# Edge Detection and Polygon Creation in Image Processing

Dr. Azimifar
CSE & IT, Shiraz University

Teaching Assistants:
Sina Hakimzadeh, Mohammad Yasin Farshad, Raha Rahmanian, Sara Shahriyary

Fall 2024

In this project, you will be exploring the world of image processing, which is an essential field in computer vision. Image processing allows machines to interpret and understand visual data in ways that are useful for tasks such as object recognition, facial recognition, and even autonomous driving. One of the key operations in image processing is edge detection, which helps identify the boundaries of objects within an image. Detecting these boundaries is crucial for many applications, as it allows us to recognize shapes, contours, and structures.
This project focuses on processing an image to detect edges and extract the main object's contour as a polygon.

Figure 1: Original Image

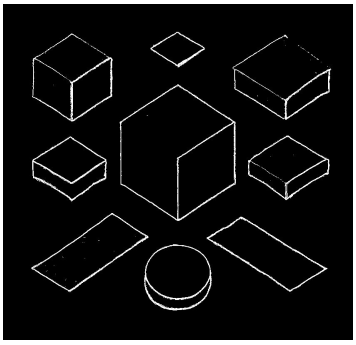

Figure 2: Edge Detection

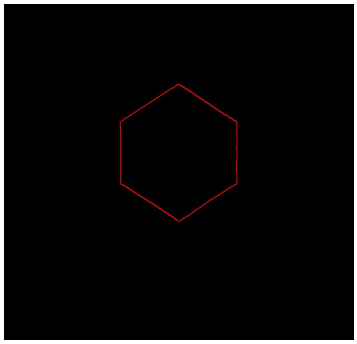

Figure 3: Polygon Representation

Figure 4: Flow of Work: From Original Image to Polygon Representation

## 1 Phase 1: Opening an Image and Converting It to Grayscale

The first step is to load an image and convert it into grayscale. Images typically come in RGB (Red, Green, Blue) format, but edge detection works best on single-channel images. Converting the image to grayscale simplifies the processing by removing color information. To achieve this, you will load the image and convert it by averaging the three color channels (Red, Green, Blue) or using a weighted sum that reflects human perception (e.g., 0.299R + 0.587G + 0.114*B). This will prepare the image for the subsequent edge detection process.

## 2 Phase 2: Applying Sobel Edge Detection

In this phase, you will apply the Sobel edge detection technique to find edges in the grayscale image. Sobel edge detection works by calculating the gradients of pixel intensities in both the horizontal and vertical directions. These gradients help identify areas where there is a significant change in pixel intensity, which usually indicates an edge.

The Sobel operator consists of two 3x3 kernels: one for detecting horizontal edges and the other for detecting vertical edges. The horizontal and vertical gradients are computed by convolving these kernels with the image.

After computing the gradients, you will calculate the magnitude of the gradient at each pixel by combining the horizontal and vertical components. A threshold is then applied to decide which pixels are considered as edges based on the magnitude of the gradient. Only pixels with a gradient magnitude above a certain threshold will be classified as edges.

# 3 Phase 3: Using a Gaussian Filter to Reduce Noise

Before applying the Sobel edge detection, it's important to reduce any noise in the image. Gaussian filtering helps achieve this by smoothing the image. A Gaussian filter works by averaging pixel values within a window, with higher weights given to pixels near the center of the window. This reduces high-frequency noise, making edge detection more accurate.

Using a library like scipy.ndimage, you can apply a Gaussian filter with a specified sigma value. The sigma parameter controls the degree of smoothing: a larger sigma value leads to more blurring, which can be useful for noisy images. By smoothing the image before edge detection, you ensure that the edges detected are not influenced by random variations or noise in the image.

# 4 Phase 4: Finding the Polygon from Detected Edges

Once the edges are detected, the next step is to identify the main object in the image. This can be done by finding the contours in the edge map. A contour is a continuous curve that traces the boundaries of objects or areas with significant intensity changes.

Using the scikit-image library's `find_contours` function, you can automatically detect contours in the binary edge map. The `find_contours` function identifies all the curves that correspond to boundaries. To focus on the main object, you can select the largest contour, assuming it represents the most significant object in the image.

After extracting the largest contour, you can represent it as a polygon and draw it on a blank canvas. The points of the polygon correspond to the boundary of the detected object. You can then save the resulting image with the polygon drawn over it.

# 5 Bonus

Writing clear and well-structured documentation for your project is highly encouraged and comes with a bonus. Good documentation not only helps others understand your work but also makes it easier to maintain, improve, and collaborate on in the future. A well-documented project demonstrates professionalism and ensures that your efforts are accessible and useful to others.

# Summary

In this project, you will process an image by detecting its edges using the Sobel operator, reducing noise using a Gaussian filter, and extracting the main object's boundary using contour detection. These steps will allow you to identify the edges and represent the object as a polygon. The combination of edge detection and contour extraction is widely used in computer vision tasks for object detection, shape recognition, and image segmentation.