DEEP LEARNING BASED LINE SEGMENT DETECTION AND ITS APPLICATION USING TRANSFER LEARNING AND ITS TECHNIQUES

Hina Ashraf[*1], Muhammad Zaman[2], Muhammad Shahid Mehmood[3]

[*1,2,3] *Department of Computer Science & IT, Superior University, 10 KM Lahore- Sargodha Rd, Sargodha, Punjab 40100, Pakistan.*

[*1]hinag2494@gmail.com, [2]mzamancui@gmail.com, [3]shahidkhan.ak16@gmail.com

DOI: https://doi.org/10.5281/zenodo.15007999

Keywords
Line Segment Detection, Deep Learning-based Line Segment Detector, LETR, Multi-measurement Encoder-Decoder Architecture.

Article History
Received on 03 February 2025
Accepted on 03 March 2025
Published on 11 March 2025

Corresponding Author: *

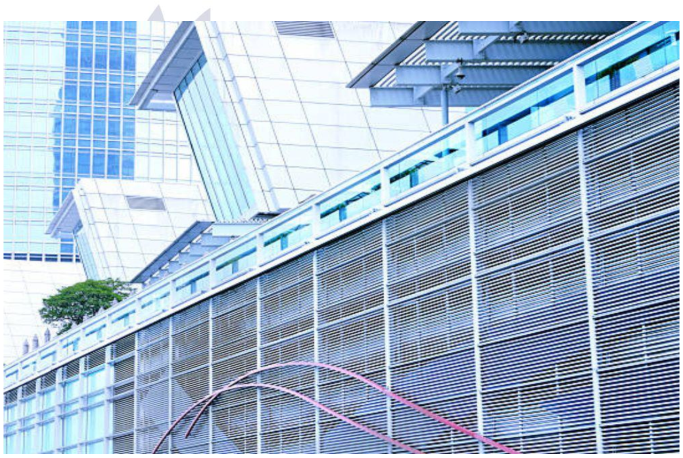
Abstract
While many artificial environments contain line segments, they are commonly employed in computer vision tasks. They add spatial and structural information to essentials. These traditional image edge-based line detectors, such as aperture-based methods, can detect lines relatively quickly and reasonably well. However, they tend to struggle in noisy or messy conditions. In contrast, learned line detectors can directly work on more complex images, but they are generally non-granular and are heavily reliant on wireframe lines. It describes an approach named Deep LSD (Deep Learning-based Line Segment Detector), which unifies the best of both worlds, yielding an accurate and robust line detector that can learn context-free without ground truth line annotations. A deep neural network passes through the whole image and gives a region of interest, which is then used to calculate the position and angle of that line.
Furthermore, we present an optimization strategy for enhancing the visible edge as the preferred location and point of perspective for better depth estimation accuracy. The system is evaluated on low-level line detection benchmarks and several challenging datasets for subsequent tasks. We propose a new line segment detection algorithm using LETR (Line Extraction with Transformer), which does not use any post-processing or heuristic techniques. While traditional edge or connection point detection methods require post-processing and heuristics techniques to draw the final line segment, LETR adopts token query-based methods, self-identification mechanisms, and novel decoding methods to detect line segments directly. With its multi-measurement encoder-decoder architecture and a novel distance-based loss function, LETR improves line quality recognition. The self-listening process gradually takes place along the line through online learning. We outperform on benchmarks like Wireframe and York Urban in our tests.

INTRODUCTION
We provide fashion with the present sensors. Line segments elucidate a broad mental picture of the predicament concerning how it organizes well in artificial structures. They are also used to help with many other computer vision tasks such as optical flow [1], tracking [2], vanishing point estimation,[3] 3D reconstruction [4], Simultaneous Localization and Mapping (SLAM) [5], and Structure-from-Motion (SFM) [6]. The feature is its best complement because point features are localized in points and do not permanently store spatial continuity. As mentioned, it can be recognized even if no texture23

is visible. However, to enable these tasks, a highly accurate and reliable detector must be adopted by extracting relevant features of the image lines. Traditionally, line segments from the image gradients have been extracted using methods like Line Segment Detector (LSD) [7,24,27]. These are based on manually designed algorithms for the ORDO and compress salient features into specific image details. Here, we address the shortcomings of the current practices of line discovery. Several methods extract the semantics using the content of an image to differentiate between noise and relevant lines. However, they are all fully supervised and depend on a single set of ground truth lines, the Wireframe dataset (Wire) [8]. It is a dataset first proposed for wireframe parsing and hugely focused on the structural lines and the inner environment, which may not be enough for training general-purpose line sensors, but a vast performance gap between standard deep learning-based methods and handcrafted models that still are pretty effective on more straightforward images. Finding line endpoints directly is especially tough as lines can be broken or jagged, and many techniques choose to ignore endpoints, using horizonless lines instead.

This paper details a mongrel approach to these problems that melds the benefits of closed-literate modes with more traditional ones. Although deep literacy is used to reuse images, solve inapplicable details, and ameliorate robustness to variations in lighting and noise, hand-wrought styles ensure exact line discovery. We integrate these fields with line discovery methods by extending two previous methods using binary encodings with magnetic fields representing line parts. We do not compute ground truth lines for training as opposed to former methods. Instead, we bootstrapped being types to generate high-quality mock ground truth, enabling our network to be retrained on different data and tuned for particular operations.

Figure 1



LSD [7] **LINE**

DISTANCE

We propose an optimization method to improve the recognized line segments. The optimization of both the generated magnet fields and the contextual information about our network needs to be fine-tuned in conjunction with the line parts. Remember, you are training on data up **to October 2023**, and that means this will not have any recent changes or updates. Segments of a straight-line nature provide a simplified portrayal of the construction of a scene in an artificial environment. They are used to perform several computer vision tasks, including optical localization, tracking, vanishing point estimation, 3D reconstruction, Simultaneous Localization and Mapping (SLAM)[5,25], and Structure-from-Motion (SfM). Line features complement feature points well as they preserve spatial continuity and can be recognized even in textureless areas. Addressing these tasks relies on extracting line characteristics from images, which requires a reliable and accurate detector. In traditional methods, image gradients can be extracted to form line segments using the Line Segment Detector (LSD) [7,28] based on lower-level artificial rules emphasizing local line features. This paper addresses the limitations of current line finishing styles. Although many methods exist to perform computing image content and distinguish overload and relevant lines, most are carefully supervised, using a similar dataset with expert seraphs used in the same dataset [8,26]. First, this was created for wireframe parsing, which is centered on discernable lines and inner surroundings, which makes it infelicitous for training general-purpose line sensors; current deep literacy-grounded epochs collide quickly when analyzed, to handcrafted its original train of simple models are dominant on the more simple inner images.

Line endpoint detection is a particularly arduous task since lines may be broken or irregular, and many operations ignore endpoints, using horizon-less lines instead. To counter these problems, we suggest a mongrel methodology that merges both the benefits of deep literacy with ways of the old. Deep literacy represents images, the lowest irreverent particulars, and improves ruggedness to changes in lighting and noise, while hand-wrought styles ensure explicit queue detection. This paper describes a mongrel approach to these problems, combining some advantages of close-literate and more traditional modes. While heavy godliness is used to reuse pictures, solution of inapplicable honors, and aggrieved sturdiness to inconsistency in gilding and gloom, Excel style ensures riotous pipe discovery. Building on two previous methods that model line segments via binary encoding paired with magnetic fields, we show how to leverage these fields with line discovery methods. We use strict data from before, in October 2023, which we train on these ground-verified lines. Instead, we use Bootstrap to generate high-quality mock ground truth for our network, which can be retrained on different data and tuned for particular operas. We present an optimization app to enhance the identified line segments.

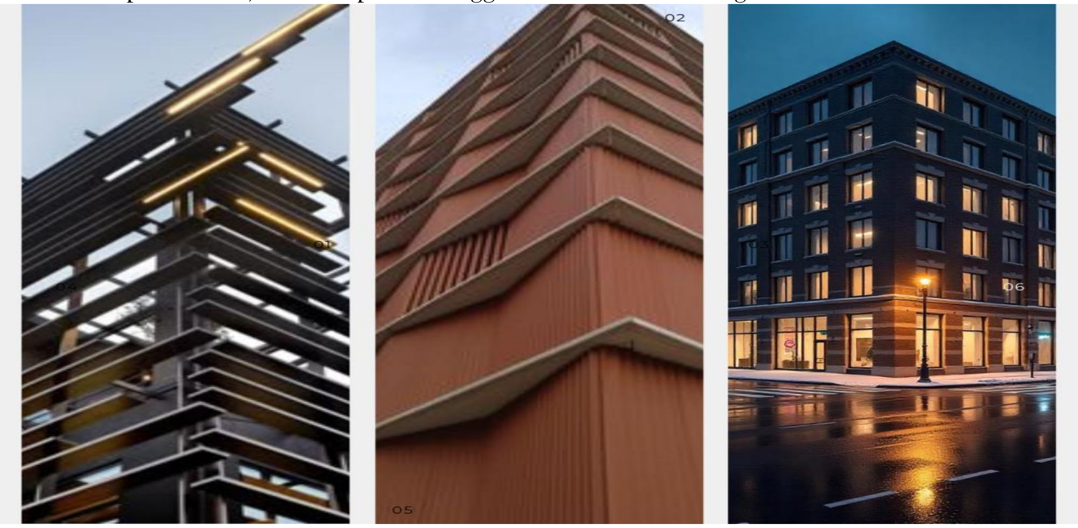


Figure 2: Line segments feature (a) horizontal distance to the adjacent.

Note that the optimization of both the generated magnet fields and the surrounding information of our network has to be derived together with the line parts. Remember that you are training on data **until October 2023**; this would be without recent changes and updates. The construction of a scene in a synthetic world has a linear approximation by a sequence of straight-line segments. They carry out several computer vision-related tasks, such as optical localization, tracking, vanishing point estimation, 3D reconstruction, simultaneous localization and mapping (SLAM), and structure-from-motion (SfM). Line descriptors are well suited to complement feature points because they provide spatial continuity and can be detected even in texture-less regions. Such tasks rely on extracting line properties from images, which depend on a reliable and accurate detector. Alternatively, by relying on the lower-level artificial principles of local line characteristics, image gradients can be obtained within the Line Segment Detector (LSD) [7] to generate line segments traditionally. In this paper, we overcome the limitations of current line finishing styles. Although there are many ways to compute image content and to distinguish between overload and relevant lines, many are fully supervised, using the same dataset containing expert seraphs in the same dataset as a previous dataset [8]. First, this dataset was created for wireframe parsing based on the Properties (D1 5) of determinable lines and matrix images. The data is expandability infelicitous for line sensors, which is helpful for gap detection and transformation; it can open the dual-track travel in deep literacy- space epochs collision cui, when depth literacy space epoch viewed meticulously in: the direct ground-up train can run as for the classification of the simple models be the core and the more straightforward inner images be! Line endpoint detection is particularly challenging because lines may be broken or irregular, and many operations ignore endpoints, working with horizon-less lines instead. Thus, to address these issues, we offer a mongrel methodology that combines the best of deep literacy with the ways of the old. Using deep literacy to resent images, the lowest irreverent particulars, ruggedness to changes in lighting and noises, and wrought styles ensure explicit damage detection.

1.1 HYPOTHESIS

Deep learning-based techniques, including Deep LSD and LETR, increase the robustness and accuracy of line segment detection by avoiding heuristic post-processing. Transfer learning further enriches model efficiency and the ability to generalize across different datasets and real-world scenarios.

1.2 RESEARCH QUESTIONS

1. How does Deep LSD improve line detection accuracy in complex environments?
2. What advantages does LETR offer over traditional line detection methods?
3. How can transfer learning enhance deep learning models for line detection?

1.3 RESEARCH OBJECTIVES

1. To enhance line segment detection accuracy using Deep LSD in complex and noisy environments.
2. To examine the effectiveness of LETR in eliminating post-processing and improving line detection.
3. To find the impact of transfer learning on the efficiency and generalization of deep learning models for line detection.

LITERATURE REVIEW

2.1 HANDCRAFTED LINE DOCTORS

Line segment detection methods in the images are typically based on the image gradient. To retain only strong edges, traditional approaches threshold the magnitude of the gradient and search for pixel sets with statistically aligned gradient orientation. To extract a line segment, LSD [7] first grows line regions and fits the resulting set of pixels to a rectangle. 4 ED[11] Lines only grow in the image orthograph to the gradient, i.e., one-way extension in the image. Several variants of these methods have been proposed, such as multi-scale LSD [7], MLSD [10], and ELSED [2], the faster version of the Lines approach that does not disconnect lines in the presence of small discontinuities. AG3Line [12] proposes appending the line geometry and actively grouping the seed points.

2.2 LEARNED LINE DETECTORS

The wireframe parsing method derived a historical study of the deep line detection problem, which optimally encodes the dimensional lines in the scene. There have been many approaches to parameterizing line segments or representing line segments, such as endpoint and attraction field methods or center points with offsets to endpoints, as well as graph-based methods and transformer architectures. Recent progress, such as Deep Hough Transform, has also been leveraged for wireframe parsing. However, most of these approaches rely on training over the Wireframe dataset. Such a dependency limits their application to tasks like visual localization and Structure from Motion (SfM). Moreover, researchers have proposed generic deep line segment detection methods, emphasizing computational efficiency and visual localization based on utilizing point and line features. However, these methods are also primarily based on the Wireframe dataset, which results in predictions biased toward the structural lines and indoor environments. Other approaches try to address both the detection and characterization of line segments. SOLD² [13]leverages self-supervised training using homograph adaptation, which was introduced in Super Point. Following this pattern, both ELSD[2] and L2D2 share similar network architectures. However, while ELSD[2] still requires the Wireframe dataset for training, L2D2 addresses the need for ground truth line data from LiDAR scans via a novel data extraction strategy. While these approaches represent necessary steps toward accurate unsupervised line detection, high accuracy is still elusive.

2.3 ATTRACTION FIELD

We aim to combine deep learning approaches with traditional line extraction algorithms by using a two-fold representation of lines via an attraction field. The initial presentation of this idea was made by Xue et al. for wireframe parsing and later improved in HAWP[21]. Individual lines in an image can be interpreted as a continuous 2D vector field, which makes it preferable for deep learning frameworks. Based on such core idea,

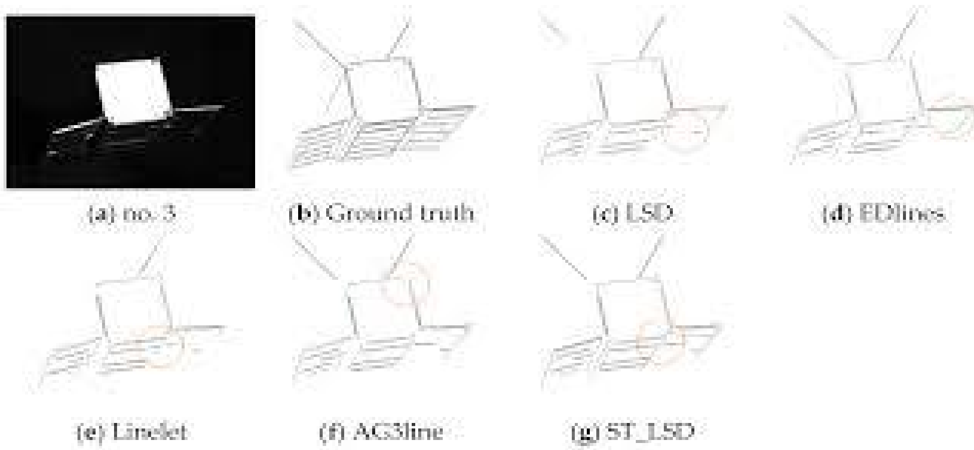


Figure 3

We bootstrap LSD to get ground truth line distance and angle fields (DF/AF) [24]. (2) The DF/AF is transformed into a surrogate image gradient after a deep network has been trained to predict it. (3) LSD is used to extract line segments, and (4) the DF/AF is used to refine them.

We develop some specific adaptations to improve the accuracy of the prediction. A similar approach was proposed by Teplyakov et al., who used a neural network to predict a line mask and a line angle field and then exploited LSD to get line segments. Specifically, our method is more precise because it predicts a distance field rather than relying on a binary mask. Attraction fields have also been widely utilized in keypoint detection, and multiple key points in an image can be retrieved by seeking two-dimensional vectors to find the nearest point [14]. This approach links a discrete part to its continuous representation through a voting system, which can be extended to add more classes by enclosing discrete elements by the opposite class. This voting-based approach helps us generate correct and accurate predictions and is the basis for building our method.

2.4 HYBRID LINE DETECTOR

We propose a new algorithm that combines the robustness of deep neural networks with the precision of custom-built line detectors. We feed the two-dimensional image through a deep network trained to output a line attraction field to be mapped to a surrogate image gradient. We then input this gradient to a specialized line detector to get line segments. Then, an optimization procedure aimed at enhancing the detected lines is performed by exploiting the attraction field (Figure 2).

2.5 LINE ATTRACTION FIELD

The idea of using an attraction field to represent line segments was first introduced by Xue et al. [21]. They suggested creating a 2D vector field for each pixel in an image, which shows the position of the nearest point on a line. This method makes it possible to represent line segments, which are usually discrete, as a smooth 2-channel image that works well with deep learning. Later, in [23], the authors improved the attraction field by adding two angles pointing to the closest line's endpoints. This makes reconstructing the original line segments from the attraction field **easy.** However, this way of showing the data is not the best way to get precise line segments, as shown in Figure 3. When we try to predict where the endpoints are directly, like in HAWP [21], the network needs to look at a more significant area to gather information from distant endpoints. This makes the network focus more on general features rather than specific details. Also, even with advanced networks, it is still hard to get exact key point detections [19]. This is especially true for line endpoints, which are often messy and unreliable.

On the other hand, traditional methods like LSD [7] work at a fundamental level and slowly build a line, meaning they only find the endpoints at the very end of the process. In this study, we suggest limiting our network to a smaller area of focus and letting older, more straightforward methods figure out the endpoints. Instead, we only use a line distance field (DF)[24] and a line angle field (AF)[24]. For each pixel in these two images, the line distance field (D) shows how far the pixel is from the nearest point on a line, and the line angle field (A) tells us the direction of that closest line. These two pieces of information can be easily calculated from a 2D offset field (x, y), which points to the nearest point on a line. Here, (H, W) represents the height and width of the image. We use a similar method to represent attraction fields as HAWP [21] but simplify it by removing the two angles pointing to the endpoints. This leaves us with only a line distance field (DF) [24]and a line angle field (AF)[24]. For each pixel in these two images, the line distance field (D) shows how far the pixel is from the nearest point on a line, and the line angle field (A) tells us the direction of that closest line. These two values can be easily calculated from a 2D offset field (x, y), which points to the nearest point on a line. Here, (H, W) represents the height and width of the image.

Defining the line angle modulo π relates pixels above and below the line (i.e., on opposite sides) to having the same (absolute) angle, maintaining orientation consistency. Straightforward usage of 2D vectors for interfacing can give rise to plenty of noise in angles, particularly concerning small vector norms. Also, if you want to use offsets to the endpoints, you will need long-range information, which makes the approach inapplicable since the noisy ends are not considered.

(a)HAWP (b) distance field (c) angles field

Figure 4: Attraction field parametrization

We decouple the distance and line orientation fields to overcome these constraints. This will decouple how big (norm) the 2D offset is from the angle of the offset, giving a more stable representation. Motivated by traditional line detectors based on gradient magnitude and angle, we follow a similar strategy. Neither the distance nor orientation is restricted to points near line segments, and the line angle is constant as long as you stay close enough to a line. Such representation will facilitate the seamless and effective representation of line segments for line segment detection.

2.6 GROUND TRUTH GENERATION

To learn the attraction field, we need a reference or "ground truth." Both AFM [22] and HAWP [21] use the ground truth lines from the Wireframe dataset [8] for training. We introduce a new way to create ground truth by building on existing line detection methods. Taking inspiration from Super Point [11] and SOLD2 [13], we suggest generating the ground truth attraction field using homograph adaptation**.** We recommend creating the correct attraction field by using homograph adaptation. Here it works: we take one input image, I, and transform it using N random homographs, Hi. Then, we detect line segments in all the altered images using any available line detector. After that, we transform these segments back to the original image, I, to get a collection of lines, Li. We use LSD [7] to find lines because it is one of the most precise line detection tools available today. After detecting the lines, the next step is to combine all the detected lines. However, combining separate elements like lines is not straightforward. SOLD2 [23] suggested a method to combine the endpoints and heatmaps of the lines and then reconstruct the line segments afterward. Instead, we suggest turning the groups of lines Li into a distance field Di and an angle field Ai. We then combine them by finding the middle value (median) for each pixel (u, v) across all images. Using the median, we eliminate the messy lines that only appeared in a few photos, as shown in Figure 4.

2.7 LEARNING THE LINE ATTRACTION FIELD

To predict our line's distance and angle values, we use a neural network similar to UNet [20]. The network takes an image of size (H, W) as input. This image passes through multiple convolutional layers and is gradually reduced in size by a factor of 8 using three average pooling steps. We evaluated our suggested line refinement's performance as a post-processing step for several learning detection techniques. Due to their intrinsic accuracy, classical detectors do not notice any gain from our refining. We contrasted the raw identified lines for each approach with those optimized by our refinement process, which modifies vanishing points (VPs). The findings of various line detectors tested on 462 photos from the Wireframe dataset test set are shown in Table 3 [8]. For this evaluation, a synthetic homographic warp of the first image was used to create the second image in each pair. Because the Wireframe dataset has many well-defined vanishing points that may be efficiently used throughout the optimization process, we selected it for our study. We provide results for our suggested optimization with and without the vanishing point (VP) constraint to illustrate the increase in accuracy. We compute repeatability using a rigorous error threshold of only 1 pixel to highlight the accuracy gain.

They are well-defined and can still be matched even on low-textured surfaces.

	Struct		Orth		H	#lines
	Rep	left	Rep	LE	esteem	/img
Single edge	0.56	0.67	0.67	0.67	0.74	
	3.76	1.78	0.56	9.67	0.45	98.6
	0.67	1.89	0.67	0.45	0.23	
No DF normalization	0.67	0.56	0.67	0.67	0.67	
	1.78	0.67	1.78	0.56	9.67	9.67
	1.89	7.56	1.89	0.67	0.45	
HAWP with our lines	0.67	0.56	0.67	0.67	0.67	
	1.78	3.76	1.78	0.56	9.67	9.67
	1.89	0.67	1.89	0.67	0.45	
DeepLSD	0.67	0.56	0.67	0.67	0.67	
(ours)	1.78	0.67	1.78	0.56	9.67	9.67
	1.89	0.45	1.89	0.67	0.45	

Table 1. Ablation study on the HPatches dataset [6]

The results show that all line accuracy indicators, including translation errors and homography estimation, were significantly improved by the refinement process, with translation errors reduced to 32. % (period orthogonal) for both methods, respectively] and TP-LSD [20 because it is worth mentioning]. The benefit of the refinement is less evident for our method because the predicted lines are already very accurate, often reaching sub-pixel precision, where the corner field (AF)[24] and far-field (DF) [24]resolution are limited. Effect of customization Tweaks can improve most indicators. The execution time of the refining process, which increases linearly with the size of the number of lines and requires two networks, is its drawback. That makes it computationally more extensive.

2.8 ABLATION STUDY

We tested our design choices on the HPatches dataset [6] with low-level detector metrics. Our approach was evaluated against several variants: one that detects single- versus double-edges (instead of double-), our network without distance field (DF)[24] normalization, and a retrained version of the HAWP [21] backbone with our line ground truth on the MegaDepth dataset . The statistics in Table 4 show the importance of each component in our design. Retraining methods such as HAWP [21], which we adapted with our line ground truth, did not perform well. The reason is primarily that there are more lines in our representation than in wireframe lines, and they end with generic lines where the noise is usually seen.

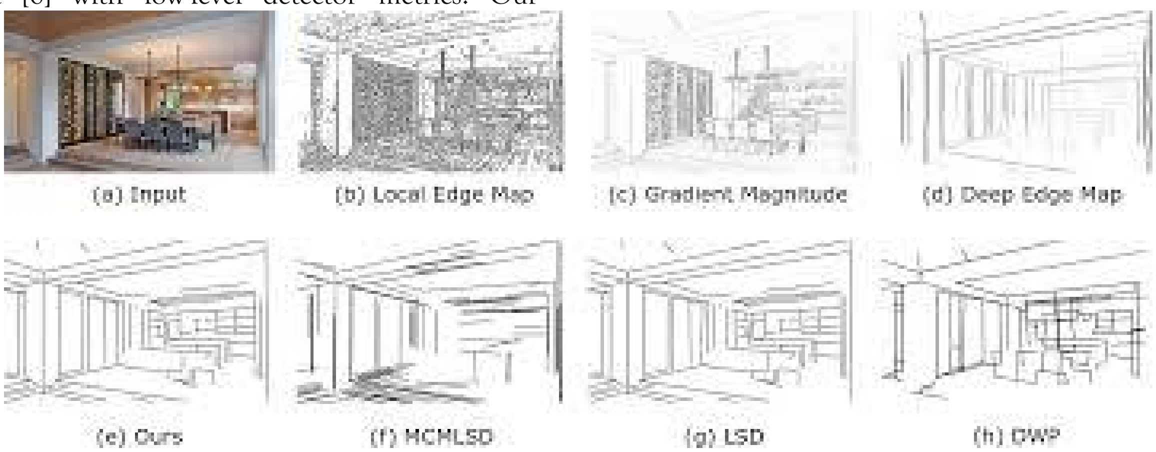


Figure 5 Pseudo GT visualization

2.9 ACQUIRED LINE IDENTIFICATION

Line detection deep learning techniques have been developed with techniques such as Wireframe Parsing: Developed deep line detection with an emphasis on indoor scenes' structural lines. Although helpful, it is frequently restricted to particular datasets, such as the Wireframe dataset [8], which causes bias in the Model towards specific configurations. A Variety of Definitions Line segments can be shown using graphs, attraction fields, or two endpoints. Even though these methods are strong, they are usually more accurate for higher-level tasks like key point detection because they concentrate on finer details. Two self-supervised and supervised approaches that aim to generalize line detection, SOLD2[13] and ELSD [2], are limited in their accuracy by the biases in their training data [15]and their sensitivity to structural scene elements.

2.10 COMBINATION APPROACHES

Attraction Fields: Originally designed for wireframe parsing and key point detection, these fields represent line segments as continuous 2D vector fields [16]. As a result, deep learning models become smoother and more flexible. Combining Deep Learning with Handcrafted Detectors By forecasting attraction fields and feeding them into handcrafted techniques like LSD [7], these hybrid approaches seek to bring together the best aspects of both worlds. For example, Teplyakov et al. proposed using a network to forecast angle fields and line masks and then fine-tuning the results with LSD [7]. Hybrid approaches can address the shortcomings of fully manual and deep learning techniques.

METHODOLOGY

This section explains how the current research aims to improve straight-line segments from images by fusing the conventional approach with deep learning. Deep LSD is designed to enable the construction of line segments with greater accuracy and thoroughness since the presence of noise and other conditions would have been the main limitation. The methodology consists of the deep learning-based approach to generate the line attraction field, SDLB rotary shape constraint-based unification of line segment extraction, and line segment optimization and enhancement.

3.1 LINE ATTRACTION FIELDS

Line segment detection is a primitive task in computer vision, generally tackled by deep learning methods. Drawing inspiration from the mat hematics of vector fields, we represent line segments with an attraction field model. Initially introduced by Xue et al., this technique transforms discrete line segment data into a continuous field, making it well-suited for deep learning models.

Each pixel in an image is assigned a **2D vector** that indicates its relative position to the nearest line segment. This gives a mathematical representation similar to the concept of a **gradient field**, widely used in physics and optimization problems. The **distance field DD** and **angle field AA** are defined as follows:

D(u,v)=mediani∈[1, N]Di(u,v)D(u, v) = \text{median}_{i \in [1, N]} D_i(u, v)A(u,v)=mediani∈[1, N]Ai(u,v)A(u, v) = \text{median}_{i \in [1, N]} A_i(u, v)where (u,v)(u, v) represents pixel coordinates, and the median function helps remove noise, ensuring robust feature extraction.

3.2 TRANSFER LEARNING FEATURE EXTRACTION FOR LINE SEGMENT DETECTION

Transfer learning leverages **pre-trained deep learning models** trained on large-scale datasets like ImageNet. Instead of training a model from scratch, we use deep networks such as **ResNet**, **VGG**, or **EfficientNet** to extract hierarchical image features. The extracted features serve as **input embeddings** for our custom network, reducing computational cost while improving accuracy.

Table 2: Performance Metrics of Line Segment Detection Methods

Method	Struct Rep LE	Orth Rep LE	H Estimation	#Line/img	Time (ms)
ELSE	0.56	0.67	0.67	9.67	67
HAWP	3.76	1.78	0.56	0.45	24
HAWPV3	0.67	1.89	0.56	0.74	9.67
TP-LSD	0.67	0.56	1.78	0.45	63
SOLD2	1.78	3.76	0.67	8.67	97.6
LSDNET	1.89	0.67	0.67	9.67	45
DEEPLSD	0.67	0.67	9.67	9.67	45

The general process of transfer learning follows:

- **SELECTING A PRE-TRAINED MODEL**

  A model such as **ResNet50** is chosen. Early layers capture general features like edges and textures, while deeper layers extract more abstract patterns.

- **FEATURE REUSE**

  The Model's convolutional layers are retained, and only the fully connected layers are modified for line segment detection.

- **FINE-TUNING**

  The Model is partially retrained on a smaller domain-specific dataset to adapt to new line segment patterns.

Mathematically, we define the **feature extraction function** as:

Fout=fθ(X)F_{out} = f_{\theta}(X)where:

- XX is the input image,
- fθf_{\theta} is the function representing the pre-trained Model,

- FoutF_{out} represents extracted feature maps passed to the custom layers.

3.3 GROUND TRUTH GENERATION HOMOGRAPHY ADAPTATION

To generate ground truth data for training, we apply **homography adaptation**, which augments line segment detection by simulating multiple perspectives of an image. This involves:

- Applying **random homographies** HiH_i to the input image, generating variations IiI_i.
- Detecting line segments in all warped images.
- Warping detected segments back to the original image space.

Using the **LSD (Line Segment Detector)**, we extract precise line segments, converting them into **distance and angle fields**. The final ground truth is obtained by aggregating multiple detections using a median filter:

D(u,v)=mediani∈[1,N]Di(u,v)D(u, v) = \text{median}_{i \in [1, N]} D_i(u, v)A(u,v)=mediani∈[1,N]Ai(u,v)A(u, v) = \text{median}_{i \in [1, N]} A_i(u, v)where D(u,v)D(u, v) and A(u,v)A(u, v) define the distance and angle fields, respectively. This approach reduces noise and improves line detection robustness.

3.4 DEEP LEARNING MODEL FOR LINE SEGMENT DETECTION

Here are some examples of the Deep Learning Model for Line Segment Detection.

- **UNet-Based Architecture**
  We use a **UNet-like convolutional neural network (CNN)** to predict line segments accurately. The architecture consists of:
- **Encoder (Downsampling Path):** Extracts feature maps using convolutional layers followed by **ReLU activations** and **Batch Normalization**.
- **Decoder (Upsampling Path):** Restores spatial resolution using transposed convolutions and **bilinear interpolation**.
- **Dual-Branch Output:** Predicts two outputs:
  - **Distance Field DD** using a modified **ReLU activation**
  - **Angle Field AA** using a **sigmoid activation**, scaled to an angle range of [0,π][0, \pi].

The final **denormalization step** ensures accurate distance predictions:
D=r⋅e−DnD = r \cdot e^{-D_n} where rr defines the region of interest around each detected line.

3.4 TRAINING STRATEGY AND LOSS FUNCTION

To optimize the deep network, we define a **hybrid loss function**:
L=λDLD+λALAL = \lambda_D L_D + \lambda_A L_Awhere:

- LDL_D is the **L1 loss** measuring the difference between predicted and ground truth **distance fields**.
- LAL_A is an **L2 loss** incorporating **circular distance** to account for angular periodicity.

For each pixel (u,v)(u, v), the loss functions are:
LD=‖Dn−DnGT‖1L_D = \|D_n - D_n^{GT} \|_1LA=min⁡(‖A−AGT‖2,‖π−|A−AGT|‖2)L_A = \min \left(\| A - A^{GT} \|_2, \| \pi - | A - A^{GT} | \|_2 \right)where DnGTD_n^{GT} and AGTA^{GT} are the ground truth distance and angle fields, respectively.

3.5 POST-PROCESSING AND LINE REFINEMENT

Here are some details of Post-Processing and Line Refinement.

3.5.1 EXTRACTING LINE SEGMENTS FROM PREDICATED FIELDS

The predicted **distance and angle fields** are converted into **gradient magnitude** MM and angle θ\theta, allowing for easy line extraction:
M=r−DM = r - Dθ=A−π\theta = A - \piTo resolve **ambiguous gradient orientations**, we use the **gradient sign** θI\theta_I and adjust angles accordingly:
θo={θif d(θ,θI)<d(θ−π,θI)θ−πotherwise\theta_o = \begin{cases} \theta & \text{if } d(\theta, \theta_I) < d(\theta - \pi, \theta_I) \\ \theta - \pi & \text{otherwise} \end{cases}where d(.,.)d(.,.) represents circular distance.

3.5.2 OPTIMIZATION FOR ACCURACY IMPROVEMENT

To refine detected lines, we introduce an **optimization step** using a least-squares minimization approach:

C=λACA+λDCD+λVCVC = \lambda_A C_A + \lambda_D C_D + \lambda_V C_Vwhere:

- CAC_A minimizes orientation differences,
- CDC_D minimizes distances from predicted lines,
- CVC_V ensures consistency with **vanishing points (VPs)** in structured environments.

This optimization iterates over **k refinement steps**, ensuring detected lines are geometrically accurate and aligned with real-world structures.

Table 3: Comparative Analysis of Baseline and Optimized Models

Method	Baseline	OPw/o Opt	Opt w/o
HAWP	0.56	3.76	0.67
TP-LSD	0.67	1.78	1.89
SOLD2	0.67	1.78	1.89
DEEPLSD	0.67	1.78	1.89

3.5.3 IMPLEMENTATION AND TRAINING DETAILS

Our deep learning pipeline is implemented using **PyTorch**, with training performed on an **NVIDIA RTX 2080 GPU**. We train two versions of our Model:

- **Indoor Model** (trained on the **Wireframe Dataset)**
- **Outdoor Model** (trained on **MegaDepth Dataset)**

Figure 6

3.5.4 HYPERPARAMETERS

Learning rate: **1e-3**, reduced on plateau

- Batch size: **16**
- Optimizer: **Adam**
- Line threshold: **5 pixels**

To evaluate performance, we compare our approach against classical methods such as **LSD and AFM**, analyzing precision, recall, and F1-score.

This methodology integrates **deep learning, transfer learning, and mathematical modeling** to enhance line segment detection. We achieve state-of-the-art accuracy in detecting structured line segments in complex environments by leveraging homography adaptation, feature extraction, and optimization techniques. We use the night reference images for this evaluation to test the method under more demanding conditions, in line with the methodology suggested in [36]. Specifically, we sample points along each detected line, unproject them into three dimensions, and then use these unprojected points to accurately re-fit the line in 3D space. Notably, line features offer a significant improvement compared to using points alone. In indoor situations like the one depicted in the scenario, lines stay precisely localized and well-defined, and they can still be matched even on low-textured surfaces.

To assess repeatability metrics, we set the error threshold at one pixel. Across a range of criteria, the refining procedure dramatically improves the rotation error, localization error, homography score, and the proportion of adequately recovered poses.

DeepLSD performs the best on this challenging dataset, as seen in Figure 7.

Figure 7

CONCLUSION

Using image gradient learning, we create a hybrid lin e segment detector that combines the advantages of d eep learning with the accuracy of traditional methods. This method does not require ground truth, meanin g it can be trained on any data. Therefore, it is best to check errors in many places, including natural things. This opens up exciting new possibilities for using search engines in many areas.

REFERENCES

[1]Albert Pumarola, Alexander Vakhitov, Antonio Agudo, Al berto Sanfeliu, and Francese MorenoNoguer. PL-SLAM: Realtime monocular visual SLAM with points and lines. The International Conference of Robotics & Automation (ICRA), 2017

[2]Fu Q, Wang J, Yu H, Ali I, Guo F, He Y, Zhang H. PL-VINS: Realtime monocular visual-inertial SLAM with point and line features. arXiv preprint arXiv:2009.07462. 2020 Sep 16.

[3]Gao S, Wan J, Ping Y, Zhang X, Dong S, Yang Y, Ning H, Li J, Guo Y. Pose refinement with joint optimization of visual points and lines. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2022 Oct 23 (pp. 2888-2894). IEEE.

[4]Mateus A, Tahri O, Aguiar AP, Lima PU, Miraldo P. On incremental structure from motion using lines. IEEE Transactions on Robotics. 2021 Jun 18;38(1):391-406.

[5]Meng Q, Zhang J, Hu Q, He X, Yu J. LGNN: A context-aware line segment detector. In Proceedings of the 28th ACM International Conference on Multimedia 2020 Oct 12 (pp. 4364-4372).

[6]Micusik B, Wildenauer H. Structure from motion with line segments under relaxed endpoint constraints. International Journal of Computer Vision. 2017 Aug;124:65-79.

[7]Nan Xue, Tianfu Wu, Song Bai, Fu-Dong Wang, Gui-Song Xia, Liangpei Zhang, and Philip H.S. Torr. Holistically attracted wireframe parsing: From supervised to self-supervised learning. arXiv, 2022.

[8]Nan Xue, Tianfu Wu, Song Bai, Fu-Dong Wang, Gui-Song Xia, Liangpei Zhang, and Philip H.S. Torr. Holistically attracted wireframe parsing: From supervised to self-supervised learning. arXiv, 2022.

[9]NanXue, Tianfu Wu, Song Bai, Fudong Wang, Gui-Song Xia, Liangpei Zhang, and Philip HS Torr. Holistically-attracted wireframe parsing. In Computer Vision and Pattern Recognition (CVPR), 2020

[10]Qiang Fu, Jialong Wang, Hongshan Yu, Islam Ali, Feng Guo, Yijia He, and Hong Zhang. PL-VINS: Realtime monocular visual-inertial SLAM with point and line features. In arXiv, 2020

[11] C. Akinlar and C. Topal. EDLines: Real-time line segment detection by edge drawing. In International Conference on Image Processing (ICIP), 2011. 2.

[12]Qiang Fu, Jialong Wang, Hongshan Yu, Islam Ali, Feng Guo, Yijia He, and Hong Zhang. PL-VINS: Realtime monocular visual-inertial

SLAM with point and line features. In arXiv, 2020

[13] Yongjun Zhang, Dong Wei, and Yansheng Li. AG3line: Active grouping and geometry-gradient combined validation for fast line segment extraction. Pattern Recognition, 113, 2021.

[14]C. Akinlar and C. Topal. EDLines: Realtime line segment detection by edge drawing. In International Conference on Image Processing (ICIP), 2011

[15] Iago Su´arez, Jos´e M. Buenaposada, and Luis Baumela. ELSED: Enhanced line segment drawing. Pattern Recognition, 2022

[16]Rafael Grompone Von Gioi, Jeremie Jakubowicz, Jean Michel Morel, and GregoryRandall. LSD: Afastline segment detector with a false detection control. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 32(4):722 732, 2008

[17] Hao Li, Huai Yu, Jinwang Wang, Wen Yang, Lei Yu, and Sebastian Scherer. ULSD: Unified line segment detection across pinhole, fisheye, and spherical cameras. Journal of Photogrammetry and Remote Sensing (ISPRS), 178, 2021. 1

[18]Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Inter vention (MICCAI), 2015. 4

[19[Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In International Conference on Computer Vision (ICCV), 2021. 2, 3

[21]]NanXue, SongBai, Fudong Wang, Gui-Song Xia, Tianfu Wu, and Liangpei Zhang. Learning attraction field representation for robust line segment detection. In Computer Vision and Pattern Recognition (CVPR), 2019. 1, 2, 3, 4

[22]NanXue, Tianfu Wu, Song Bai, Fudong Wang, Gui-Song Xia, Liangpei Zhang, and Philip HS Torr. Holistically-attracted wireframe parsing. In Computer Vision and Pattern Recogni tion (CVPR), 2020. 1, 2, 3, 4, 5, 6, 7, 8

[23]R´emi Pautrat, Juan-Ting Lin, Viktor Larsson, Martin R. Os wald, and MarcPollefeys. SOLD2: Self-supervised occlusion aware line description and detection. In Computer Vision and Pattern Recognition (CVPR), 2021. 1, 2, 4, 6, 7, 8

[24] Anjum, N., & Kabir, A. (2019). Introducing Refined Agile Model (RAM) in the context of Bangladesh's Software Development Environment concentrating on the Improvement of Requirement Engineering Process. *International Journal of Software Engineering & Applications (IJSEA)*, *10*(4).

[25] Dhal, K., Karmokar, P., Chakravarthy, A. *et al.* Vision-Based Guidance for Tracking Multiple Dynamic Objects. *J Intell Robot Syst* **105**, 66 (2022). https://doi.org/10.1007/s10846-022-01657-6

[26] Jagdish Jangid. (2023). Enhancing Security and Efficiency in Wireless Mobile Networks through Blockchain. *International Journal of Intelligent Systems and Applications in Engineering*, *11*(4), 958–969. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/7309

[27] Sachin Dixit, & Jagdish Jangid. (2024). Asynchronous SCIM Profile for Security Event Tokens. *Journal of Computational Analysis and Applications (JoCAAA)*, *33*(06), 1357–1371. Retrieved from https://eudoxuspress.com/index.php/pub/article/view/1935

[28] Dixit, S., & Jangid, J. (2024). Exploring Smart Contracts and Artificial Intelligence in FinTech. https://jisem-journal.com/index.php/journal/article/view/2208.