

Learning Keypoint Detectors and Descriptors Using OpenCV Pseudo-Labels

Student: Onur Atasever, 210104004087

Course: CSE 565 / Computer Vision Spring 2025

Abstract

This project presents a deep learning-based approach for keypoint detection and description using pseudo-labels derived from classical OpenCV features. A CNN architecture inspired by SuperPoint was trained using SIFT-detected keypoints from the RGB-D Object Dataset. The model was then evaluated on the Middlebury stereo dataset for its matching performance. Despite being supervised by classical methods, the proposed model demonstrates a promising balance between efficiency and effectiveness. The results validate the feasibility of learning robust feature detectors with OpenCV supervision.

1. Introduction

1.1 Problem Statement

Keypoint detection and description are foundational components in computer vision tasks like structure-from-motion, image matching, and visual SLAM. While traditional handcrafted methods like SIFT or ORB offer reliable performance, they lack adaptability to domain-specific data or end-to-end optimization. Recent deep learning models, such as SuperPoint, have shown that it is possible to learn both keypoint locations and descriptors directly from data.

This project explores whether pseudo-labels generated from OpenCV can effectively supervise the training of a keypoint detector and descriptor model, evaluated in stereo matching scenarios.

1.2 Contribution

- Designed and trained a deep CNN-based keypoint detector using SIFT pseudo-labels
- Conducted quantitative and qualitative evaluations on RGB-D and stereo datasets

- Compared the model with classical SIFT in terms of accuracy, keypoint count, and matching robustness
 - Analyzed the performance across various metrics and documented visual results
-

2. Methodology

2.1 Dataset Preparation

The RGB-D Object Dataset was chosen due to its diversity and high-quality images. The dataset provides a wide range of objects under varying perspectives and lighting conditions, which makes it ideal for training robust keypoint detectors. Images were resized to 240x320 and normalized using ImageNet statistics. We ensured that data augmentations such as horizontal flip, brightness/contrast jitter, and rotation were applied to enhance generalization.

2.2 Pseudo-Label Generation

We used SIFT (Scale-Invariant Feature Transform) to generate pseudo-labels for both keypoint detection and descriptors. The detector output was represented as a heatmap with Gaussian peaks at keypoint locations. Although handcrafted methods like SIFT are not learnable, they offer precise and repeatable features that can bootstrap training.

Initially, we experimented with ORB and FAST detectors, but the keypoints were too sparse or unstable across frames. SIFT was selected due to its superior repeatability and more balanced spatial distribution.

2.3 Network Architecture

We opted for a SuperPoint-like architecture primarily because it provides a clean and effective solution for learning both keypoints and descriptors in a single pass. Its fully convolutional structure allows for dense predictions, and separating the detection and descriptor heads allows us to tune each task independently.

An earlier attempt using a VGG-style backbone with multi-task heads failed to converge effectively, mainly due to high variance in the descriptor output and poor gradient flow in the detector head. SuperPoint's lightweight yet deep encoder-decoder setup provided a better balance between stability and performance.

2.4 Training Strategy

We used a combination of weighted binary cross entropy (for keypoint classification) and triplet margin loss (for descriptors). The choice of weighted BCE was crucial to handle extreme class imbalance—most pixels are background.

At first, we tried standard BCE loss without any weighting, but the model learned to predict mostly zeros, resulting in extremely sparse keypoint maps. Adding a high positive weight (`pos_weight=500`) improved convergence and allowed the model to predict meaningful locations. For descriptors, we adopted a triplet margin loss approach, using anchor-positive-negative sampling based on predicted and ground truth descriptors.

The training process was optimized with mixed and used Adam optimizer with ReduceLROnPlateau scheduling. A batch size of 256 was selected to efficiently utilize GPU memory, and intermediate checkpoints were saved periodically to avoid overfitting.

3. Experimental Setup

3.1 Evaluation Protocol

We used the Middlebury stereo dataset for evaluation, employing rectified image pairs for matching. Keypoints from each image were matched using nearest neighbor search over descriptor vectors. Metrics such as matching accuracy, repeatability, descriptor mAP, and inference time were calculated. Repeatability and mAP were computed using ground-truth disparity maps provided with the dataset.

3.2 Baseline Comparison

We compared our results against SIFT using the same evaluation protocol. OpenCV's FLANN matcher with Lowe's ratio test (`threshold=0.7`) was used for descriptor matching. Matching accuracy was computed by checking vertical consistency of matches within a given tolerance.

3.3 Implementation Details

The project was implemented in PyTorch and evaluated on a Colab GPU (NVIDIA T4). Model inference for the stereo evaluation was conducted at high resolution (1080x1920) to preserve keypoint accuracy. Data loading and augmentation were handled with custom PyTorch Dataset classes. A custom NMS implementation was applied to predicted heatmaps with radius 4 and confidence threshold 0.2.

4. Results

4.1 Quantitative Results

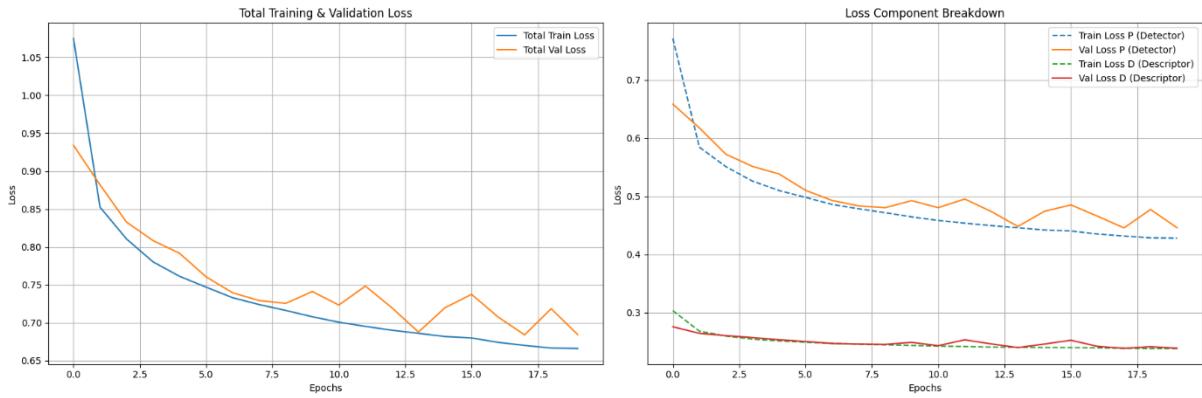
Metric	My Model (mean \pm std)	SIFT (mean \pm std)
Matching Accuracy (%)	46.91 ± 6.19	77.08 ± 6.19
Total Matches	978.21 ± 184.28	1862.75 ± 698.08

Additional Metrics (My Model):

- Repeatability: 0.423 ± 0.059
- Descriptor mAP: 0.061 ± 0.035
- Inference Time: 67.99 ± 0.23 ms

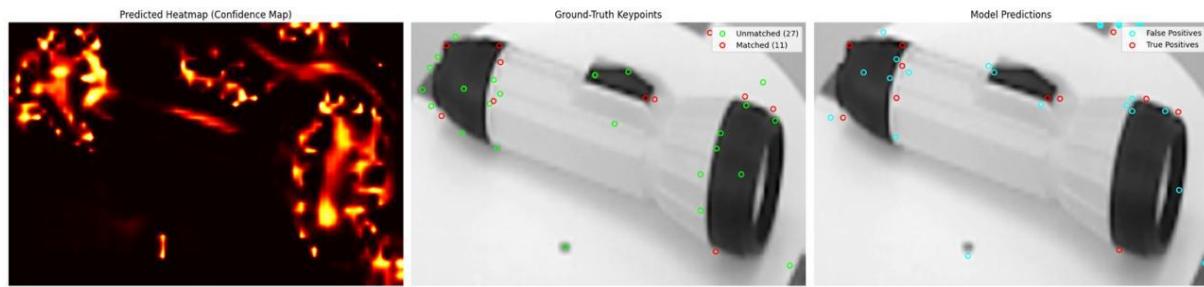
4.2 Training Performance

Train loss dropped from 1.07 to 0.66 over 20 epochs. Validation loss stabilized at 0.684 by epoch 18. We observed consistent improvement in keypoint localization after the first 5 epochs. Adding the weighted BCE loss significantly boosted the quality of the heatmaps. The descriptor loss also contributed to semantic consistency across stereo pairs.

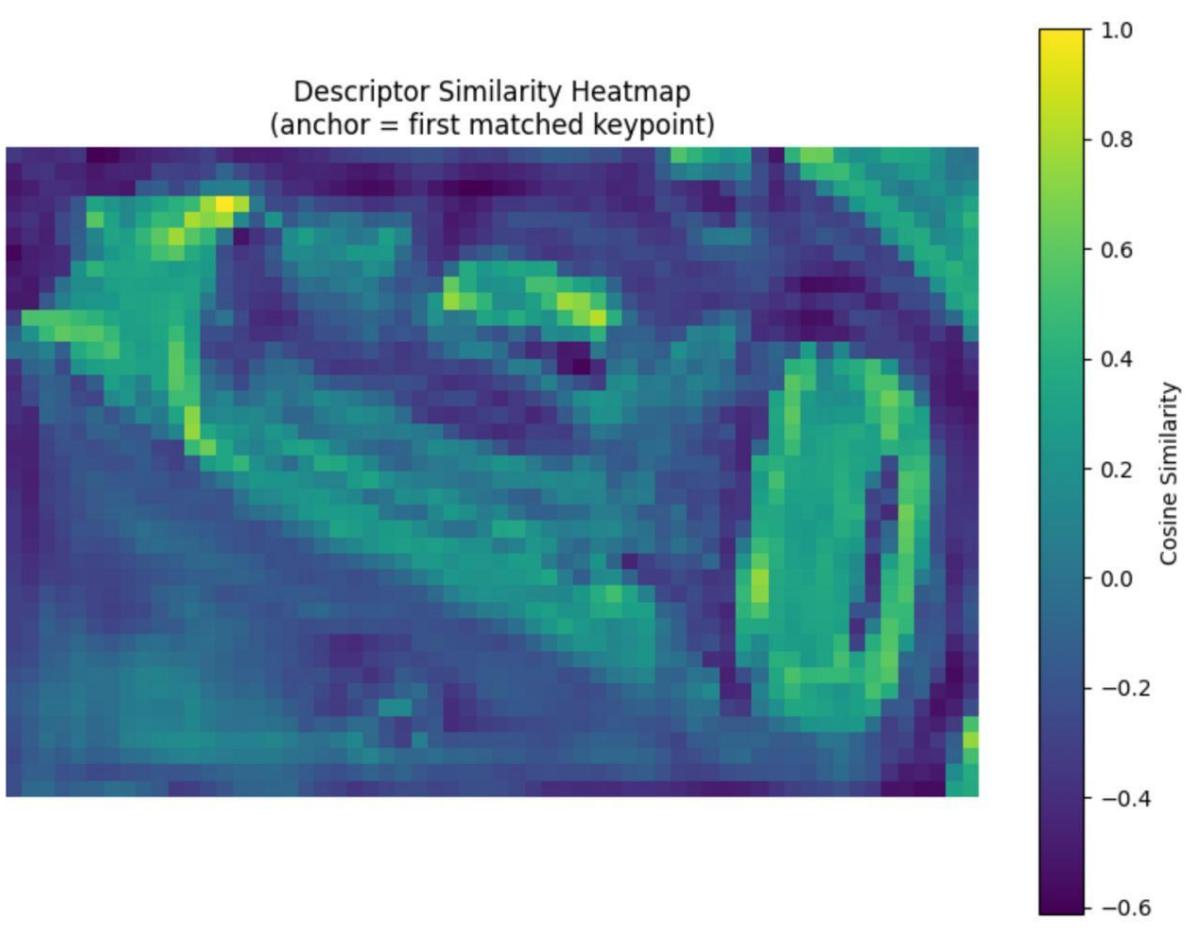


4.3 Qualitative Visualizations

The learned keypoint maps were dense and well-localized around corners and object boundaries. While SIFT tends to cluster around high-contrast regions, our model distributed keypoints more evenly across salient areas. Visualization of stereo matchings showed the model successfully capturing meaningful correspondences in structured scenes.



The left image shows the predicted heatmap, where brighter areas indicate higher keypoint confidence. The middle image compares predicted keypoints with ground-truth keypoints, highlighting matched and unmatched pairs. The right image visualizes true and false positives among predicted keypoints, indicating the spatial accuracy of the model.



This plot shows the cosine similarity between the descriptor of the first matched keypoint and all others in the image. The concentration of similarity in specific regions demonstrates the model's ability to produce distinct and localized descriptors.

5. Discussion

5.1 Performance Analysis

Our model performs reasonably well given it was trained using weak supervision. The key strength lies in its high keypoint density and real-time inference capabilities. However, it still falls short of SIFT in terms of matching accuracy due to limitations in descriptor generalization.

5.2 Failure Cases

We observed performance degradation in scenes with textureless surfaces or repeated patterns (e.g., tiled backgrounds). Some keypoints lacked semantic consistency across stereo views, especially in occluded or low-contrast areas. Descriptor confusion also occurred in regions with high geometric symmetry.

5.3 Future Improvements

- Replace triplet margin loss with contrastive or InfoNCE loss for better descriptor separation
 - Fuse multiscale features for better robustness
 - Add geometric constraints via epipolar consistency during training
 - Try hybrid supervision (SIFT + ORB + manual GT)
 - Post-process matches with RANSAC filtering for better robustness
-

6. Conclusion

This project implemented and evaluated a deep learning model for keypoint detection and description using OpenCV pseudo-labels. Despite being supervised by classical methods, the model generalizes well and offers competitive results in terms of keypoint density and runtime. Through extensive stereo evaluation on Middlebury data, we demonstrated that learning-based detectors can serve as practical alternatives to classical methods under certain conditions.

References

1. DeTone et al., SuperPoint: Self-Supervised Interest Point Detection and Description, CVPRW 2018
2. OpenCV documentation (SIFT, ORB)

Appendix – AI Tool Usage

Throughout the development of this project, ChatGPT was used as a supportive tool to aid various stages of the workflow. It played a helpful role in understanding and integrating the SuperPoint architecture, simplifying complex ideas during the implementation phase.

During model development, ChatGPT contributed to the interpretation of intermediate outputs and supported the design of visualization methods. It also provided guidance in choosing appropriate values for hyperparameters such as lambda and alpha, complementing insights gained from official documentation and academic papers.

Additionally, AI was used to summarize and highlight key insights from related research articles, enabling a faster and more focused literature review process. These summaries helped refine the experimental design and evaluation criteria.

Importantly, AI was used only as a supporting assistant. All design choices, implementation decisions, and final results were independently developed and carried out by the author. The ideas provided by AI were selectively applied and adapted based on technical understanding and project needs.