Light Skeleton Detection: Utilizing vehicle light positions for angle agnostic signal state detection

Jonas Benjamin Krug[1], Martin Ludwig Zehetner[1] & Yuan Xu[1]

[1]Distributed Artificial Intelligence Laboratory, Technical University of Berlin, Germany

November 1st, 2023

1 Introduction

Vehicle lights serve as indispensable indicators to human drivers of the near-term behavior of other road users. Whether this is in the form of brake lights, signaling a reduction in speed or a complete stop to following drivers, or turn signals, indicating an impending turn or lane change, providing the driver with information about the possible acceleration, deceleration, or lateral movement of other road users. Consequently, various reports and studies have shown that the lack of information about the vehicle lighting of other road users can significantly increase the risk of accidents, e.g., [1] and [2]. Incorporating these signals and decoding their underlying meaning thus offers a promising and directly applicable avenue to enhance the capabilities (e.g. trajectory prediction, path planning, etc.) and safety of autonomous driving systems.

Since the current Autoware project does not provide such blinker or vehicle light detection modules, and other openly available methods do not incorporate comprehensive, multi-perspective approaches, we propose a novel method for detecting the light states of surrounding vehicles (i.e., blinking, braking, or disabled) using estimated custom keypoint skeletons of designated vehicle lights and their associated states (i.e., lights on or off). We propose `Light Skeleton Detection` (LSD) leveraging real-world urban traffic scenes recorded in the context of the BeIntelli [3] research project.

The following sections outline the state of current vehicle light detection approaches and highlight shared shortcomings. Furthermore, we provide an overview of our proposed method and subsequently the structure of our data and annotation process will be described. Next our implementation will be detailed and the achieved results will be summarized. Finally, the next possible steps will be presented.

2 Background

2.1 Vehicle Light Datasets

There are two types of datasets that can be considered relevant to the problem at hand. These are datasets that address either the vehicle light detection task or the signal state detection task. Vehicle light detection describes the task of localizing vehicles and their associated individual visible vehicle lights, while signal state detection can be defined as the task of detecting vehicles and assigning them a behavioral vehicle light state (i.e., blinking, braking, disabled, etc.) based on the illumination of these individual vehicle lights.

The two types of datasets and their corresponding tasks can be considered correlated, since estimates from vehicle light detections, i.e., the positions of the vehicle lights, can be easily exploited when seeking to determine the signal state of observed vehicles. Unfortunately, the number of openly available datasets is small for both the vehicle light, e.g., [4] and [5], and signal state detection tasks, e.g., [6] and [7], and existing datasets suffer from some common drawbacks. Most importantly, the existing datasets focus mainly on solely the rear or the front of observed vehicles. Accordingly, often only a single front mounted camera is used to capture traffic scenes. As a result, annotated side views or differently angled images are often severely underrepresented or entirely non-existent.

2.2 Signal State Detection

The aforementioned focus on the primary detection of vehicle rear lights among signal state detection approaches can be seen in [8], [9], [10], and [11]. These solutions typically utilize a two-step approach. First, the individual vehicles and rear lights are detected, i.e., vehicle light detection, and secondly, the localized tail light candidates are leveraged to predict a signal state estimate for the detected vehicle.

In this context, [8] and [9] represent purely feature-based methods. Which initially extract candidate regions based on color clusters from the road scenes and subsequent filter and post-process the vehicle light candidates using, respectively, symmetry checks and Kalman filtering [8], or pre-computed cluster thresholds and convex-hull computations [9]. Signal state prediction are then

generated, using calculated correlations of the brightness values of the extracted regions-of-interest (ROI).

In contrast, [10] and [11] use state-of-the-art neural networks in their approaches. While [10] extracts bounding boxes of vehicles using a faster RCNN-based workflow, [11] generates region proposals of vehicle rear ends using pre-trained YOLOv4 model components. [10] then performs image threshold segmentation to assign image pixels to target objects, i.e., vehicle rear lights, and edge detection and morphological closure operations are used to extract the roi proposals. ROI proposals are paired afterwards, by correlating gradient histogram parameters, distance and color. The signal state prediction is then based on the analysis of the histogram characteristic parameters of the segmented ROI proposals. Conversely, [11] uses region-based adaptive thresholds for tail light segmentation and an MLP network for signal state classification based on an encoding of the computed tail light segments.

Currently, Autoware lacks any implementation for detecting either the vehicle lights or the signal state of observed vehicles. Furthermore, as illustrated, proposed solutions can also be seen as inherently limited, partly due to the equally limited nature of the available datasets, as they generally lack a holistic consideration of the entire surrounding environment, e.g., vehicles viewed perpendicular to the ego vehicle.

3 Proposed Solution

We propose a novel approach, called `Light Skeleton Detection` (LSD), for the detection of vehicle lights and corresponding vehicle signal states, that is not constrained to specific vehicle angles or camera positions. Our approach represents an end-to-end model, leveraging knowledge gained from internal vehicle light detection components to improve the overall performance of the signal state detection. To this end, instead of just detecting the lights, our proposed approach is skeleton-based, with each vehicle light represented by a node in the skeleton. The skeleton, as shown in Figure 1, is made up of seven nodes for the two frontal lights, two mirrors, two rear side lights and one center rear light. A visualization of these skeletons on real world data can be seen in Figure 2.

Each node has a location and a state. The state is a two dimensional vector with values between zero and one. The first dimension represents the confidence of the light having an active break indicator and the second dimension represents the confidence of the light having an active blinking indicator. This way the model gets a strong learning signal for the light state, even if the vehicle is facing the camera at an angle. The model is trained on our custom BeIntelli-LSD-Dataset and is integrated into the Autoware stack for prediction and visualization.

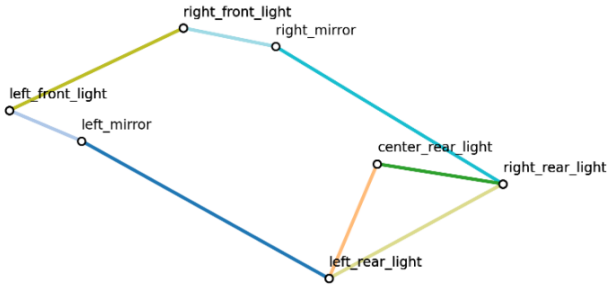

Figure 1: Light State Skeleton

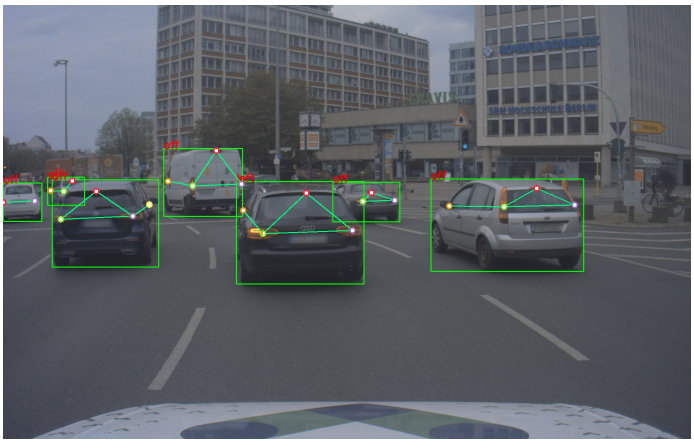

Figure 2: Visualization of the light state skeleton on sample data.

4 Data capture

A VW Tiguan from our BeIntelli project fleet, equipped with seven cameras mounted on a roof rack, was used to record our training and test data in a central urban area of Berlin. The camera stack consists of two narrow field of view cameras (60°) mounted at the front and rear of the vehicle, and five wide field of view cameras (120°) mounted at the front and each corner of the vehicle.

By recording from all seven cameras we are able to holistically record the surroundings of the test vehicle at all times, and capture a diverse range of views and perspectives from all vehicle angles. The images are synchronously recorded with a resolution of 1920×1208 at about 10 Hz and rectified to counteract any camera specific image distortions. So far only day time data from a confined urban area was evaluated. One example recording from our of all cameras can be seen in Figure 3.

Figure 3: Sample images from all seven camera views

Our labeling workflow when generating the BeIntelli-LSD-Dataset can be described as follows. First, we feed the pre-processed images into a pre-trained YOLOv8 model [12] to generate ROI's for the vehicles contained. The so detected vehicle ROI's were afterward passed to a modified version of the OpenPif-Paf [13] keypoint detection model to generate vehicle light keypoint proposals (assigned to the given vehicle). These automatically generated pre-annotations were checked and, if necessary, corrected and supplemented. For each identified vehicle ROI, we then assigned one of seven labels representing the observed signal state:

- Brake
- Hazard
- Brake & Hazard
- Brake & Left Blinker
- Left Blinker
- Brake & Right Blinker
- Right Blinker
- Lights Disabled

Individual labels for the light keypoints are not necessary, as the light states, i.e. on or off, of the vehicle lights can be derived from the assigned signal states.

5 Model

The model utilized for detecting these Light State Skeletons is a modified version of PyTorch's Keypoint-RCNN [14]. The number of keypoints was reduced to

seven and the number of bounding box classes was increased to nine, to account for every possible combination of light states with one additional background class. It was endowed with a custom head that predicts the two additional states per keypoint. The modified network architecture is depicted in Figure 4.

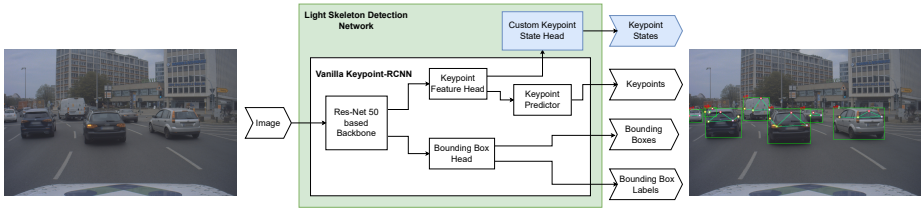

Figure 4: LSD-Network architecture showing the vanilla Keypoint-RCNN from PyTorch and its extension with the keypoint state head

The custom keypoint state head receives the internal feature map from the keypoint head. This is passed through two convolutional and two linear layers to generate the keypoint state prediction.

6 Results

So far the results of our model seem promising. Visual inspection shows that the model can accurately detect keypoints and their states. It also shows the ability to generalize to unseen vehicles.

We evaluated the model with a 16:1 split on our BeIntelli-LSD-Dataset. As seen in Table 5 the model achieves an average Intersection over Union (IoU) of 0.891 and an average bounding-box label accuracy of 0.953. Since our bounding box labels contain a combination of light states (e.g. brake and turn left), we also evaluate the model on the accuracy for each light category independently. For this we split each label in to three states, one for each light. Then we calculate the average accuracy over all labels. Here the accuracy is even higher with an average value of 0.984.

Metric	Average
IoU	0.891
Bounding-Box Label Accuracy	0.953
Split Light State Accuracy	0.984

Figure 5: Evaluation of the model on the BeIntelli-LSD-Dataset

One of the objectives for this challenge was the ability to execute our solution in real time on embedded hardware. We performed our evaluation for this on

a NVIDIA GeForce RTX 3070 Laptop GPU and visualized the results in Table 6. On this hardware our inference ROS2 node runs at an average of 11Hz. The latency of only the PyTorch model is on average 81.5ms. This could further be improved by optimizing the model with methods such as quantization or by converting it to TensorRT.

Metric	Average
ROS2 Node Frequency (RTX3070m)	11Hz
Model Latency (RTX3070m)	81.5ms

Figure 6: Performance analysis of the ROS2 inference node and PyTorch model on mobile hardware

Example pairs of images that show the input image together with a visualization of the model output are shown in Figure 7. These images are all recorded from different cameras on the vehicle.

In the first pair of images we can see that the model accurately detects keypoint positions on vehicles viewed from different angles. In the second pair of images we can see another such example on parked cars. In both pairs all light states are off, which is detected properly. The third pair of images shows the benefit of predicting the states per keypoint. While the bounding box head fails to detect the correct labels for two cars (both are predicted as off), the keypoint state prediction produces the correct output. Looking closely we can see that the state inside of the keypoints is accurately predicted as braking and blinking left, for the left most vehicle, and as braking for the right most vehicle.

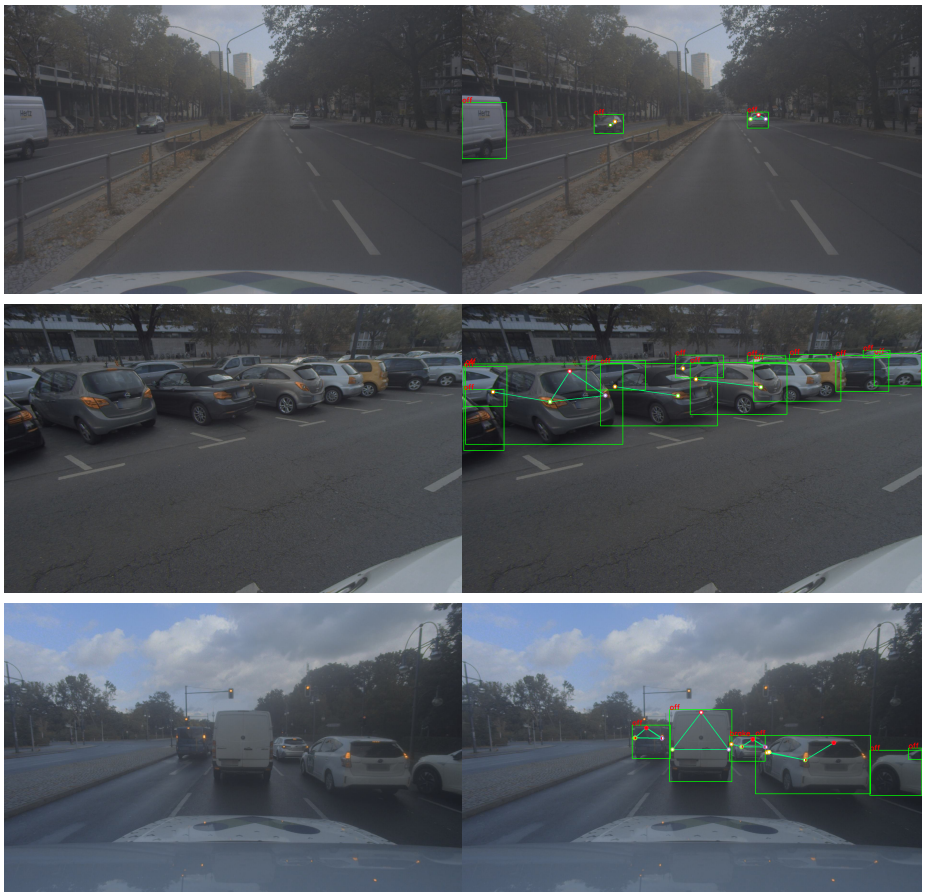

Figure 7: Sample pairs of images visualizing a plain input image and a version with the model output plotted on it. Keypoints are visualized as colorful rings with the skeleton connecting them in the form of green lines. The individual keypoint states are visualized inside of the ring with white indicating off, red indicating breaking and orange indicating blinking. The orange markers are drawn either on the left or right to show the direction of the blinking indicator.

7 Integration in to Autoware

Integrating this model in to Autoware was straight forward. We implemented a ROS2 node that runs the inference on our model and publishes the results as a `DetectedObjectsWithFeature` message. It publishes the bounding-boxes of the detected cars together with a list of classifications. All objects have a classification for the car class and additional classifications in case the left turn indicator, the right turn indicator or the break lights are on. These classifications are represented as integer values with 100 representing an activated break light, 101 representing an activated left turn indicator and 102 representing an activated right turn indicator. Each light classification is paired with a floating point number between zero and one that indicates the confidence in this classification.

Autoware already possesses the ability to fuse these classifications with 3D objects. However, it only fuses the most likely classification. This makes sense in cases where the classifications are mutually exclusive, like car and bicycle. Since we want an object to have multiple classifications we restrict this filtering on classifications with IDs under 100, which currently restricts it to only filter out irrelevant vehicle types.

This classification can then be used to aid the prediction module in creating a more accurate forecast of a vehicles movement. A proof of concept for the improvement in prediction accuracy was not yet finalized at the time of this submission. The classification can be visualized in RViz by adding additional markers to the 3D objects. Two examples of this can be seen in Figure 8.

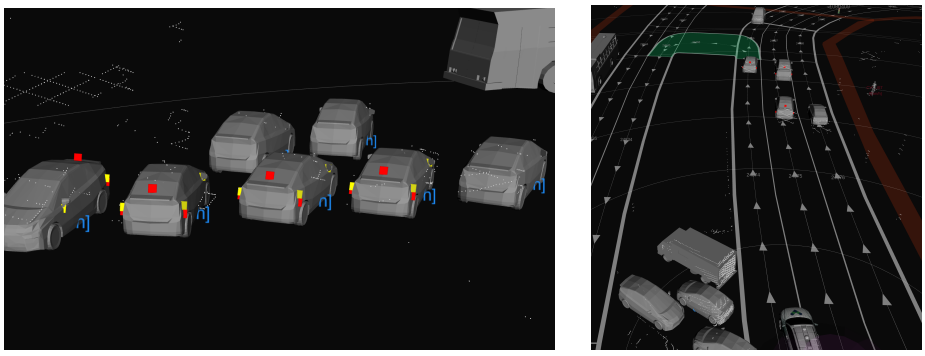

Figure 8: Visualization of sample Light State Markers on vehicles in RViz.

8 Outlook

Our current model was trained on a quite limited amount of data from a confined area. The amount of training and testing data should be increased to

improve the models robustness.

Another way of improving the models results could be by fusing surround view images together. This could be especially beneficial for vehicles that are close to the test vehicle and span across multiple camera images.

We observed that the model can struggle with accurate detection of the class for front lights. They can get mixed up with rear lights. Our current explanation for this is an imbalance in the training data, but this should be investigated further.

The training procedure should also be improved. It would be helpful to improve the data balance and weights in the loss function, add more augmentation and improve the learning rate scheduling. This is all currently being worked on.

Additionally, we want to integrate the output of this with the prediction module. While we already modified Autoware so that the predicted light states arrive in the 3D tracking and prediction modules, this data is currently not yet utilized to improve the prediction.

9 Conclusion

In this report we presented `Light Skeleton Detection`, a novel approach for angle agnostic signal state detection. We point out a current gap in the literature, where not much attention is payed to vehicle angle agnostic light state detection. We presented our proposed solution for filling this gap, explained our model architecture and evaluated it on real world data captured in a diverse urban environment and from cameras with different positions and field of view. Afterwards we presented how this model was easily integrated into Autoware by utilizing additional classifications for detected objects. This implementation was shown to run in real time on mobile hardware. It was shown that the current implementation could easily be extended to improve the prediction module in Autoware by relying on additional input data about a vehicles state.

The model checkpoints and code can be shared upon request.

References

[1] National Highway Traffic Safety Administration: *Analysis of Lane-Change Crashes and Near-Crashes*, 2009.

[2] Ponziani, Richard: *Turn Signal Usage Rate Results: A Comprehensive Field Study of 12,000 Observed Turning Vehicles.* SAE Technical Paper 2012-01-0261, SAE International, Warrendale, PA, 2012.

[3] Distributed Artifical Intelligence Laboratory: *BeIntelli - Future Mobility.* Webpage. https://be-intelli.com.

[4] Rapson, Christopher J., Boon-Chong Seet, Kate J. Lee, N. Asif Naeem, Mahmoud Al-Sarayreh Reinhard Klette: *Reducing the Pain: A Novel Tool for Efficient Ground-Truth Labelling in Images. Image and Vision Computing New Zealand (IVCNZ), Auckland, New Zealand, 19-21 November*, 2018.

[5] Greer, Ross, Akshay Gopalkrishnan, Maitrayee Keskar Mohan Trivedi: *Patterns of Vehicle Lights: Addressing Complexities in Curation and Annotation of Camera-Based Vehicle Light Datasets and Metrics*, 2023. arXiv:2307.14521 [cs].

[6] Hsu, Han-Kai, Yi-Hsuan Tsai, Xue Mei, Kuan-Hui Lee, Naoki Nagasaka, Danil V. Prokhorov Ming-Hsuan Yang: *Learning to tell brake and turn signals in videos using CNN-LSTM structure. IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[7] Lai, Ruili, Chumei Wen, Jingmin Xu, Delu Zeng Bo Wu: *VLS: Vehicle Tail Light Signal Detection Benchmark. Proceedings of the 2022 5th International Conference on Algorithms, Computing and Artificial Intelligence*, 1–6, Sanya China, 2022. ACM.

[8] Casares, Mauricio, Akhan Almagambetov Senem Velipasalar: *A Robust Algorithm for the Detection of Vehicle Turn Signals and Brake Lights. 2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, 386–391, 2012.

[9] Cui, Zhiyong, Shao-Wen Yang Hsin-Mu Tsai: *A Vision-Based Hierarchical Framework for Autonomous Front-Vehicle Taillights Detection and Signal Recognition. 2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 931–937, 2015.

[10] Wang, Zhenzhou, Wei Huo, Pingping Yu, Lin Qi, Shanshan Geng Ning Cao: *Performance Evaluation of Region-Based Convolutional Neural Networks Toward Improved Vehicle Taillight Detection.* Applied Sciences, 9(18), 2019. Number: 18 Publisher: Multidisciplinary Digital Publishing Institute.

[11] Shi, Peicheng, Heng Qi, Zhiqiang Liu Aixi Yang: *Research on intelligent vehicle lamp signal recognition in traffic scene*. SN Applied Sciences, 4(12), 2022.

[12] Jocher, Glenn, Ayush Chaurasia Jing Qiu: *YOLO by Ultralytics*, 2023.

[13] Kreiss, Sven, Lorenzo Bertoni Alexandre Alahi: *OpenPifPaf: Composite Fields for Semantic Keypoint Detection and Spatio-Temporal Association*. IEEE Transactions on Intelligent Transportation Systems, 1–14, March 2021.

[14] PyTorch: *Keypoint R-CNN*. Webpage. https://pytorch.org/vision/main/models/keypoint$_r$*cnn.html*.