

# Screw Detection System: Technical Report

## 1. Introduction

This document outlines the architecture, working methodology, and analytical capabilities of a screw detection system developed using **Kivy** for the user interface and **OpenCV** for computer vision processing. The primary objective of the system is to identify and classify various types of screws from images or video inputs by analysing their geometric and visual features. The system employs a combination of classical image processing techniques—such as grayscale transformation, edge detection, and contour analysis—to extract key shape descriptors like area, aspect ratio, and circularity. These features are then evaluated through a structured rule-based algorithm to determine the most likely screw category. Each classification is mapped to a comprehensive metadata profile stored in a structured **JSON file**, which includes the screw's type, size, material, head and drive type, threading style, coating, and application context. The solution is designed to be lightweight, interpretable, and extendable, making it well-suited for applications in industrial automation, inventory management, smart kiosks, educational tools, and quality control systems.

## 2. System Overview

The screw detection system is designed as a modular and user-friendly application that combines a graphical interface with computer vision algorithms. It operates through three core components:

- **Graphical User Interface (GUI):** Built using the Kivy framework, the interface allows users to upload images or videos for analysis. It provides real-time feedback, displays visual previews, and outputs detailed screw classification results in a structured and accessible format.
- **Image and Video Processing Pipeline:** This component handles the technical analysis of the uploaded media. It preprocesses the input using grayscale conversion, blurring, and edge detection to generate an edge matrix. From this matrix, it extracts contours and calculates key shape features such as area, perimeter, circularity, and aspect ratio.
- **Rule-Based Classification Engine:** Based on the extracted features, this module applies a structured set of logical rules to determine the most probable screw type. The classification is matched against a JSON-based database containing real-world screw specifications. Each match returns a complete profile of the detected screw, including size, drive type, material, and usage.

This modular architecture ensures that the system is both flexible and extensible — allowing for future integration of machine learning models, larger screw databases, or more complex hardware recognition tasks.

## 3. Detection Algorithm and Matrix-Based Image Analysis

The detection algorithm follows a classical computer vision approach using OpenCV and NumPy to perform geometric and statistical analysis. Here's a detailed explanation:

### 3.1 Image Matrix Conversion and Preprocessing

- An image is uploaded and converted into a **NumPy array**, forming a 2D matrix of pixel values.
- This matrix is converted to **grayscale**, simplifying the data by reducing color channels.
- A **Gaussian blur** is applied to the matrix to suppress noise and prevent false edge detection.
- **Canny edge detection** is performed, generating a binary edge map matrix, where:
  - 1 represents edges (white)
  - 0 represents background (black)

This edge matrix is the basis for all subsequent detection.

### 3.2 Feature Extraction from Matrix

After the image is processed through Canny edge detection, the result is an **edge matrix**, where each pixel represents either an edge (foreground) or non-edge (background). From this binary matrix, key features are extracted to assess the likelihood that the object is a screw.

- **Edge Density:** Edge density refers to the proportion of pixels in the image that are part of detected edges. It is calculated by dividing the number of edge pixels (those with value greater than zero) by the total number of pixels in the image. This metric helps determine whether the image contains a screw-like object. Screws typically produce a **moderate edge density** — not as low as flat surfaces, but not as high as noisy or overly detailed textures. This serves as an early filter to reject invalid images.
- **Contour Detection:** Using the edge matrix, the algorithm identifies continuous curves around shapes — known as contours. These are extracted using standard OpenCV functions. Among all detected contours, the one with the **largest area** is selected, as it most likely corresponds to the screw in the image. This dominant contour is used in later stages to compute geometric properties such as shape, size, and orientation.

Together, edge density and contour detection provide the foundation for further analysis in the screw classification process. They enable the system to isolate the object of interest and quantify its visual structure in preparation for rule-based decision making.

### 3.3 Shape Metrics from Contours

**From the largest contour:** Once the largest contour (suspected to be a screw) is isolated from the edge matrix, several **geometric shape metrics** are calculated to describe its form. These help the system decide what type of screw (or object) it is:

- **Area:** The total number of pixels inside the contour, representing how much space the object occupies.  
→ Calculated using: `cv2.contourArea(contour)`
- **Perimeter:** The length of the outer edge of the contour — essentially, how long the border of the shape is.  
→ Calculated using: `cv2.arcLength(contour, closed=True)`

**Circularity:** A measure of how close the shape is to a perfect circle.

- Formula:

$$Circularity = \frac{4\pi * Area}{(Perimeter)^2}$$

Value  $\approx$  1: shape is circular

Value  $\ll$  1: shape is elongated or irregular

This is useful to distinguish between **round-head screws**, **bolts**, or **threaded rods**.

- **Aspect Ratio:** Describes how stretched or narrow the screw appears by comparing the bounding box's width and height.

- Formula:

$$\text{Aspect Ratio} = \frac{\text{Width}}{\text{Height}}$$

Values:

- Close to 1: the object is square or round
- Much < 1 or > 1: the object is elongated (like most screws)

These measurements are fed into a rule-based logic engine that determines the screw type. They help separate screws from bolts, nuts, and irrelevant shapes like tools or background clutter.

### 3.4 Thread Structure and Alignment

- **Thread Count:** Sub-contours are counted inside the bounding box, indicating presence of threading
- **Edge Variation:** Vertical edge pixel sums are computed, and their standard deviation is used to detect chaotic structures

### 3.5 Rule-Based Decision Matrix

Once an object is identified as a screw-like shape through contour detection and preprocessing, the system needs to decide **what type of screw** it is.

To do this, the algorithm relies on three main geometric features:

1. **Aspect Ratio (AR):**  
Measures how tall or wide the object is.
  - A value near 1 means the object is nearly square or round.
  - A lower value (< 0.5) means it's long and narrow (like a lag screw).
2. **Area:**  
Represents the size (in pixels) of the detected shape.
  - Larger areas usually indicate bigger screws or bolts.
  - Smaller areas might suggest machine screws or trim screws.
3. **Circularity:**  
Indicates how circular the object is (1 = perfect circle).
  - High circularity: rounded head screws (like shiny decorative ones).
  - Low circularity: longer, irregular shapes.

The algorithm uses these values in **conditional checks** to match the screw to a known type. This is done by defining a set of **if-elif rules** (a “decision matrix”) that compare the object's geometry to the patterns of screws stored in the JSON file.

## 4. JSON Metadata Matching

Each screw is assigned a category ID, which is then used to extract screw information from a structured JSON file. The fields include:

- Name, Type, Material, Size
- Head Type, Drive Type, Thread Type
- Strength Grade, Coating

- Application and Description

**Note:** The current JSON supports a subset of commonly used screw sizes and drive types. However, in real-world applications, screw variants are far more extensive.

### **Additional Common Sizes (Not in JSON):**

- M1, M1.6, M2, M2.5, M2.6, M3.5, M7, M9, M12, M14, M16, M20, M24, etc.

### **Additional Drive Types (Not in JSON):**

- Slotted Drive
- Square Drive (Robertson)
- Tri-Wing
- Pozidriv
- Spanner (Snake Eye)
- Torq-Set
- Clutch Head
- Combination Drives (e.g., Phillips-Slotted)

## **5. Example Detection Output**

From an image of a single screw, the system produced the following analysis:

#### **Detected Screw Information:**















- Name: Shiny Screw
- Type: Screw
- Material: Chrome Plated
- Size: M5
- Head Type: Round Head
- Drive Type: Phillips Drive
- Thread Type: Sharp Point
- Strength Grade: Grade 4.8
- Coating: Chrome Plated
- Application: Decorative projects requiring shiny finishes.
- Description: This chrome-plated screw with a round head is designed for decorative applications requiring an appealing, shiny finish. Ideal for interior fixtures and trim work, this screw offers moderate strength and corrosion resistance due to its chrome plating. Its round head provides an attractive appearance, making it a preferred choice in automotive interiors, home decor, and other projects where visual appeal is essential.









#### **Detection Metrics:**

- Aspect Ratio: 1.00
- Circularity: 0.90
- Area: 32938.00
- Bounding Box: Width=205, Height=205

## **6. Screw Types in the System (with Image Placeholder)**

The JSON file currently contains 14 fastener types. Below is a classification list showing which ones are supported by the program logic, with image placeholders for documentation purposes:

ID	Name	Type	Size	Drive Type	Included?	Image Placeholder	Head Of Screw
1	Long Lag Screw	Screw	M10	Wrench Drive	✓ Yes		
2	Lag Wood Screw	Screw	M8	Wrench Drive	✓ Yes		
3	Wood Screw	Screw	M6	Phillips Drive	✓ Yes		
4	Short Wood Screw	Screw	M4	Phillips Drive	✓ Yes		
5	Shiny Screw	Screw	M5	Phillips Drive	✓ Yes		
6	Black Oxide Screw	Screw	M5	Torx Drive	✓ Yes		
9	Torx Screw	Screw	M4	Torx Drive	✓ Yes		

ID	Name	Type	Size	Drive Type	Included?	Image Placeholder	Head Of Screw
10	Phillips Screw	Screw	M3	Phillips Drive	✓ Yes		
11	Hex Socket Cap Screw	Screw	M6	Hex Allen Key	✓ Yes		
12	Flat Head Machine Screw	Screw	M3	Phillips Drive	✓ Yes		
13	Pan Head Sheet Metal Screw	Screw	M4	Phillips Drive	✓ Yes		
14	Self-Tapping Screw	Screw	M5	Phillips Drive	✓ Yes		

## 7. Conclusion

The screw detection system demonstrates a practical and efficient approach to identifying and classifying screws using classical computer vision techniques. By leveraging edge-based feature extraction, contour analysis, and a rule-based classification matrix, the system can recognize screw types with good accuracy and associate them with detailed metadata from a structured JSON database. Its intuitive Kivy-based interface, support for both image and video input, and integration with real-world screw specifications make it highly adaptable for industrial, retail, and educational environments. This rule-based detection framework is lightweight, interpretable, and does not require complex training data — making it ideal for scenarios where fast, offline, or embedded inference is needed.

Looking ahead, the system can be enhanced by:

- Expanding the screw database with additional types, sizes, and drive patterns
- Introducing machine learning classifiers for improved generalization
- Supporting additional fasteners such as nuts, bolts, and washers
- Integrating a graphical report generation module with visual highlights of detection

Overall, this project offers a solid foundation for automated hardware recognition and can be a valuable tool for inventory systems, automated retail kiosks, or smart manufacturing stations.