Machine Vision and Applications (2021) 32:12
https://doi.org/10.1007/s00138-020-01138-6

ORIGINAL PAPER

LS-Net: fast single-shot line-segment detector

Van Nhan Nguyen[1] · Robert Jenssen[1] · Davide Roverso[2]

Received: 18 March 2020 / Revised: 23 September 2020 / Accepted: 6 October 2020 / Published online: 29 October 2020

Abstract
In unmanned aerial vehicle (UAV) flights, power lines are considered as one of the most threatening hazards and one of the most difficult obstacles to avoid. In recent years, many vision-based techniques have been proposed to detect power lines to facilitate self-driving UAVs and automatic obstacle avoidance. However, most of the proposed methods are typically based on a common three-step approach: (i) edge detection, (ii) the Hough transform, and (iii) spurious line elimination based on power line constrains. These approaches not only are slow and inaccurate but also require a huge amount of effort in post-processing to distinguish between power lines and spurious lines. In this paper, we introduce LS-Net, a fast single-shot line-segment detector, and apply it to power line detection. The LS-Net is by design fully convolutional, and it consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor. Due to the unavailability of large datasets with annotations of power lines, we render synthetic images of power lines using the physically based rendering approach and propose a series of effective data augmentation techniques to generate more training data. With a customized version of the VGG-16 network as the backbone, the proposed approach outperforms existing state-of-the-art approaches. In addition, the LS-Net can detect power lines in near real time. This suggests that our proposed approach has a promising role in automatic obstacle avoidance and as a valuable component of self-driving UAVs, especially for automatic autonomous power line inspection.

Keywords Line segment detection · Power line detection · Power line inspection · Deep learning · UAVs

1 Introduction

Obstacle detection and avoidance are the key to ensure low altitude fight safety. Due to their extremely small size, power lines are considered as one of the most threatening hazards and one of the most difficult obstacles for unmanned aerial vehicles (UAVs) to avoid [31].

In automatic autonomous vision-based power line inspection, power line detection is crucial, not only for ensuring flight safety, and for vision-based navigation of UAVs, but also for inspection to identify faults on power lines (e.g., corroded and damaged power lines) and surrounding objects, such as vegetation encroachment [26].

In recent years, many techniques have been proposed to detect power lines automatically. However, most of the proposed methods are typically based on a common three-step approach: First, an edge detector such as Canny [6] is applied to produce edge maps. Then, the Hough transform [7], the Radon transform, or a line tracing algorithm, are utilized to detect straight lines from the edge maps. Finally, power line constraints, such as parallel lines, are applied to eliminate spurious lines and detect the power lines. These approaches not only are slow and inaccurate but also require a considerable amount of effort in post-processing to distinguish between power lines and spurious lines.

With the aim of facilitating real-time and accurate power line detection for UAV vision-based navigation and inspection, we propose in this paper LS-Net, a fast single-shot line-segment detector, and apply it to power line detection.

The work presented in this paper is part of an ongoing effort involving the exploitation of recent advances in deep learning (DL) and UAV technologies for facilitating auto-

✉ Van Nhan Nguyen
nhan.v.nguyen@esmartsystems.com

Robert Jenssen
robert.jenssen@uit.no

Davide Roverso
Davide.Roverso@esmartsystems.com

[1] The UiT Machine Learning Group, UiT The Arctic University of Norway, 9019 Tromsø, Norway

[2] Analytics Department, eSmart Systems, 1783 Halden, Norway

matic autonomous vision-based inspection of power lines. In our previous work [26], we first proposed a novel automatic autonomous vision-based power line inspection concept that uses UAV inspection as the main inspection method, optical images as the primary data source, and deep learning as the backbone of data analysis. We then identified six main challenges of DL vision-based UAV inspection: the lack of training data; class imbalance; the detection of small power components and faults; the detection of power lines in cluttered backgrounds; the detection of previously unseen components and faults; and the lack of metrics for evaluating inspection performance.

To move forward, we proposed approaches to address the first three challenges and built a basic automatic vision-based inspection system with two custom-built UAVs and five DL-based models for data analysis and inspection [27].

In this paper, we take this further by addressing the fourth challenge of DL vision-based UAV inspection, which is to detect power lines in cluttered backgrounds, with our proposed LS-Net. The LS-Net is a feed-forward, fully convolutional neural network (CNN) [37] and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor. Due to the unavailability of large datasets with annotations of power lines, we render synthetic images of power lines using the physically based rendering (PBR) approach [18] and propose a series of effective data augmentation techniques to generate more training data. With a customized version of the VGG-16 network [34] as the backbone, the proposed LS-Net outperforms existing state-of-the-art DL-based power line detection approaches and shows the potential to facilitate real-time power line detection for obstacle avoidance in low-altitude UAV flights.

The contribution of this paper is fourfold. First, we propose a novel single-shot line segment detector, called LS-Net. The proposed LS-Net can be trained end-to-end via a weighted multitask loss function, which is a combination of Focal loss [13] for addressing the class imbalance in classification and Wing loss [8] for restoring the balance between the influence of errors of different sizes in multiple points regression. Second, we resolve the issues of single-shot detectors, which typically employ a traditional one-grid approach, when applied to line segment detection by proposing a four-grid approach. To the authors knowledge, such an approach is new in the single-shot approaches based on CNNs. Thirdly, we address the lack of training data by using synthetic data rendered by the PBR approach and applying a series of effective data augmentation techniques to generate more training data. Finally, this work is in our opinion paving the way for fully automatic autonomous vision-based power line inspection, in which high-speed UAVs equipped with sensors, cameras, a DL vision-based UAV navigator, and a DL-based model for data analysis, can automatically navigate along power lines to collect data for offline inspections and perform online inspections to identify potential faults quickly.

The remainder of the paper is structured as follows: Sect. 2 presents background knowledge and relevant related work, before we describe our proposed LS-Net in Sect. 3. Next, in Sect. 4, we present in detail our experimental results and ablation studies. Then, in Sect. 5, we discuss the potential of our proposed LS-Net in UAV navigation and UAV inspection as well as in detecting other linear structures. Finally, in Sect. 6, we conclude the paper with a summary.

2 Background and related work

In the past few years, many approaches to power line detection have been proposed. These approaches can be roughly categorized as (i) line-based methods; (ii) piece-wise line segment-based methods; (iii) auxiliaries assisted methods; (iv) and DL-based methods.

2.1 Line-based methods

A straight-forward approach to power line detection is to treat the power line as a straight line and apply line detection algorithms directly. For example, Li et al. utilized the Hough transform to detect straight lines from pulse-coupled neural network filtered images and employed K-means clustering to discriminate power lines from other mistakable linear objects [22,23].

Although this approach is effective and easy to implement, its strong assumptions on the characteristics of power lines, including (i) a power line has uniform brightness, (ii) a power line approximates a straight line, and (iii) power lines are approximately parallel to each other, make it a less practical approach. Due to the strong assumptions, line-based methods often mistakenly detect linear objects, such as metallic fence lines [22], as power lines and misdetected power lines that appear as arc curves due to the influence of gravity [31].

2.2 Piece-wise line segment-based methods

With the aim of detecting both straight power lines and curvy ones, some researchers have proposed to segment a power line into piece-wise line segments so that they can be approximated by straight lines [36,42]. For example, Yan et al. utilized the Radon transform to extract line segments of a power line, then employed a grouping method and the Kalman filter to link each line segment, and connect the linked line segments into a complete line [42]. Song et al. applied matched filter and first-order derivative of Gaussian to detect line segments, then used a graph cut model based on

Fig. 1 Sample augmented images (from left to right): original image; pixel-level annotation, image with Gaussian-distributed additive noise; Gaussian blurred image; color manipulated image; elastic transformed image, image with new background, cropped and flipped image with new background

graph theory to group the detected line segments into whole power lines [36].

Similar to line-based methods, piece-wise line segment-based methods also often mistakenly detect linear objects with similar line features in the background, such as metallic fence lines and building edges, as power lines [36].

2.3 Auxiliaries-assisted methods

To address the existing problems of line-based and piece-wise line segment-based methods, much effort has been made toward utilizing correlation information and context features provided by auxiliaries. For example, Zhang et al. proposed to use the spatial correlation between the pylon and the power line to improve transmission line detection performance [45]. The proposed method outperforms line-based and piece-wise line segment-based methods; however, the performance drops significantly when the pylon is absent or occluded.

To eliminate the need for manually selecting auxiliaries and defining spatial relationships between auxiliaries and power lines, Shan et at. proposed an optimization-based approach for automatic auxiliaries selection and contexts acquisition [31]. The proposed approach surpasses traditional methods that use manually assigned auxiliaries both in terms of detection accuracy and false alarm probability; however, it is quite slow due to the sliding window-based object extraction and the context representation between the auxiliaries and the hypotheses.

To further improve auxiliaries-assisted power line detection accuracy and speed, Pan et al. proposed a metric for measuring the usefulness of an auxiliary in assisting power line detection, named spacial context disparity, based on two factors: spatial context peakedness and spatial context difference and applied it for automatic selection of optimal auxiliaries [29]. According to the authors, the proposed method is robust and can achieve satisfactory performance for power line detection.

2.3.1 DL-based methods

One of the earliest attempts to use deep learning for power line detection was the work of Jayavardhana et al. [14]. The authors proposed a CNN-based classifier that uses histogram of gradient (HoG) features as the input and applied it in a sliding window fashion to classify patches of size 32×32 into two classes: "Line present" and "No line present." The authors also fine-tuned the GoogleNet on patches of original images for the same task. According to the authors, the proposed CNN-based classifier achieves an F-score of 84.6% and outperforms the GoogleNet, which achieves an F-score of 81%.

Ratnesh et al. [25] treated wire detection as a semantic segmentation task and performed a grid search over a finite space of CNN architectures to find an optimal model for the task based on dilated convolutional networks [44]. The model was trained on synthetic images of wires generated by a ray-tracing engine and fine-tuned on real images of wires from the USF dataset [16]. According to the authors, the proposed model outperforms the previous work that uses traditional computer vision and various CNN-based baselines such as FCNs, SegNet, and E-Net; the model achieves an average

precision (AP) score of 0.73 on the USF dataset and runs at more than 3Hz on the NVIDIA Jetson TX2 with input resolution of 480 × 640.

Although treating wire detection as a semantic segmentation task has been proved to be a powerful approach for detecting wires [25], its requirement of pixel-level annotated ground-truth data makes it less practical than traditional computer vision approaches. Sang et al. proposed to use weekly supervised learning with CNNs for localizing power lines in pixel-level precision by only using image-level class information [20]. First, a classifier adapted from the VGG19 is applied to classify subregions (128 × 128) from an input image (512×512) by using a sliding window approach. Then, feature maps of intermediate convolutional layers of subregions that are classified as "sub-region with power lines" are combined to visualize the location of the power lines. Although the localization accuracy of the proposed approach is still far from an applicable level of industrial fields, it can be applied, according to the authors, to generate ground-truth data in pixel-level roughly.

Yan et al. proposed a power detection pipeline based on pyramidal patch classification in [21]. First, input images are hierarchically partitioned into patches. Next, a CNN classifier is trained to classify the patches into two classes: patches with power lines and patches without power lines. Then, the classified patches are used as inputs for edge feature extraction using steerable filters and line segment detection using the progressive probabilistic Hough transform (PPHT). Finally, the detected line segments are connected using a power line segments correlation module to form complete power lines. The authors concluded that the proposed approach significantly improves the detection rate of the power line detection and largely decreases the false alarm rate.

3 The line segment detector (LS-Net)

3.1 Data generation

3.1.1 Synthetic data generation

Due to the unavailability of large datasets with annotations of power lines, we collaborate with Nordic Media Lab (NMLab)[1] to render synthetic images of power lines using the physically based rendering (PBR) approach [18]. First, we model aluminum conductor steel-reinforced (ACSR) cables, which are typically composed of one steel center strand and concentric layers of high-purity aluminum outer strands, using the Autodesk 3DS Max program. To increase the realistic appearance of the cables, we utilize the bevel and the twist modifiers together with the metal brushed steel texture. Then, we randomly superimpose the cables on 71 8K high dynamic range images (HDRIs) collected from the Internet.[2] Next, to further increase the realistic appearance of the cables, we employ cube mapping to capture the reflection and the lighting data from the HDRIs and apply them to the cables. Finally, we apply a series of effective variations, with respect to the camera angle, the camera distance, out-of-focus blur, cable colors, the number of cables, and the distance between cables, to render more synthetic images.

3.1.2 Data augmentation

Inspired by the success of data augmentation for improving the performance of CNNs in [40] and [33], we propose a series of effective data augmentation techniques to generate more training data by applying transformations in the data-space. These are all implemented using the scikit-image [38] and the OpenCV libraries [4].

The first technique replaces the background of the generated synthetic images with real background images to increase the diversity of the dataset and to account for various types of background variations during the inspection (e.g., different seasons, weather conditions, and lighting conditions).

The second technique adds Gaussian-distributed additive noise to account for noisy image acquisition (e.g., sensor noise caused by poor illumination and/or high temperature, and/or transmission) [3]. The augmented image $f(i, j)$ is the sum of the true image $s(i, j)$ and the noise $n(i, j)$:

$$f(i, j) = s(i, j) + n(i, j). \tag{1}$$

The noise term, $n(i, j)$, follows a Gaussian random distribution:

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}, \tag{2}$$

where z represents the gray level, μ is the mean value, and σ is the standard deviation.

To account for possible out-of-focus, Gaussian blur is employed by convolving the image with a two-dimensional Gaussian function:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{3}$$

where x and y are distances from the origin in the horizontal axis and the vertical axis, respectively, and σ is the standard deviation [32].

[1] http://nmlab.no/.

[2] https://hdrihaven.com/.

To introduce invariance to changes in lighting and to capture minor color variations, especially in power lines, a series of color manipulations including random brightness, random saturation, random contrast, and random hue are utilized. In addition, to further extend color invariance, we randomly remove colors from RGB images by first converting them to grayscale and then converting the grayscale images back to RGB.

With the aim of training models that can detect not only perfectly straight line segments but also curvy ones, elastic deformations [33] are employed. First, two random displacement fields for the x-axis (Δx) and y-axis (Δy) are generated as follows:

$$\Delta x(x, y) = rand(-1, +1), \tag{4}$$
$$\Delta y(x, y) = rand(-1, +1), \tag{5}$$

where $rand(-1, 1)$ is a random number between -1 and $+1$, generated with a uniform distribution. Next, the fields Δx and Δy are convolved with a two-dimensional Gaussian function similar as shown in Eq. (3) to form elastic deformation fields. Then, the elastic deformation fields are scaled by factor α that controls the intensity of the deformation. Finally, the fields Δx and Δy are applied to images.

To account for various camera distances and viewing angles, zoom and rotation operators are employed [39]. The zoom operator is applied by randomly cropping images and scaling them to their original size. The rotation operator is employed by multiplying images with a rotation matrix R:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

where θ is the rotation angle. The final technique flips the images horizontally and vertically (Fig. 1).

3.2 LS-Net architecture

Inspired by the success of single-shot object detectors such as SSD [24] and YOLO [30] in terms of speed and accuracy, we propose a single-shot line segment detector, named LS-Net. The LS-Net is based on a feed-forward, fully convolutional neural network and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor connected as shown in Fig. 2.

The design of the LS-Net architecture is mainly inspired by state-of-the-art single-shot object detectors such as SSD [24] and YOLO [30]. Specifically, the LS-Net divides the input image of size $W \times H \times C$ into a grid, and each grid cell of size $C \times C$ predicts coordinates and a confidence score for the longest line segment in the cell. The confidence score indicates the probability of the cell containing a line segment, and the coordinates are the normalized distances of the two endpoints of the line segment to the local x-axis and y-axis of the cell.

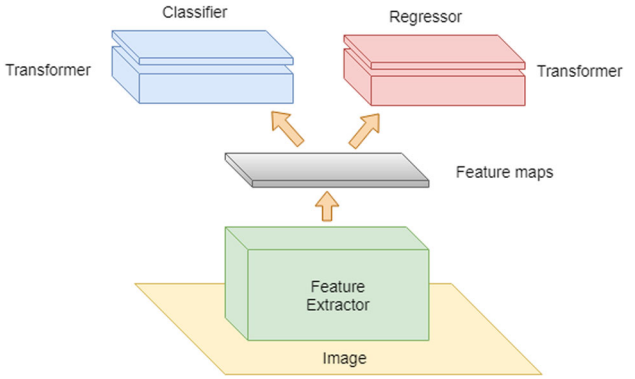

Fig. 2 LS-Net is a feed-forward, fully convolutional neural network and consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor

The traditional one-grid approach has been proven to work well for single-shot object detectors such as SSD [24] and YOLO [30]; however, it faces two problems when applied to line segment detection: (i) discontinuities and gaps at cell borders, and (ii) discontinuities and gaps at cell corners. In the one-grid approach, due to regression errors, the detected line segments can be shorter than the ground truths. This can result in discontinuities and gaps in the detected lines at borders of adjacent cells that make regression errors. In addition, the one- grid approach ignores short line segments, especially at cell corners, due to the lack of features. This can also lead to discontinuities and gaps in the detected lines (see Fig. 3 and Fig. 4).

To address the two above-mentioned problems, we propose to replace the one-grid approach by a four-grid approach. Specifically, the four-grid LS-Net divides the input image into four overlapping grids: a $S_m \times S_m$ grid (main grid), a $S_m \times S_a$ grid (horizontal grid), a $S_a \times S_m$ grid (vertical grid), and a $S_a \times S_a$ grid (center grid), where $S_a = S_m - 1$ (see Fig. 5). The main grid, which works exactly the same as the grid used by SSD and YOLO for detecting objects, is employed for detecting line segments in grid cells. The horizontal and vertical grids are utilized for closing the gaps at horizontal and vertical borders, respectively. The central grid is used for detecting short line segments at cell corners that were ignored by the main grid. All the detected line segments from the four grids are combined together to form a line segment map. Since the four-grid LS-Net utilizes three additional grids to detect short line segments ignored by the main grid and close gaps at horizontal and vertical borders, the discontinuities in the detected lines are significantly eliminated (see Fig. 5).

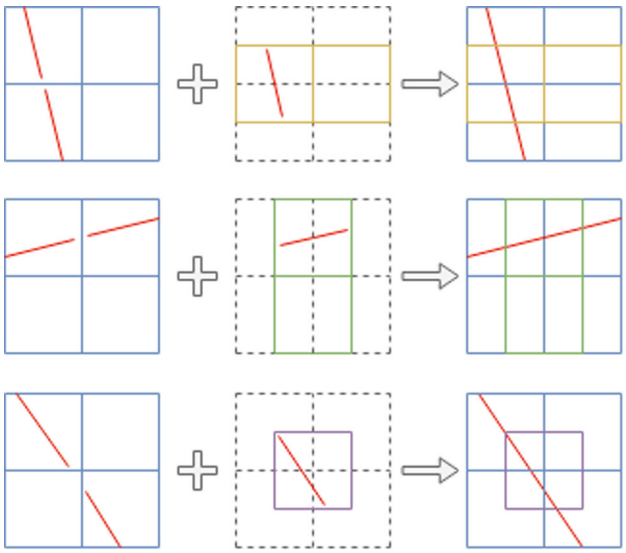

Fig. 3 Illustration of the four-grid approach. LS-Net with the traditional one-grid approach (the first column) ignores short line segments at cell corners and create gaps at cell borders in the detected lines. LS-Net with the four-grid approach (the third column) utilizes three additional grids (the second column) to detect line segments ignored by the main grid and close gaps at horizontal and vertical borders, which significantly eliminate the discontinuities in the detected lines

3.2.1 Fully convolutional feature extractor

The LS-Net feature extractor is inspired by the VGG-16 network [34]. We truncate the network before the last max-pooling layer and substitute the remaining max-pooling layers by strided convolutional layers with stride 2. Max-pooling layers have been used extensively in CNNs for image classification; however, they are not an optimal choice for the proposed LS-Net since they throw away spatial information that is useful for predicting line segment end-points.

With the aim of easing the optimization, enabling the network to converge faster, and eliminating the dependence on batch sizes, we adopt Group Normalization [41] before activations in every convolutional layer.

3.2.2 Classifier

The classifier sub-network takes feature maps extracted by the fully convolutional feature extractor as input and predicts whether each grid cell contains a line segment or not. The sub-network consists of two layers: The first is a 2×2 convolutional layer with stride 1 that works as a *transformer (transformation layer)* and transforms the input feature maps into four sets of feature maps corresponding to the four overlapping grids. The second layer is a 1×1 convolutional layer that predicts a confidence score for each grid cell.

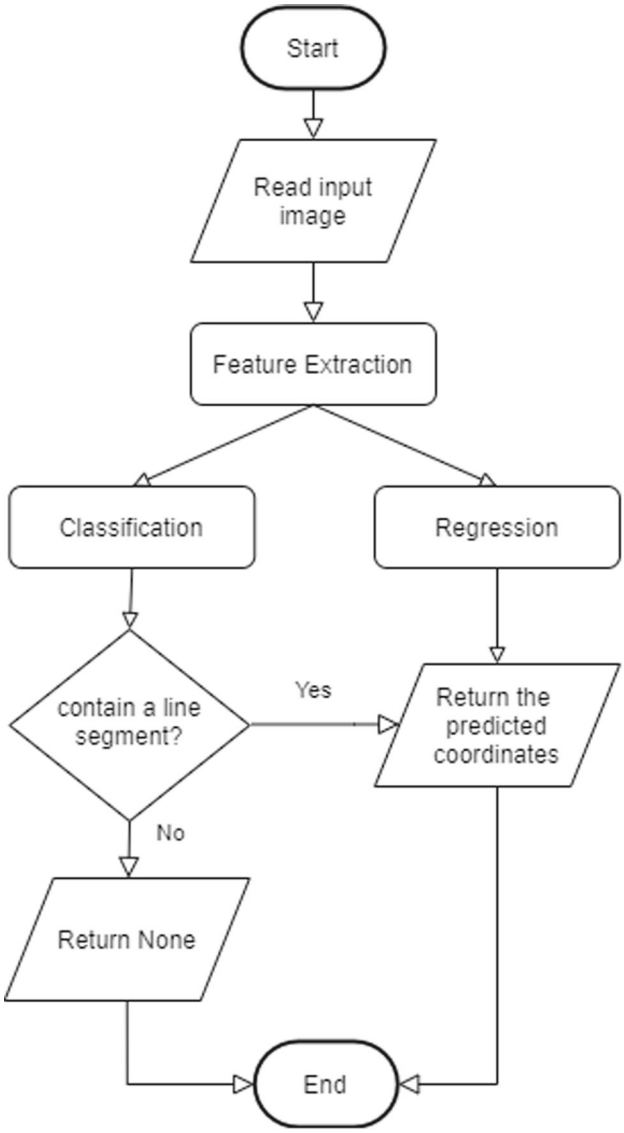

Fig. 4 LS-Net's flowchart. The flowchart shows the process by which a line segment is predicted in a grid cell. Since the LS-Net is a single-shot detector, it predicts line segments for all the grid cells in a single forward pass

3.2.3 Line segment regressor

The line segment regressor sub-network takes feature maps extracted by the fully convolutional feature extractor as input and predict coordinates of the longest line segment in each grid cell. The sub-network also consists of two layers: The first layer is similar to the first layer of the classifier sub-network. The second is a 1×1 convolutional layer that is responsible for predicting line segment coordinates.

3.2.4 Summary

With the four-grid approach, the output of the LS-Net is very similar to that of a traditional sliding-window detector of size $C \times C$ with stride $C/2$; however, the LS-Net has to major advantages over the sliding-window approach: The first is that instead of applying a costly forward pass hundreds of times, one for each cell, the LS-Net makes predictions for all cells in a single forward pass, which was made possible thanks to the single-shot detector architecture and the combination of our proposed four-grid approach and our proposed transformation layers. The second advantage is that the LS-Net, with a large effective receptive field, can take into account contextual information when making predictions. In other words, the LS-Net looks at not only the target cell but also its neighboring cells to make predictions for the cell.

To evaluate the effectiveness of the proposed LS-Net architecture, we train the LS-Net on input images of size $512 \times 512 \times 3$ to detect line segments in cells of size 32×32, i.e., $S_m = 16$; however, the proposed LS-Net architecture can be easily generalized to handle images of any sizes and to detect line segments in cells of any sizes. A detailed configuration of the LS-Net used in our experiments in this paper is shown in Table 1. All convolutional layers in the feature extractor are padded so that they produce an output of the same size as the input. Padding is not applied in convolutional layers in the classifier and the regressor.

Table 1 LS-Net's Configuration. The convolutional layer parameters are denoted as "conv(receptive field size)-(number of channels)[-S(stride)]". The default stride is 1

Input image ($512 \times 512 \times 3$)	
Conv3-64	
Conv3-64	
Conv3-64-S2	
Conv3-128	
Conv3-128	
Conv3-128-S2	
Conv3-256	
Conv3-256	
Conv3-256-S2	
Conv3-512	
Conv3-512	
Conv3-512-S2	
Conv2-512	Conv2-512
Conv1-2	Conv1-4

3.3 LS-Net multitask loss

The LS-Net has two sibling output layers. The first sibling layer outputs a discrete probability distribution, $p_t^i = (p^i, 1-p^i)$, for each grid cell, indexed by i, over two classes: *cell with line segments* and *cell without line segments*. The probability distribution p_t^i is computed by a softmax over the two outputs of a 1×1 convolution layer at the i^{th} cell. The second sibling layer outputs coordinates of the two end-points of the longest line segment, $e^i = (e^i_{x1}, e^i_{y1}, e^i_{x2}, e^i_{y2})$, for each grid cell, indexed by i.

Each training cell is labeled with a ground-truth class label $y^i \in \{\pm 1\}$ and a ground-truth end-point regression target $t^i = (t^i_{x1}, t^i_{y1}, t^i_{x2}, t^i_{y2})$. We use a weighted multitask loss function, L, to jointly train for cell classification and line segment end-point regression:

$$L(p_t, y, e, t) = L_{cls}(p_t, y) + \lambda[y = 1]L_{reg}(e, t), \tag{6}$$

where the Iverson bracket indicator function $[y = 1]$ evaluates to 1 when $y = 1$ and 0 otherwise.

The first task loss, L_{cls}, is a Focal loss [13] defined as follows:

$$L_{cls}(p_t, y) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t), \tag{7}$$

where $\gamma \geq 0$ is a tunable focusing parameter, $\alpha_t \in [0, 1]$ is a weighting factor defined as follows:

$$\alpha_t = \begin{cases} \alpha & \text{if } y = 1 \\ 1 - \alpha & \text{otherwise} \end{cases}, \tag{8}$$

and $p_t \in [0, 1]$ is the model's estimated probability defined as follows:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases}. \tag{9}$$

Since the number of cells without line segments is much larger than the number of cells with line segments, the Focal loss is employed instead of a standard Cross-Entropy loss [12] to address the class imbalance during training.

The second task loss, L_{reg}, is a Wing loss [8] defined as follows:

$$L_{reg}(e, t) = \begin{cases} w \ln(1 + d/\epsilon) & \text{if } d < w \\ d - C & \text{otherwise} \end{cases}, \tag{10}$$

where w is a nonnegative upper bound that sets the range of the nonlinear part to $(-w, w)$, ϵ is a constant that limits the curvature of the nonlinear region, $C = w - w \ln(1 + w/\epsilon)$ is a constant that smoothly links the piecewise-defined linear and nonlinear parts, and d is our proposed error function,

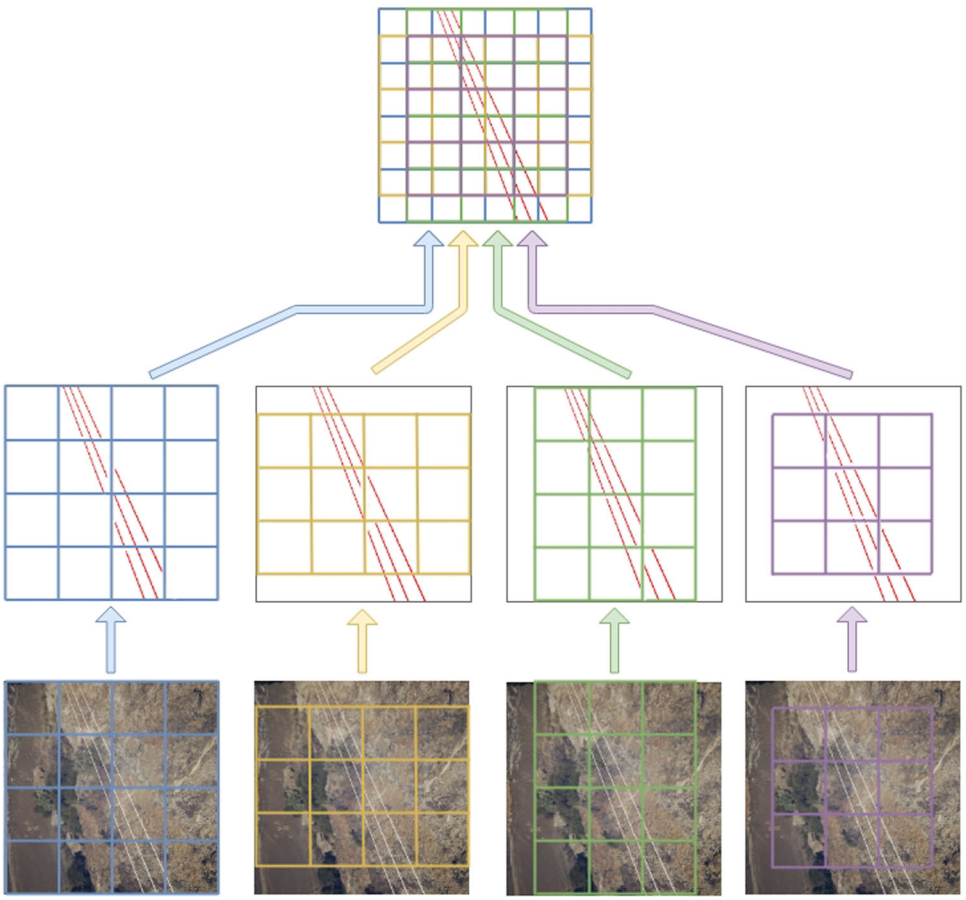

Fig. 5 Illustration of the four overlapping grids approach. LS-Net with one grid (the leftmost branch) ignores short line segments at cell corners and creates gaps at cell borders in the detected lines. LS-Net with four overlapping-grid approach utilizes three additional grids to detect line segments ignored by the first grid and close gaps in horizontal and vertical lines, which significantly eliminate the discontinuities in the detected lines

which computes the minimum absolute difference between the predicted end-points $e = (e_{x1}, e_{y1}, e_{x2}, e_{y2})$ and the target end-points $t = (t_{x1}, t_{y1}, t_{x2}, t_{y2})$ defined as follows:

$$d(e,t) = \min\left(\sum(|t-e|), \sum(|t-swap(e)|)\right), \qquad (11)$$

where $swap(e) = (e_{x2}, e_{y2}, e_{x1}, e_{y1})$ is a function that swaps the order of the two end-points.

The error function d is employed to allow the LS-Net to predicts the two end-points of a line segment regardless of the order, and the Wing loss is utilized instead of standard L_2 [11] or smooth L_1 [10] losses to restore the balance between the influence of errors of different sizes and to allow the model to regress the line segment end-points more accurately.

3.4 Training and testing

The LS-Net can be trained end-to-end by backpropagation and stochastic gradient descent (SGD) [19]. We implement the LS-Net using the Tensorflow framework [1]. We train the LS-Net from scratch using the Adam optimizer [17] with initial learning rate 0.0001, 0.9 momentum1, 0.999 momentum2, and batch size 8 (due to memory limitation) on a TITAN X (Pascal) GPU. We use early stopping to prevent the network from overfitting. Our network converges after 3.5 epochs, which takes around 48 hours of training time.

Before training, we augment our dataset by generating five random crops and their flipped versions from each image; we further augment the dataset by replacing the background from each image with five randomly selected backgrounds from our background image dataset.

During training, we apply data augmentation on-the-fly by adding Gaussian-distributed additive noise, by applying Gaussian blur, by performing a series of color manipulations, and by employing elastic deformations. All the on-the-fly data augmentation techniques are applied with a probability of α. We use $\alpha = 0.25$ in our experiments.

For $512 \times 512 \times 3$ input, the LS-Net runs at 21.5 Frames Per Second (FPS) on a TITAN X (Pascal) GPU at test time. However, the speed can be further increased by employing a shallower, thinner feature extractor and by decreasing the input size.

4 Experiments

4.1 Comparisons with the state-of-the-art results

As presented in Sect. 2, there are very few relevant DL-based approaches for power line detection. In addition, two approaches among the four reviewed ones apply deep learning for patch classification only, while the line detection step is still addressed by a traditional line detection or line segment detection algorithm such as the progressive probabilistic Hough transform (PPHT) [9] or the line segment detector (LSD) [14]. This typically results in low analysis speed and a need for post-processing to distinguish between power lines and spurious lines. Since our goal is to facilitate real-time power line detection and avoidance in low-altitude UAV flights with deep learning, in this section, we compare our proposed LS-Net only to state-of-the-art DL-based approaches for power line detection that offer high analysis speed and require minimal effort in post-processing in terms of FPS, pixel-level Averaged Recall Rate (ARR), pixel-level averaged precision rate (APR), and pixel-level averaged F_1 Scores:

$$ARR = \frac{1}{N}\sum_{i=1}^{N} R_i = \frac{1}{N}\sum_{i=1}^{N} \frac{TP_i}{TP_i + FN_i}, \tag{12}$$

$$APR = \frac{1}{N}\sum_{i=1}^{N} P_i = \frac{1}{N}\sum_{i=1}^{N} \frac{TP_i}{TP_i + FP_i}, \tag{13}$$

$$F_1\ Scores = \frac{1}{N}\sum_{i=1}^{N} 2 \times \frac{P_i \times R_i}{P_i + R_i}, \tag{14}$$

where R_i, P_i, TP_i, FN_i, FP_i are pixel-level recall, pixel-level precision, number of true positive pixels, number of false negative pixels, and number of false positive pixels of the ith image, respectively, and N is the number of test images.

First, we compare our proposed LS-Net with the weakly supervised learning with CNNs (WSL-CNN) approach proposed in [20] on the publicly available ground truth of power line dataset (Infrared-IR and Visible Light-VL) [43], which is one of the most widely used power line datasets. The LS-Net and the WSL-CNN approaches share a similar objective that is to localize power lines by using cheaper ground-truth data (GTD) than pixel-level GTD (e.g., image-level class information and line end-point information). For a fair comparison, we convert line segment maps generated by the LS-Net to pixel-level segmentation maps using a similar procedure as applied in [20]. First, the pixel-level segmentation maps, S, are generated as follows:

$$conf(x, y) = max(\{conf(LS_i) \mid (x, y) \in LS_i\}), \tag{15}$$

$$S(x, y) = \begin{cases} 0 & \text{if } (x, y) \notin LS_i\ \forall i \in [1, L] \\ conf(x, y) & \text{otherwise} \end{cases}, \tag{16}$$

where L is the number of detected line segments, LS_i is the list of all pixels belonging to the i^{th} line segment, and $conf(LS_i)$ is a function that returns the confidence score of the i^{th} line segment. Since each line segment predicted by the LS-Net is represented by a pair of two end-points, we apply the 8-connected Bresenham algorithm [5] to form a close approximation to a straight line between the two end-points. We vary the width of the straight line, W_l, from 1 to 5 and select $W_l = 2$ and $W_l = 3$ since they result in the highest F_1 scores. We call these models LS-Net-W2 and LS-Net-W3, respectively. Then, the generated segmentation maps are smoothed by convolving with a two-dimensional Gaussian function, as shown in Eq. (3). Finally, the predicted segmentation maps are binarized by using the Otsu's method [15,28].

Then, we implement the dilated convolution networks for wire detection (WD-DCNN) proposed in [25] in Tensorflow. In addition, we improve the WD-DCNN approach by adopting Group Normalization [41] to accelerate the training of the networks and Focal loss [13] for restoring the balance between the influence of errors of different sizes in multiple points regression. We create three improved models. In the first model, we add a group normalization layer after each convolutional layer in the WD-DCNN model (WD-DCNN-GN). We replace the class-balanced Cross-Entropy loss function [44], adopted by the WD-DCNN model, by the Focal loss to train the second model (WD-DCNN-FL). Finally, we combine both Group Normalization and Focal loss to train the third model (WD-DCNN-GNFL). We train the WD-DCNN model and its improved versions on the same training dataset that we use to train our proposed LS-Net. The predicted segmentation maps of the four models are binarized by using the Otsu's method [15,28].

Finally, for the sake of completeness, we also compare the LS-Net to one of the most well-known traditional approach for line detection, the PPHT [9]. We apply the same procedure described above to convert the PPHT's detection results to segmentation maps. The test results are shown in Table 2

As can be seen from Table 2, both our proposed LS-Net-W2 and LS-Net-W3 models achieve state-of-the-art performance in terms of F_1 score. In addition, the LS-Net-W2 model surpasses all the existing state-of-the-art methods in terms of APR, while the LS-Net-W3 model attains state-of-the-art ARR by considerable margins. Visual comparisons of the LS-Net-W2 model, the PPHT, and the state-of-the-art DL-based approaches for power line detection including the WD-DCNN, the WD-DCNN-GN, and its improved versions are shown in Fig. 6.

Table 2 Detection results on the ground truth of power line dataset (Infrared-IR and Visible Light-VL). FPS was measured on a Titan X (Pascal) GPU. *Results reported by [20]

	ARR	APR	F_1 Score	FPS
WSL-CNN [20]*	0.6256	-	-	-
PPHT [9]	0.8409	0.3217	0.4392	8.6
WD-DCNN [25]	0.7192	0.4713	0.4835	**36.5**
WD-DCNN-GN	0.8292	0.4148	0.4882	19.8
WD-DCNN-FL	0.7514	0.4680	0.5079	**36.5**
WD-DCNN-GNFL	0.7930	0.4690	0.5218	19.8
LS-Net-W2	0.7972	**0.4874**	**0.5344**	21.5
LS-Net-W3	**0.8525**	0.4483	**0.5256**	21.5

Bold values highlight the best methods and results

The results show that even though the LS-Net is trained for localizing the power lines instead of detecting power line pixels, it still outperforms methods that are trained explicitly for that task. To compare with these methods, we convert the detected line segments to pixel-level segmentation maps by simply drawing a line between the two end-points of every predicted line segment. We believe that a more sophisticated method for converting line segments to pixel-level segmentation maps would increase the ARR, APR, and F_1 Scores of the LS-Net further. However, since our main goal is to localize the power lines instead of detecting the power line pixels. We employ a very simple conversion method to highlight the LS-Net's advances in power line localization. More test results of the LS-Net on real images are shown in Fig. 7.

4.2 Ablation study

To investigate the effectiveness of the proposed LS-Net architecture and the loss function, we conducted several ablation studies using the publicly available ground truth of power line dataset (Infrared-IR and Visible Light-VL) [43]. We use the approach presented in Sect. 4.1 to convert line segment maps generated by the LS-Net to pixel-level segmentation maps and compare different variants of the LS-Net in terms of APR, ARR, and F_1 Score. To increase the interpretability of the comparison results, we apply a simple thresholding method ($t = 0.5$) to binarize segmentation maps instead of the Otsu method and set the width of the line segment W_l to 1 when applying the 8-connected Bresenham algorithm. This could result in lower APR, ARR, and F_1 score; however, it is not an issue since improving the performance of the LS-Net is not the primary goal of the ablation studies.

First, we evaluate the effects of replacing max-pooling layers by strided convolution layers. To do this, we compare the proposed LS-Net with strided convolutional layers (LS-Net-S) with an LS-Net with max-pooling layers (LS-Net-P), which is constructed by replacing each stride-2 convolutional layer in the LS-Net-S by a stride-1 convolutional layer followed by a max-pooling layer. The comparisons between the LS-Net-S' and the LS-Net-P' performances and losses are

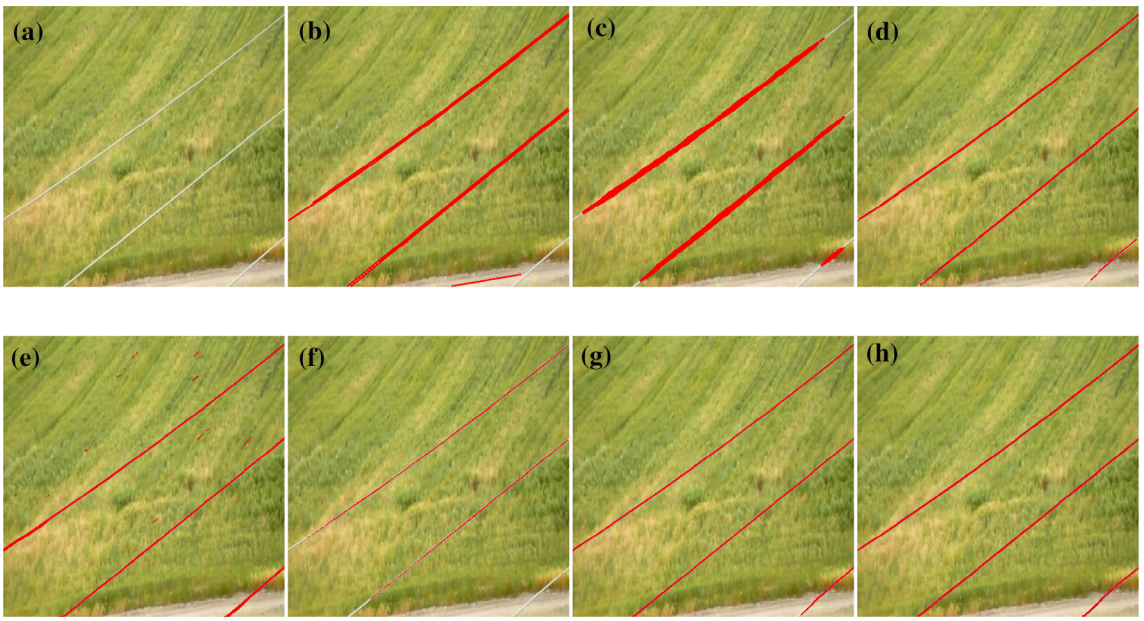

Fig. 6 Visual comparisons of the LS-Net-W2 model and the state-of-the-art methods. From left to right, top to bottom are, respectively, **a** the original image, **b** the PHT's detection results, **c** the WSL-CNN's detection results, **d** the WD-DCNN's detection results, **e** the WD-DCNN-GN's detection results, **f** the WD-DCNN-FL's detection results, **g** the WD-DCNN-GNFL's detection results, and **h** the LS-Net's detection results

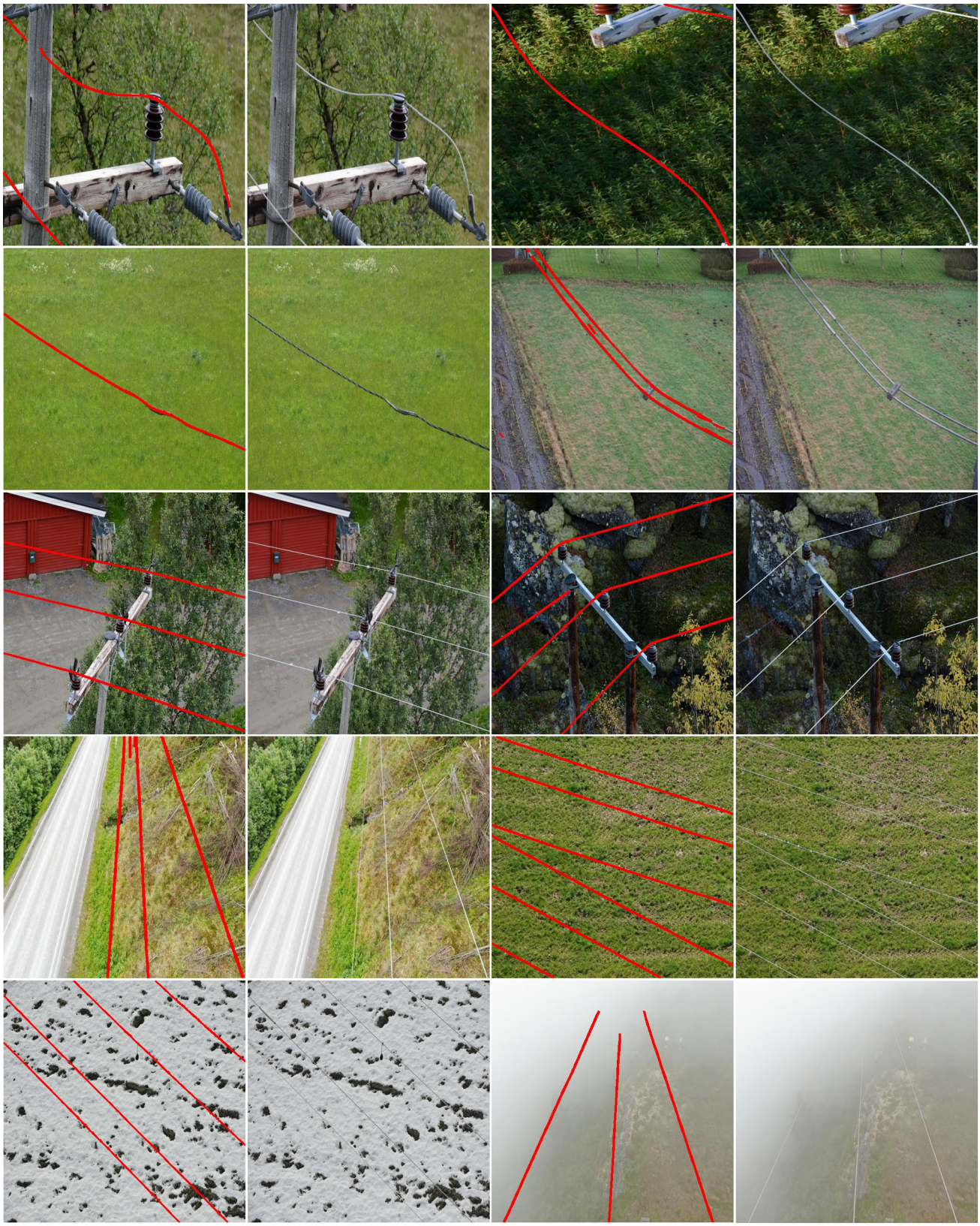

Fig. 7 Test results of the LS-Net on real images. The first and the third columns show the outputs from the LS-Net while the second and the fourth columns show the original images

Table 3 Comparisons between the LS-Net with strided convolutional layers (LS-Net-S) and the LS-Net with max-pooling layers (LS-Net-P)

Method	APR	ARR	F_1 Score
LS-Net-P	0.7828	**0.5378**	0.5885
LS-Net-S	**0.8004**	0.5368	**0.5940**

Bold values highlight the best methods and results

shown in Table 3. As can be seen from Table 3, the LS-Net-S architecture outperforms the LS-Net-P architecture in terms of APR and F_1 Score.

We observe that both LS-Net-S and LS-Net-P perform similarly on the classification sub-task; however, the LS-Net-S outperforms the LS-Net-P on the line segment regression sub-task (see Fig. 8). This indicates that strided convolution is a more suitable choice for our proposed LS-Net architecture than the standard max pooling.

Then, we investigate the impact of the four-grid approach. We compare against LS-Net with one, two, three, and four grids, respectively. Table 4 shows that as the number of grids increases, APR decreases slightly, but ARR increases dramatically. This results in an increase of F_1 score as the number of grids increases. Since LS-Net with more grids makes more predictions then LS-Net with fewer grids, their APRs are slightly lower than that of LS-Net with fewer grids. However, as the additional grids detect short line segments ignored by the main grid at cell corners and close gaps at horizontal and vertical borders, the ARR of LS-Net with more girds is significantly higher than that of LS-Net with fewer grids. As can be seen from Table 4 and Fig. 9, the LS-Net with four grids outperforms LS-Net with one, two, and three grids in terms of ARR and F_1 score and significantly eliminates the discontinuities in the detected lines. This suggests that our proposed four-grid approach is more suited for our proposed LS-Net architecture.

Next, we show the effects of the Wing loss on line segment regression performance. We compare against LS-Net trained with Wing loss (LS-Net-W) and its variants: LS-Net trained with L2 loss (LS-Net-2), L1 loss (LS-Net-1), and Smooth L1 loss (LS-Net-S) [10], respectively. Table 5 shows that LS-Net trained with Wing loss outperforms its variants in terms of APR; however, it performs worse in terms of ARR. Wing loss biases the optimizer toward minimizing small regression errors at the end of the training by increasing the gradient, given by $1/x$, as the errors approach zero error. This results in lower regression errors that lead to a significantly higher APR compared to the LS-Net-1, LS-Net-2, and LS-Net-S. However, this causes the classification errors to increase as we use a fixed weight in the multitask loss (see Eq. (6)). This leads to a slightly lower ARR compared to the LS-Net-1, LS-Net-2, and LS-Net-S. Since the increase in APR is much more than the decrease in ARR, the F_1 score of LS-Net trained with Wing loss is higher than LS-Net trained with the standard regression losses such as L2, L1, and Smooth L1. This indi-

Table 4 Performance of LS-Net with the one, two, three, and four grids, respectively. The methods are denoted as "'LS-Net-⟨number of grids⟩-⟨grids⟩". M, H, V, C represent main, horizontal, vertical, and central grids, respectively

Method	APR	ARR	F1 Score
1 Grid			
LS-Net-1-M	**0.8312**	0.3791	0.4847
2 Grids			
LS-Net-2-MH	0.8174	0.4717	0.5540
LS-Net-2-MV	0.8173	0.4703	0.5533
LS-Net-2-MC	0.8165	0.4776	0.5574
3 Grids			
LS-Net-3-MHC	0.8080	0.5121	0.5792
LS-Net-3-MVC	0.8080	0.5117	0.5791
LS-Net-3-MVH	0.8064	0.5165	0.5826
4 Grids			
LS-Net-4-MHVC	0.8004	**0.5368**	**0.5940**

Bold values highlight the best methods and results

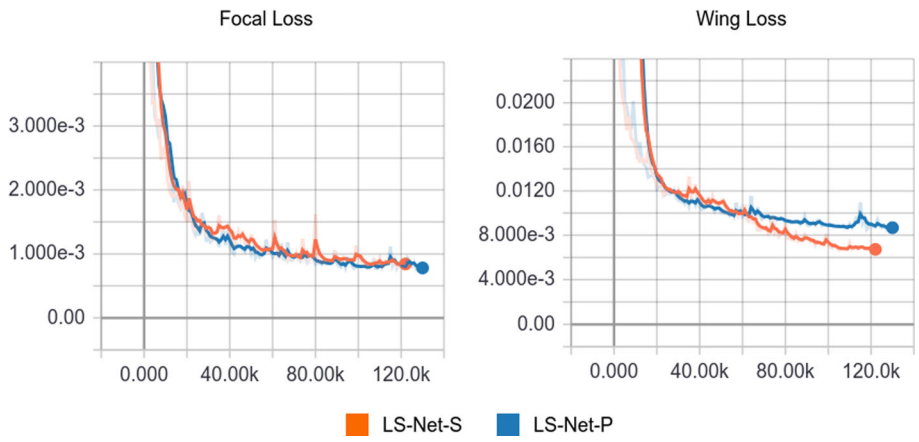

Fig. 8 Comparisons between LS-Net-S' and LS-Net-P' test losses. LS-Net-S and LS-Net-P perform similarly on the classification sub-task (Focal loss); however, the LS-Net-S outperforms the LS-Net-P on the line segment regression sub-task (Wing loss)

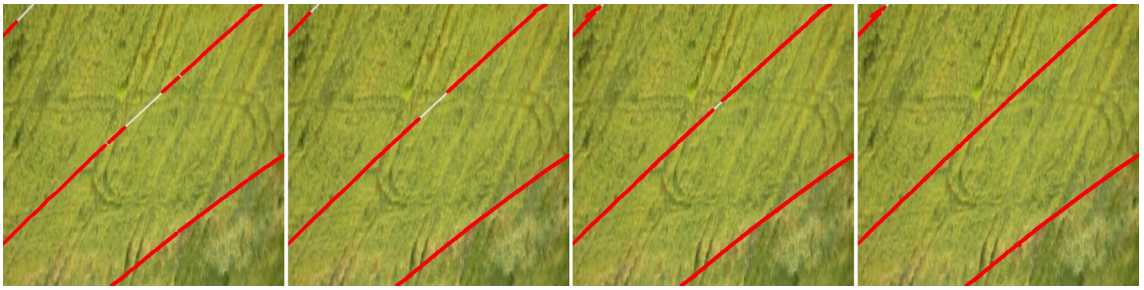

Fig. 9 Test results of the LS-Net with (from left to right) one, two, three, and four grids, respectively. (The width of the line segments is increased to 5 pixels for better visualizations.) The one-grid LS-Net approach (the leftmost image) ignores short line segments at cell corners and leaves gaps at cell borders in the detected lines. The four-grid LS-Net approach (the rightmost image) detects line segments ignored by the first grid and close gaps in horizontal and vertical lines, which significantly eliminate the discontinuities in the detected lines

Table 5 Performance of the LS-Net trained with Wing loss (LS-Net-W), L2 loss (LS-Net-2), L1 loss (LS-Net-1), and L1 smooth loss (LS-Net-S)

Method	APR	ARR	F1 Score
LS-Net-2 (L2)	0.7277	**0.5789**	0.5866
LS-Net-1 (L1)	0.7032	0.5694	0.5765
LS-Net-S (Smooth L1)	0.7317	0.5495	0.5728
LS-Net-W (Wing loss)	**0.8004**	0.5368	**0.5940**

Bold values highlight the best methods and results

Table 6 Comparisons between LS-Net trained with Focal Loss (LS-Net-FL) and LS-Net trained with standard Cross-Entropy loss (LS-Net-CE)

Method	APR	ARR	F1 Score
LS-Net-CE	0.7946	0.5353	0.5899
LS-Net-FL	**0.8004**	**0.5368**	**0.5940**

Bold values highlight the best methods and results

cates that the Wing loss is a more suitable choice for training the line segment regressor in our proposed LS-Net architecture; however, an adaptive weighting approach is needed for balancing the training of the line segment regressor and the cell classifier. We leave this for future work.

Finally, we evaluate the effect of the Focal loss on cell classification performance. We compare between LS-Net trained with Focal loss (LS-Net-FL) and LS-Net trained with standard Cross-Entropy loss (LS-Net-CE). As can be seen from Table 6, LS-Net trained with Focal loss outperforms LS-Net trained with standard Cross-Entropy loss in terms of APR, ARR, and F_1 score. This indicates that the Focal loss is a more suitable choice for training the cell classifier in our proposed LS-Net architecture than the standard Cross-Entropy loss.

5 Discussion

With the ability to detect power line segments in near real-time (21.5 FPS), the LS-Net shows the potential to facilitate real-time power line detection and avoidance in low-altitude UAV flights to ensure flight safety. During UAV flights, power line segment maps produced by the LS-Net can be employed to detect power lines and identify dangerous zones quickly, and these information sources can be used as additional inputs to improve the performance of obstacle avoidance and path recovery algorithms.

In addition, the LS-Net can be utilized for vision-based UAV navigation and for vision-based inspection of power lines. In automatic autonomous power line inspection, the UAV needs to flight along the power lines to take pictures for offline inspections and performs online inspection to identify faults on the power lines (e.g., corroded and damaged power lines) and surrounding objects, such as vegetation encroachment. When GPS-based navigation is not possible, power line segment maps produced by the LS-Net can be employed to navigate the UAV along the power lines. Besides, the power line segment maps can be used for steering the cameras mounted on the UAV to take higher quality pictures of the power lines to improve the performance and reduce the costs of both online and offline inspections.

Since the LS-Net can be trained end-to-end and performs very well even when trained only on synthetic images, it can potentially be adapted for detecting other linear structures. One example is railway track detection. In recent years, the need for automatic vision-based inspection of railway tracks using UAVs has been increasing since UAVs do not require separate tracks for data acquisition as in traditional inspection methods [35]. Similar to power line inspection, the LS-Net can be potentially applied for detecting railway tracks from images taken from UAVs. These detections can be utilized both for navigating the UAVs along the railway tracks and for steering the cameras mounted on the UAVs to take pictures of the railway tracks for offline inspections. Another

example is unburied onshore pipeline detection in automatic UAV-based gas leak inspection [2]. Since the width of gas pipelines is relatively big in images taken from UAVs, the LS-Net cannot be applied directly to detect gas pipelines. However, this problem can potentially be addressed by casting the gas pipeline detection as a gas pipeline edge detection problem. The LS-Net can be applied for detecting the edges of gas pipelines. The edge detection results can be used for navigating the UAVs along the pipelines, for steering other sensors such as thermal cameras for detecting gas leaks, and even for sizing the pipelines.

In addition to railway track detection and unburied onshore pipeline detection, the LS-Net can potentially be applied for road detection in low- and mid-altitude aerial imagery which facilitates many applications of UAVs such as traffic monitoring and surveillance, path planning, and inspection [46]. In UAV images, roads are usually very wide; hence, the edge detection approach as used in unburied onshore pipeline detection can be applied. Roads in satellite images, on the other hand, are usually very narrow and thus can be modeled as lines or curves; this means that the LS-Net can potentially be applied directly for detecting roads in satellite images.

6 Conclusion

This paper introduces LS-Net, a fast single-shot line segment detector. The LS-Net is by design fully convolutional, and it consists of three modules: (i) a fully convolutional feature extractor, (ii) a classifier, and (iii) a line segment regressor. The LS-Net can be trained end-to-end by back-propagation and stochastic gradient descent (SGD) via a weighted multitask loss function. The proposed loss function is a combination of Focal loss for addressing the class imbalance in classification and Wing loss for restoring the balance between the influence of errors of different sizes in multiple points regression.

With a customized version of the VGG-16 network as the backbone, the proposed approach outperforms existing state-of-the-art DL-based power line detection approaches. In addition, the LS-Net can run in near real-time (21.5 FPS), which can facilitate real-time power line detection for obstacle avoidance in low-altitude UAV flights, for vision-based UAV navigation and inspection in automatic autonomous power line inspection. Since the LS-Net can be trained end-to-end and performs very well even when trained only on synthetic images, it can potentially be adapted for detecting other linear structures, such as railway tracks, unburied onshore pipelines, and roads from low- and mid-altitude aerial images.

Acknowledgements The authors would like to thank eSmart Systems and UiT Machine Learning Group for support in the work with this paper. This work was supported by the Research Council of Norway [RCN NÆRINGSPHD grant no. 263894 (2016-2018) on Power Grid Image Analysis] and eSmart Systems.

Funding Open Access funding provided by UiT The Arctic University of Norway.

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: A system for large-scale machine learning. In: 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16), pp. 265–283 (2016)
2. Barchyn, T.E., Hugenholtz, C.H., Myshak, S., Bauer, J.: A UAV-based system for detecting natural gas leaks. J. Unmanned Veh. Syst. **6**(1), 18–30 (2017)
3. Boyat, A.K., Joshi, B.K.: A review paper: noise models in digital image processing. Sig. Image Process. **6**(2), 63 (2015)
4. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
5. Bresenham, J.E.: Algorithm for computer control of a digital plotter. IBM Syst. J. **4**(1), 25–30 (1965)
6. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679 (1986)
7. Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (1972)
8. Feng, Z.H., Kittler, J., Awais, M., Huber, P., Wu, X.J.: Wing loss for robust facial landmark localisation with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
9. Galamhos, C., Matas, J., Kittler, J.: Progressive probabilistic Hough transform for line detection. In: Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), vol. 1, pp. 554–560. IEEE (1999)
10. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
12. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, chap. 6.2.1, p. 173. MIT Press (2016)
13. Goyal, P., Kaiming, H.: Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. **39**, 2999–3007 (2018)
14. Gubbi, J., Varghese, A., Balamuralidhar, P.: A new deep learning architecture for detection of long linear infrastructure. In:

2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), pp. 207–210. IEEE (2017)
15. Jianzhuang, L., Wenqing, L., Yupeng, T.: Automatic thresholding of gray-level pictures using two-dimension otsu method. In: China., 1991 International Conference on Circuits and Systems, pp. 325–327. IEEE (1991)
16. Kasturi, R., Camps, O.I.: Wire detection algorithms for navigation. NASA Technical Report (2002)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 (2015)
18. Lafortune, E.P., Willems, Y.D.: A theoretical framework for physically based rendering. Comput. Graph. Forum **13**(2), 97–107 (1994)
19. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (1989)
20. Lee, S.J., Yun, J.P., Choi, H., Kwon, W., Koo, G., Kim, S.W.: Weakly supervised learning with convolutional neural networks for power line localization. In: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1–8. IEEE (2017)
21. Li, Y., Pan, C., Cao, X., Wu, D.: Power line detection by pyramidal patch classification. IEEE Trans. Emerg. Topics Comput. Intell. **3**(6), 416–426 (2018)
22. Li, Z., Liu, Y., Hayward, R., Zhang, J., Cai, J.: Knowledge-based power line detection for UAV surveillance and inspection systems. In: 2008 23rd International Conference Image and Vision Computing New Zealand, pp. 1–6. IEEE (2008)
23. Li, Z., Liu, Y., Walker, R., Hayward, R., Zhang, J.: Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved Hough transform. Mach. Vis. Appl. **21**(5), 677–686 (2010)
24. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: single shot multibox detector. In: European Conference on Computer Vision, pp. 21–37. Springer, Berlin (2016)
25. Madaan, R., Maturana, D., Scherer, S.: Wire detection using synthetic data and dilated convolutional networks for unmanned aerial vehicles. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3487–3494. IEEE (2017)
26. Nguyen, V.N., Jenssen, R., Roverso, D.: Automatic autonomous vision-based power line inspection: a review of current status and the potential role of deep learning. Int. J. Electr. Power Energy Syst **99**, 107–120 (2018)
27. Nguyen, V.N., Jenssen, R., Roverso, D.: Intelligent monitoring and inspection of power line components powered by UAVs and deep learning. IEEE Power Energy Technol. Syst. J. **6**(1), 11–21 (2019)
28. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
29. Pan, C., Shan, H., Cao, X., Li, X., Wu, D.: Leveraging spatial context disparity for power line detection. Cogn. Comput. **9**(6), 766–779 (2017)
30. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
31. Shan, H., Zhang, J., Cao, X., Li, X., Wu, D.: Multiple auxiliaries assisted airborne power line detection. IEEE Trans. Industr. Electron. **64**(6), 4810–4819 (2017)
32. Shapiro, L., Stockman, G.: Computer Vision, 1st edn, p. 154. Prentice Hall, New Jersey (2001)
33. Simard, P.Y., Steinkraus, D., Platt, J.C., et al.: Best practices for convolutional neural networks applied to visual document analysis. In: Icdar vol. 3, no. 2003 (2003)
34. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)
35. Singh, A.K., Swarup, A., Agarwal, A., Singh, D.: Vision based rail track extraction and monitoring through drone imagery. ICT Express **5**(4), 250–255 (2019)
36. Song, B., Li, X.: Power line detection from optical images. Neurocomputing **129**, 350–361 (2014)
37. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.A.: Striving for simplicity: The all convolutional net. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings (2015). arXiv:1412.6806
38. Van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T.: Scikit-image: image processing in python. PeerJ **2**, e453 (2014)
39. Wolberg, G.: Digital Image Warping, vol. 10662. IEEE computer society press, Los Alamitos (1990)
40. Wong, S.C., Gatt, A., Stamatescu, V., McDonnell, M.D.: Understanding data augmentation for classification: when to warp? In: 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp. 1–6. IEEE (2016)
41. Wu, Y., He, K.: Group normalization. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19 (2018)
42. Yan, G., Li, C., Zhou, G., Zhang, W., Li, X.: Automatic extraction of power lines from aerial images. IEEE Geosci. Remote Sens. Lett. **4**(3), 387–391 (2007)
43. Yetgin, Ö.E., Gerek, Ö.N.: Ground truth of powerline dataset (infrared-ir and visible light-vl). Mendeley Data, v8, **8** (2017) https://doi.org/10.17632/twxp8xccsw
44. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In: 4th International Conference on Learning Representations, ICLR 2016 (2016)
45. Zhang, J., Shan, H., Cao, X., Yan, P., Li, X.: Pylon line spatial correlation assisted transmission line detection. IEEE Trans. Aerosp. Electron. Syst. **50**(4), 2890–2905 (2014)
46. Zhou, H., Kong, H., Wei, L., Creighton, D., Nahavandi, S.: On detecting road regions in a single UAV image. IEEE Trans. Intell. Transp. Syst. **18**(7), 1713–1722 (2016)

Van Nhan Nguyen received his B.Eng. (2014) in computer science & engineering from Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam, a M.S. (2016) in computer science from Østfold University College, Halden, Norway, and a Ph.D. (2019) in deep learning at the UiT Machine Learning Group, UiT The Arctic University of Norway, Tromsø, Norway. His research interests include deep vision (deep learning for computer vision), especially image classification, object detection, semantic segmentation, one-shot learning, zero-shot learning, deep learning-based line detection, and vision-based automatic autonomous inspection of power lines based on unmanned aerial vehicles (UAV) and deep learning.

Robert Jenssen (M02) received the Ph.D. (Dr. Scient.) in Electrical Engineering from the University of Tromsø in 2005. Currently, he is a Professor with the Department of Physics and Technology, UiT The Arctic University of Norway, Tromsø, Norway. He directs the UiT Machine Learning Group: http://site.uit.no/ml. He is also a Research Professor with the Norwegian Computing Center, Oslo, Norway. He was a Guest Researcher with the Technical University of Denmark,

Kongens Lyngby, Denmark, from 2012 to 2013, with the Technical University of Berlin, Berlin, Germany, from 2008 to 2009, and with the University of Florida, Gainesville, FL, USA, from 2002 to 2003, spring 2004, and spring 2018. Jenssen is on the IEEE Technical Committee on Machine Learning for Signal Processing (MLSP). He is the president of the Norwegian section of IAPR (NOBIM–nobim.no) and serves on the IAPR Governing Board. He is an associate editor with the journal Pattern Recognition since 2010.

Davide Roverso CAO (Chief Analytics Officer), holds a PhD degree in Computing Science. He has over 25 years of experience in the field of Machine Learning and Big Data Analytics, with applications in diagnostics, prognostics, condition monitoring, and early fault detection in complex processes, in sectors ranging from energy to medicine and environmental monitoring. He has authored over 100 publications in international journals, conference proceedings and edited books. He is currently Chief Analytics Officer at eSmart Systems, where he leads the analytics and data science group.