orientation at each pixel [4]. Using an edge definition such as an intensity threshold, edges can then be extracted.

One of the challenges of ED is that edges can exist at different scales [1]. The notion of scale emerges from the fact that kernels have a finite size. This means the operations of a filter take into account a particular region of pixels, whereas the gradient in the image corresponding to an edge might be broader or narrower. As a result, a single gradient filter may not be suitable for detecting all the desired edges in an image [3].

2. Current State of the Art

With the basics of ED established, let's consider common practices for modern applications. Any modern edge filter typically begins with a noise reduction step. Different approaches can be used, but the most common is to smooth the image–like a blurring effect. For this purpose a variety of filters are available. In general, they operate by computing a weighted average of the pixel values around each pixel [1].

Historically, the most common smoothing filter appears to be the Gaussian. In the early development of ED, Marr and Hildreth argued Gaussian smoothing is optimal based on it's localized and smooth impulse [3]. This remains largely true for it's implementation with a kernel.

Locality is defined by the extent of the filter kernel, and smoothness is approximated by the discretization of a function into the kernel elements.

Gaussian filters are still commonly used for smoothing, but other approaches have been developed as well. Amoako-Yirenkyi et al. compared the performance of Gaussian, median, and cubic spline smoothing filters [7]. They introduced different types of noise to an image and tested each filters' ability to remove the noise while also minimizing distortion to the image. The metric of comparison was the Structural Similarity Index Measure, which gave a quantitative comparison of the filtered result to the original image. It was found the Gaussian filter performed best with speckle noise, the median filter performed best for salt-and-pepper noise, and the cubic spline filter performed best for noise due to motion blurring.

Of course, a key consideration apart from performance is efficiency. A study by Schneider et al. on contour extraction in semiconductor scanning electron microscopy (SEM) imaging compared the results of Gaussian, Median, Nagao and NL Means smoothing filters [8]. The NL Means filter provided the bests results for subsequent ED but also had a runtime three orders of magnitude greater than a Gaussian blur.

Following noise reduction, edges can be processed using one or more gradient filters. These filters approximate the local derivatives between pixel values to determine the edge strength at each pixel. After edge strength is calculated it is necessary to decide where the edges are located to create the edge map of the original image. Edges can be defined with sub-pixel accuracy but simpler methods use an intensity threshold to determine whether each pixel is an edge point.

A long-standing ED algorithm is the Laplacian of Gaussian (LoG) developed by Marr and Hildreth in 1980 [3]. The first of its two key features is the use of multiple smoothing kernels of different sizes. Edges are detected from the result of each smoothing operation separately, and the final edge map is compiled from the results of the individual edge maps. This multi-scale approach makes the LoG particularly robust.

It's second key feature is the use of second-derivatives in the image intensity rather than first-derivatives. This has both advantages and downsides. The identification of zero-crossings in the second derivative of the intensity produces edges of minimal thickness without the need for an additional edge thinning step [4]. By default, it also enables tracing of complete contours. However, the second derivative is more sensitive to noise and techniques required to compensate for the sensitivity run the risk of breaking the contours [4].

One of the most popular ED algorithms is the Canny edge detector; it was developed by John Canny in 1986 [6]. Like the LoG, it uses a multi-scale approach. However, instead of using multiple smoothing kernels, it uses multiple edge detection kernels at different scales [5]. These edge detection kernels can be based on either the first derivative or the second derivative of the image intensity [4]. Duplicate detection of edges is avoided by comparing the response of the gradients at each scale. Canny's implementation uses a Gaussian filter for smoothing [6].

In a study on implementing ED in 32-bit single chip microcontrollers, Budzyn and Rzepka compared the performance of the Canny and LoG ED algorithms [9]. They also considered the performance of so-called Fuzzy Logic and Cost Function algorithms. Their Canny and LoG implementations had the best runtime, with the LoG being slightly faster. Qualitatively, the Canny and LoG implementations also showed the best edge detection. By comparing the positions of detected edges to caliper measurements of the imaged object, they found the Canny implementation was the most accurate overall. Critical issues with the other algorithms was found, with the Fuzzy Logic implementation failing to detect complete object contours and the Cost Function implementation requiring vastly increased runtimes.

An important step towards using ED in computer vision is the composition (or extraction) of edges into contours; this is often critical for identifying objects and allows for separate regions of the image to be defined. The terms "contour" and "edge" are sometimes used interchangeably. Strictly speaking, an edge is the result of an edge filter and is composed of a collection of

points which represent the edge [1]. A contour on the other hand is a complete trace around an object. This trace is built out of multiple edge points and may even require stitching of multiple segmented edges as shown in Fig. 2 [10]. In general, any contour can be treated as an edge, but an edge is not necessarily a contour.

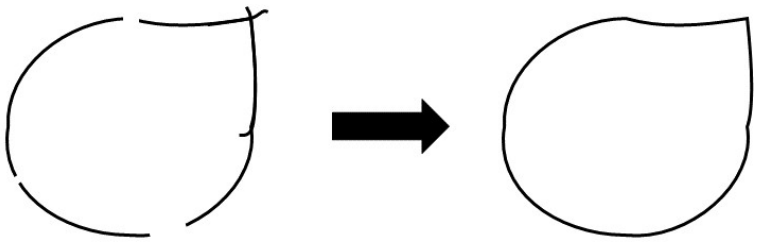

Fig. 2. Composition of several edges into a single contour.

3. Technology

ED techniques have seen extensive use in the semiconductor industry. They allow automation of visual measurements which would otherwise require significant, or even prohibitive, amounts of human labor. In addition, they provide a basis for making these measurements repeatable by either eliminating or quantifying the subjective judgement of operators. This is a key advantage in an industry where quantitative analysis is essential.

A specific area of the industry where ED is applied is in critical dimension scanning electron microscopy (CD-SEM) technology. CD-SEM instruments are outfitted for bulk metrology of wafer patterning. They build upon ED algorithms with contour extraction and alignment of repeated patterns for statistical comparison of contours [11].

An interesting facet of this technology is the ability to distinguish between top and bottom features in wafer patterns. Studies have been conducted which illustrate this. A study by Schneider et al. used a tagging system to identify top and bottom areas in an image [8]. Unique tags were given to each edge, slope, and planar region. With the knowledge that a top and bottom region must be separated by a slope, the regions were marked with alternating colors. Assumptions on average intensity were then used to decide which color of regions corresponded to top or bottom.

Another study by Zhou et al. took an alternative approach to distinguish between top and bottom edges [12]. For each slope, there is an edge at the top of the slope and the bottom. They characterized each edge as part of either an inner or an outer contour. At this point, assumptions have to be made on which level the inner and outer contours correspond to. For the patterns in their analysis, bottom regions were surrounded by top regions, so inner contours corresponded to the bottom level and outer contours to the top.

The results of contour extraction can be taken a step further by classifying image features to identify different types of defects. Schneider et al. described a basis for this by arranging images batches of replicated wafer patterns in a phylogenetic tree [8]. Differences between patterns were quantified by overlaying pairs of images and calculating the overall distance between the extracted features from each. This provided a distance matrix which they used to create a hierarchical tree separating the individual images into branches. These branches showed patterns of similar structure, thus grouping defects based on shape.

Machine learning can also be used for defect classification. Building off of computer vision techniques, machine learning approaches for image processing often use convolutional neural networks (CNN) to perform image processing algorithms in response to the CNN's training. With contour extraction as an intermediate step, the CNN can identify structures in an image based on it's training.

Beuth et al. proposed a stacked hybrid convolutional neural network (SH-CNN) for qualifying silicon wafer chips [13]. The main target of the SH-CNN was the classification of each chip on a wafer as flawless, anomalous, or faulty. Chips were classified based on the quality of the "streets" separating them from other chips. Streets were first identified using ED and contour extraction before finally comparing to an expected template. With the streets identified, their SH-CNN was able to perform the desired classification on each chip.

4. Future Directions

Partly inspired by human vision, ED has seen much use since it's initial development. At the core of the practice are many long-standing algorithms, such as the Canny edge detector and the LoG. However, there is also much room for growth in advanced applications: composition of edges into contours, exploration of effective while efficient noise reduction techniques, and the application of ED in computer vision and machine learning.

The semiconductor industry has seen much use out of ED to automate optical inspection. While the inspection of wafer patterning has been explored, post-processing steps bear further research. In particular, the small scale of today's circuit boards has driven the use of microvias to connect multiple layers. Analysis of these vias is critical for avoiding fabrication failures and their small size demands high magnification imaging techniques. CDSEM technology and studies on SEM imaging show promise for performing analysis on microvias–particularly blind microvias where both a bottom and top region are likely of interest. Zhou et al. demonstrated a possible application for groups of static RAM vias by overlaying via contours and comparing cross section statistics [12].

There are also many modern applications of ED which lie outside the semiconductor industry. The field of microbiology has a great need for effective ED techniques for the automated analysis of cells [14]. Applying ED to the identification of microorganisms poses different challenges to those encountered in semiconductor imaging. Edges are often less defined and there are additional limitations to imaging methods to avoid damaging the biological structure.

Two other examples of ED applications are illustrated in a study on depth-of-field ED and a study which used three-dimensional rather than two-dimensional ED. For both semiconductors and cells, structures mostly extend along two rather than three dimensions. Wang et al. describe an ED method, however, for images with significant depth of field, such as might be encountered in robot navigation [15]. In a similar vein, Yu et al. present an ED approach using a three-dimensional point cloud of data obtained from confocal laser measurement [16].

Clearly, the use of ED depends greatly on the specific context of the imaging. While the same basic ED principles still apply, different approaches may be required to produce consistent detection of edges for different imaging situations.

References

1. W. Burger and M. J. Burge, *Principles of Digital Image Processing: Fundamental Techniques* (Springer Publishing Company, Incorporated, 2009), 1st ed.
2. T. Lindeberg, "Feature detection with automatic scale selection," Int. J. Comput. Vis. **30**, 77–116 (1998).
3. D. Marr and E. Hildreth, "Theory of edge detection," Proc R Soc Lond B Biol Sci **207**, 187–217 (1980).
4. J. D. Gibson and A. Bovik, *Handbook of Image and Video Processing* (Academic Press, Inc., USA, 2000), 1st ed.
5. D. Ziou and S. Tabbone, "Edge detection techniques-an overview," (1998).
6. J. Canny, "A computational approach to edge detection," IEEE Transactions on Pattern Analysis Mach. Intell. **PAMI-8**, 679–698 (1986).
7. P. Amoako-Yirenkyi, J. K. Appati, and I. K. Dontwi, "Performance analysis of image smoothing techniques on a new fractional convolution mask for image edge detection," Open J. Appl. Sci. **06**, 478–488 (2016).
8. L. Schneider, V. Farys, E. Serret, and C. Fenouillet-Beranger, "Framework for sem contour analysis," in *Metrology, Inspection, and Process Control for Microlithography XXXI,* vol. 10145 M. I. Sanchez, ed., International Society for Optics and Photonics (SPIE, 2017), pp. 376 – 388.
9. G. Budzyn and J. Rzepka, "Review of edge detection algorithms for application in miniature dimension measurement modules," J. Mach. Eng. **20**, 74–85 (2020).

10. X.-Y. Gong, H. Su, D. Xu, Z. Zhang, F. Shen, and H.-B. Yang, "An overview of contour detection approaches," Int. J. Autom. Comput. **15**, 1–17 (2018).
11. B. Le-Gratiet, R. Bouyssou, J. Ducote, A. Ostrovsky, C. Beylier, C. Gardin, N. Schuch, V. Annezo, L. Schneider, M. Millequant, P. Petroni, T. Figueiro, and P. Schiavone, "Contour based metrology: "make measurable what is not so"," in *Metrology, Inspection, and Process Control for Microlithography XXXIV,* (SPIE, San Jose, United States, 2020), Metrology, Inspection, and Process Control for Microlithography XXXIV, p. 3.
12. K. Zhou, X. Guo, W. Zhou, Q. Wan, C. Du, W. Wu, A. Chen, R. Zhang, G. Fenger, S. Rampoori, and B. Samir, "Contour-based process characterization, modeling, and control for semiconductor manufacturing," in *Advanced Lithography,* (2021).
13. T. Schlosser, F. Beuth, M. Friedrich, and D. Kowerko, "A novel visual fault detection and classification system for semiconductor manufacturing using stacked hybrid convolutional neural networks," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA),* (2019), pp. 1511–1514.
14. T. Gebäck and P. Koumoutsakos, "Edge detection in microscopy images using curvelets," BMC bioinformatics **10**, 75 (2009).
15. X. Wang, J. Cao, Q. Hao, K. Zhang, Z. Wang, and S. Rizvi, "Lbp-based edge detection method for depth images with low resolutions," IEEE Photonics J. **11**, 1–11 (2019).
16. W. Yu, Y. Feng, X. Li, and J. Tang, "A laser scanner based in-process inspection approach of thru-hole for drilling robotic system," J. Adv. Control. Autom. Robotics (JACAR) **4**, 136–144 (2018).