Computer Vision

Module 600100 (Computer vision) ACW1

Word Count: 1402

Halima Abidemi Atanda 3/17/25 **202228071**

Contents

Image Processing Pipeline for Road Sign Detection Report

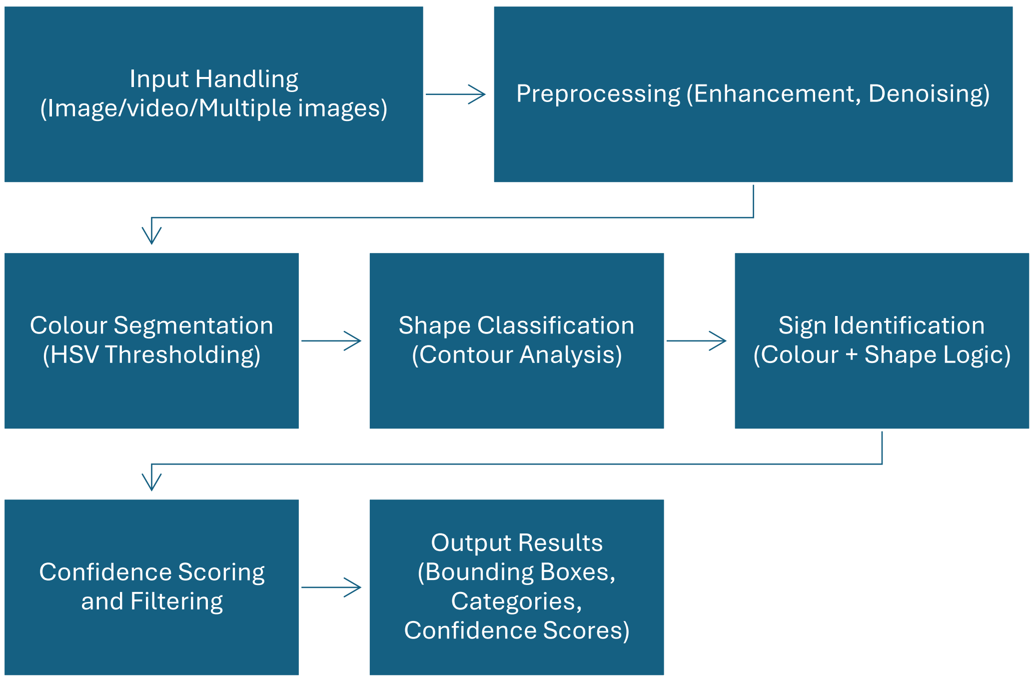
Overview

This report outlines the design, implementation, and performance of image processing pipeline which is designed to detect and classify road signs in both static images and video streams. It employs a combination of traditional computer vision techniques and heuristic approaches and avoids the use of deep learning methods to identify various types of road signs based on their colour, shape, and content. The project is structured to handle multiple input formats like single images, multiple images, and video frames in script format and provide detailed output for each detected sign in a structured format.

Pipeline Components

Flow Diagram

Here is a flow diagram of the pipeline functionality. This flow diagram illustrates the sequential progression of operations in the pipeline, with decision points at each stage influencing subsequent steps.



Design Choices and Rationale

1. Input Handling

Operations:

- The code supports single images which directly processes them using the process_image function, multiple images (via a text file) which processes via process_multiple_images function, and video inputs processes using process_video function
- It validates input paths and ensures compatibility with absolute/relative paths.
- It defaults output file for results is **output.txt** if none is specified in a structured format.
- The code uses **argparse** to handle command-line arguments. This is a robust and user-friendly way to specify input options.
- The code includes basic error handling to manage invalid inputs.

Rationale:

- The type of input (image, video, or list of images) determines the processing method.
- The modular design allows each input type to be handled independently while reusing common functions (e.g., detect_signs).
- Each argument corresponds to a specific input type, ensuring that the program can handle diverse scenarios without unnecessary complexity.
- Informing the user about missing or incorrect arguments prevents runtime errors and improves usability.
- The message explicitly states the valid input options, guiding users to correct their input.

Effectiveness:

The input handling choices in the code overall reflect a balance between flexibility, efficiency, and usability. By supporting multiple input types, providing robust error handling, and writing results in a standardized format, the program is well-suited for a wide range of applications involving road sign detection.

2. Preprocessing

Operations:

- CLAHE (Contrast Limited Adaptive Histogram Equalization) enhances local contrast to improve visibility of features.
- Denoising removes noise using **cv2.fastNlMeansDenoisingColored** while preserving the images details
- If an image is detected as blurry using the Laplacian variance method (detect_blur), it undergoes deblurring.
- Deblurring applies Wiener filter in the frequency domain to deblur images/videos.

Rationale:
Preprocessing ensures that the input image/ video is optimized for subsequent steps like colour segmentation and contour detection. CLAHE improves contrast in unevenly lit regions, while denoising reduces artifacts caused by sensor noise or compression.

Effectiveness:
These operations significantly improve the quality of binary masks generated during colour segmentation, leading to better contour detection.

3. Colour Segment Operations

Operations:
- Convert the image to HSV colour space.
- Apply thresholding to isolate specific colours (e.g., red, blue, yellow).

Rationale:
Road signs are designed with distinct colours to convey specific meanings (e.g., red for prohibition, blue for mandatory). HSV is more robust than RGB for colour segmentation because it separates intensity from hue and saturation.

Effectiveness:
- Colour segmentation reliably isolates sign regions of interest based on predefined HSV colour ranges for red, blue, yellow, white reducing false positives from irrelevant objects.
- It means overlapping colour ranges (e.g., red shades) require careful tuning.

4. Shape Classification

Operations:
- Detect and extracts contours from masks using **cv2.findContours**.
- Approximate contours with **cv2.approxPolyDP** to classify shapes (e.g., circle, triangle, octagon).
- For rectangles, aspect ratio is checked to distinguish squares from elongated rectangles.
- Create masks for each colour range using cv2.inRange.
- Convert image to HSV colour space for robust colour detection.

Rationale:
- Road signs have standardized shapes that correspond to specific categories (e.g., triangles for warnings, circles for prohibitions). Shape classification helps narrow down potential sign types.
- Contours are classified based on their geometric properties.

Effectiveness:
- The contour approximation method works well for most signs but struggles with heavily occluded or distorted shapes. Adding aspect ratio checks improves accuracy for rectangular and square signs.
- Shape classification provides an additional layer of filtering and helps in identifying specific sign types.
- This step effectively narrows down potential sign locations, reducing computational complexity

5. Sign Identification

Operations:
- Combine colour and shape information to identify predefined sign categories.
- Special handling for red circular signs:
  - Check for white regions within the contour to differentiate speed limits from prohibition signs.
  - Use adaptive thresholding to detect text or patterns inside the white region.
- For speed limit signs:
  - Check for a significant white region within red circles.

Rationale:

Signs are classified based on their visual characteristics. Speed limit signs are identified by their unique structure: a red circle with a white region containing black text.

Effectiveness:

- This approach effectively distinguishes between similar signs (e.g., prohibition vs. speed limit).
- This heuristic-based approach works well for common road signs but may struggle with rare or complex signs.

6. Confidence Scoring

Operations:

Confidence scores are calculated based on multiple factors:

- Shape reliability (e.g., circles have higher confidence than complex shapes).
- Colour reliability (e.g., red and blue are prioritized).
- Image quality (non-blurry images increase confidence).
- Contour area (larger areas indicate higher confidence).

Rationale:

Confidence scoring ensures that only high-quality detections are included in the output. This step filters out weak or ambiguous detections.

Effectiveness:

- The scoring mechanism improves overall accuracy by penalizing low-confidence detections.
- Confidence scoring helps filter out false positives and rank detections.

7. Output Generation

Operations:

- Write detection results to a file, including bounding box coordinates, category IDs, and confidence scores.
- Group detections by filename/frame number.
- Select the best detection per image/frame based on confidence scores.
- Validate bounding box coordinates to ensure they are within valid ranges.

Rationale:

Structured output enables easy evaluation and integration with downstream systems about detected signs.

Effectiveness:

The output format meets the requirements and facilitates analysis of detection performance.

Code Logic Explanation

The pipeline logic is modular and follows a sequential flow:

- Input handling determines how data is loaded into the pipeline (image/video/multiple images).
- Image enhancement improves input quality before processing begins.
- Sign detection uses colour segmentation and contour analysis to locate potential signs.
- Shape classification filters contours further by identifying geometric shapes.
- Sign identification combines shape and colour information to classify signs into categories like warning, prohibition, imperative.
- Confidence calculation assigns scores to detections based on reliability metrics.
- Output generation filters detections and writes structured results to an output file.

Each function is designed with clear responsibilities and minimal dependencies, ensuring modularity and ease of debugging.

Effectiveness Discussion

The pipeline performs well under controlled conditions where road signs are clearly visible with distinct colours and shapes. The use of HSV colour space makes it robust against lighting variations, while shape classification adds precision in identifying specific sign types.

However, there are limitations:

- Occlusion Handling: Partially occluded signs may not be detected accurately due to reliance on complete contours.
- Rare Signs: Uncommon signs or those with unusual colours/shapes may be misclassified as "unknown."
- Lighting Challenges: Extreme lighting conditions can affect colour segmentation accuracy despite HSV usage.

Potential Improvements

- Machine Learning Integration:
  Implementing a CNN for sign classification could significantly improve accuracy and robustness. This approach has shown success in recent research, with some methods achieving detection rates above 99% in continuous image sequences.
- Advanced Tracking for Videos:
  Incorporating object tracking algorithms like Kalman filters or optical flow methods could improve consistency in video processing and reduce computational load.
- Handling Occlusions:
  Developing methods to detect and classify partially visible signs would enhance real-world performance.
- Adaptive Thresholding:
  Implementing dynamic confidence thresholds based on image quality and scene complexity could improve detection reliability.
- Speed Optimization:
  Utilizing GPU acceleration or parallel processing could enhance real-time performance, especially for video analysis.
- Expanded Sign Categories:
  Adding support for more sign types and improving classification of challenging signs (e.g., speed limit signs) would increase the pipeline's utility.
- Data Augmentation:
  Training on a diverse dataset with various lighting conditions, weather effects, and sign variations could improve robustness.
- Multi-Scale Processing:
  Implementing multi-scale image inputs during training could improve detection of small objects or distant signs.

Conclusion

This image processing pipeline provides a comprehensive approach to road sign detection and classification. Its modular design allows for easy integration of future improvements and extensions. While the current implementation relies heavily on traditional computer vision techniques, there is significant potential for enhancing its capabilities through the integration of modern machine learning approaches which would significantly enhance its performances and reliability across various real-world scenarios.