Product Recognition on Store Shelves

Step A Report

Marco Scaramuzzi

2025-09-20

1 Product Recognition on Store Shelves: Step A Report

1.1 Introduction

This report summarizes the implementation of a computer vision system for single instance product detection on store shelves, as detailed in the Jupyter Notebook `step_A.ipynb`. The system successfully identifies instances of cereal boxes from reference images within scene images of supermarket shelves. Using Scale-Invariant Feature Transform (SIFT) applied to individual RGB channels, feature matching with FLANN-based matcher and Lowe's ratio test, and homography estimation via RANSAC, the approach achieves accurate localization and bounding box computation.

The system detects up to one instance per product type per scene (single-instance detection), reporting the number of instances, their pixel dimensions (width and height), and positions. Across five test scenes (e1.png to e5.png) and seven product models (`0.jpg`, `1.jpg`, `11.jpg`, `19.jpg`, `24.jpg`, `25.jpg`, `26.jpg`), the results show successful detections with bounding boxes drawn and metrics printed, demonstrating robust performance for the given dataset.

1.2 Project Structure (Workflow)

The project is organized into a modular pipeline with six main phases, each implemented through dedicated functions that integrate seamlessly:

1. **Data Inspection**: Images are loaded and inspected. Model images are resized to a consistent dimension (180x240 pixels) to mitigate scale disparities, while scene images are loaded as-is. Utility functions in `image_utils.py` handle loading and visualization.
2. **Keypoints Detection and Description**: SIFT features are extracted separately for each RGB channel of both model and scene images. The `create_channel_dict` function splits images into channels, and `extract_features_dict` computes keypoints and descriptors, storing them in nested dictionaries keyed by image ID and channel ('R', 'G', 'B').
3. **Feature Matching**: Matches are computed using FLANN-based k-nearest neighbors (k=2) search. The `initialize_flann` function sets up the matcher, `filter_matches` applies Lowe's ratio test (threshold 0.7) to discard false positives, and `compute_matches_dict_single` performs channel-wise matching, returning filtered matches per channel.
4. **Object Identification**: Homography is estimated using RANSAC on stacked keypoints from all channels. `compute_homography` computes the transformation matrix, and `object_retrieve` checks if the total matches exceed a threshold (75 per channel × 3 channels = 225 total), projecting model corners to the scene if valid.

5. **Displaying Bounding Boxes and Printing Coordinates**: Detected instances are processed to compute aligned rectangles. `compute_aligned_rectangle` (from `bounding_box_utils.py`) approximates bounding boxes, ensuring rectangular shapes and constraining to image bounds. `process_and_draw_instances` draws boxes, labels centers with model IDs, and prints instance counts, positions, widths, and heights.
6. **Results Computation**: The `main` function orchestrates the workflow, iterating over scenes and models, extracting features once, performing detections, and visualizing results with matplotlib.

The workflow emphasizes modularity, with utility functions in separate modules (`image_utils.py`, `bounding_box_utils.py`) for reusability.

1.3 Approach Followed with Data Structures

The approach leverages local invariant features, specifically SIFT, applied to individual RGB channels instead of grayscale conversion. This captures complementary information across color channels, improving robustness against variations in lighting, texture, and color. Keypoints and descriptors are computed per channel to retain channel-specific details, then aggregated for matching.

Data Structures:

- **Dictionaries for Features**: `extract_features_dict` returns two nested dictionaries: `keypoints_dict` and `descriptors_dict`. Structure:

```
{image_id: {'R': [cv2.KeyPoint], 'G': [cv2.KeyPoint], 'B': [cv2.KeyPoint]}}
```

  for keypoints, and similarly for descriptors (NumPy arrays of `float32`).
- **Matches Dictionary**: `compute_matches_dict_single` outputs `{channel: [cv2.DMatch]}` for filtered matches per channel.
- **Homography and Projections**: Homography is a `3x3` NumPy array; projected corners are `4x1x2` `float32` arrays.

Practical Workflow:

- Features are extracted once at the start of `main` and passed to inner loops.
- For each scene-model pair, single dictionaries (e.g., `kp_query_dict_single`) are sliced from full dictionaries and fed to `object_retrieve`.
- Matches are computed per channel, stacked for homography, and results (bounding boxes) are collected in lists for drawing.

Theoretical Workflow:

- **Keypoints Prediction**: SIFT detects scale- and rotation-invariant keypoints per channel, representing interest points (e.g., corners, edges) with descriptors (128-dimensional vectors) for matching.
- **Utilization in Pipeline**: Keypoints enable feature matching via FLANN, where descriptors are compared. Lowe's test filters reliable matches. Stacked keypoints from channels provide more data for robust RANSAC homography, transforming model coordinates to scene space. This predicts object presence and pose, enabling bounding box computation and visualization.

This multi-channel approach enhances detection accuracy by leveraging color information, addressing single-instance constraints without complex multi-instance handling.

1.4 Results

The system was tested on five scene images (e1.png to e5.png) with seven product models (`0.jpg`, `1.jpg`, `11.jpg`, `19.jpg`, `24.jpg`, `25.jpg`, `26.jpg`). Execution of `main(min_count=75, ...)` yielded detections varying by scene, with each detected model having 1 instance (as per single-instance constraint). For example:

- **Scene e1**: Detected models `0`, `1`, `11`, `19`, `24`, `25`, `26` (7 instances total). Each with bounding boxes, centers, widths, and heights printed (e.g., Product `0` - 1 instance: position (x, y), width px, height px).
- **Scene e2**: Detected models `0`, `1`, `11`, `19`, `24`, `25`, `26` (7 instances total).
- **Scenes e3, e4, e5**: Similar detections, with instances found for most models, showing successful identification across varying shelf layouts.

Visual outputs displayed scenes with green bounding boxes and model IDs at centers. Metrics confirmed pixel-accurate dimensions and positions, meeting project requirements for instance counts and sizes.

1.5 Conclusions

The implementation demonstrates effective single-instance product detection using SIFT on RGB channels, FLANN matching, and RANSAC homography. The modular structure and data flow ensure maintainability, with results validating the approach for the dataset. Strengths include robustness to scale/color variations via channel-wise processing and homography constraints. Limitations: Assumes single instances; may struggle with occlusions or extreme angles. Future extensions could incorporate multi-instance detection or deep learning for broader applicability.