Computer Vision and Image Understanding 194 (2020) 102947

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

ALCN: Adaptive Local Contrast Normalization ☆

Mahdi Rad [a,*], Peter M. Roth [a], Vincent Lepetit [b,a]

[a] Institute of Computer Graphics and Vision, Graz University of Technology, Graz, Austria
[b] LIGM, IMAGINE, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France

ARTICLE INFO

Communicated by Nikos Paragios

ABSTRACT

To make Robotics and Augmented Reality applications robust to illumination changes, the current trend is to train a Deep Network with training images captured under many different lighting conditions. Unfortunately, creating such a training set is a very unwieldy and complex task. We therefore propose a novel illumination normalization method that can easily be used for different problems with challenging illumination conditions. Our preliminary experiments show that among current normalization methods, the Difference-of-Gaussians method remains a very good baseline, and we introduce a novel illumination normalization model that generalizes it. Our key insight is then that the normalization parameters should depend on the input image, and we aim to train a Convolutional Neural Network to predict these parameters from the input image. This, however, cannot be done in a supervised manner, as the optimal parameters are not known *a priori*. We thus designed a method to train this network jointly with another network that aims to recognize objects under different illuminations: The latter network performs well when the former network predicts good values for the normalization parameters. We show that our method significantly outperforms standard normalization methods and would also be appear to be universal since it does not have to be re-trained for each new application. Our method improves the robustness to light changes of state-of-the-art 3D object detection and face recognition methods.

1. Introduction

Over the last years, Deep Networks (LeCun et al., 1998; Krizhevsky et al., 2012; Simonyan and Zisserman, 2015) have spectacularly improved the performance of computer vision applications. Development efforts to date, however, have mainly been focused on tasks where large quantities of training data are available. To be robust to illumination conditions for example, one can train a Deep Network with many samples captured under various illumination conditions.

While for some general categories such as faces, cars, or pedestrians, training data can be exploited from other data, or the capturing of many images under different conditions is also possible, these processes become very unwieldy and complex tasks for others. For example, as illustrated in Fig. 1, we want to estimate the 3D pose of specific objects without having to vary the illumination when capturing training images. To achieve this, we could use a contrast normalization technique such as Local Contrast Normalization (Jarrett et al., 2009), Difference-of-Gaussians or histogram normalization. Our experiments show, however, that existing methods often fail when dealing with large magnitudes of illumination changes.

Among the various existing normalization methods, Difference-of-Gaussians still performs best in our experiments, which inspired us to introduce a normalization model building on a linear combination of 2D Gaussian kernels with fixed standard deviations. But instead of using fixed parameters, we propose to adapt these parameters to the illumination conditions of the different image regions: By this means, we can handle bigger illumination changes and avoid manual tuning.

However, the link between a given image and the best parameters is not straightforward. We therefore want to learn to predict these parameters from the image using a CNN. Since we do not have *a priori* knowledge-the parameters to predict, we cannot train this CNN in a standard supervised manner. Our solution is to train it *jointly* in a supervised way together with another CNN to achieve object detection under illumination changes.

We call this method Adaptive Local Contrast Normalization (ALCN), as it is related to previous Local Contrast Normalization methods while being adaptive. We show that ALCN outperforms previous methods for illumination normalization by a large margin while we do not need any manual tuning. It also outperforms Deep Networks including VGG (Simonyan and Zisserman, 2015) and ResNet (He et al., 2016)

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.cviu.2020.102947.

* Corresponding author.
E-mail address: rad@icg.tugraz.at (M. Rad).

https://doi.org/10.1016/j.cviu.2020.102947
Received 18 April 2019; Received in revised form 19 December 2019; Accepted 21 February 2020
Available online 29 February 2020

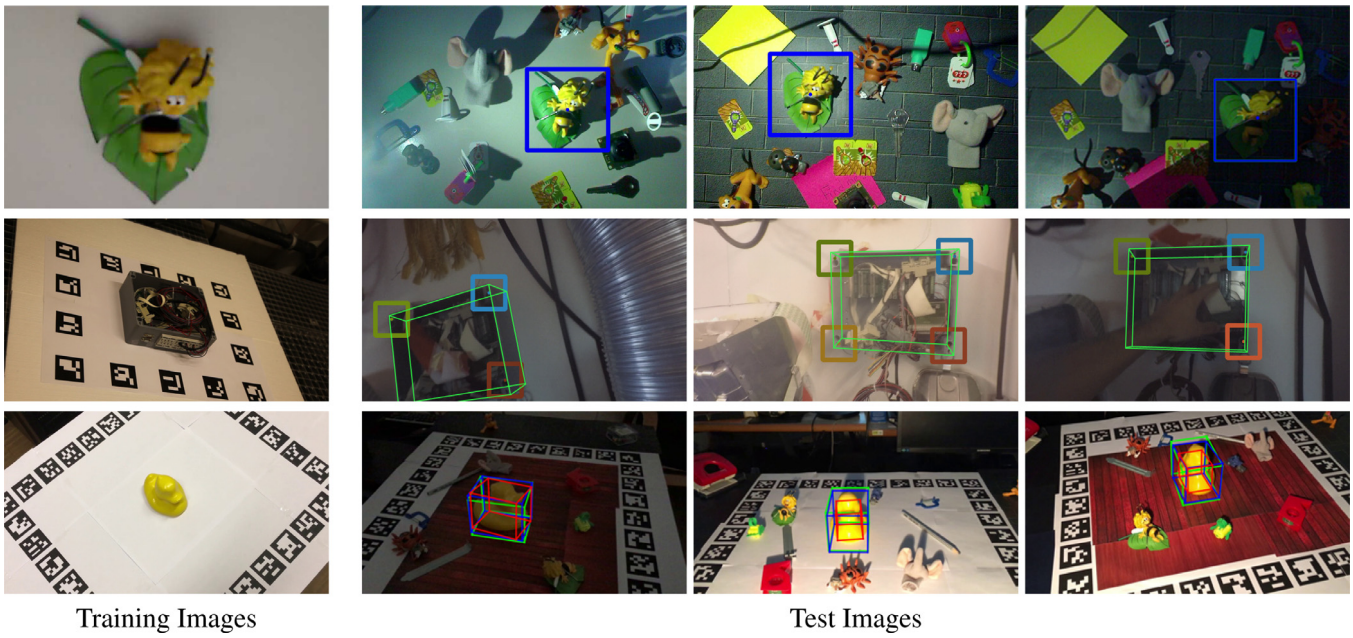

Fig. 1. We propose a novel approach to illumination normalization, which allows us to deal with strong light changes even when only few training samples are available. We apply it to 2D detection (first row) and 3D object detection using the methods of Crivellaro et al. (2015) (second row) and of Rad and Lepetit (2017) (third row): Given training images under constant illumination, we can detect the object and predict its pose under various and drastic illumination. In the third row, green bounding boxes show the ground truth pose, blue bounding boxes represent the pose obtained with ALCN, and red bounding boxes the pose obtained without normalization. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

trained on the same images, showing that our approach can generalize better with unseen illumination variations than a single network.

In summary, our main contribution is an efficient method that makes Deep Networks more robust to illumination changes that have not been seen during training, therefore requiring much far less training data. Furthermore, we created new datasets for benchmarking of object detection and 3D pose estimation under challenging lightening conditions with distractor objects and cluttered background.

We published a first version of this work in Rad et al. (2017). This paper extends this work in the following manner:

- We provide an extensive overview of existing normalization methods.
- We perform thorough ablation studies to justify our contribution.
- We also perform experiments on network design and the impact of different activation functions.
- We evaluate our normalization method on other applications such as 3D object detection and pose estimation and face recognition, which our approach was not trained for.

In the remainder of this paper, we first discuss related work in Section 2, we then review the existing normalization methods and introduce our normalization model in Section 3, and we evaluate it on different applications in Section 4.

2. Related work

Reliable computer vision methods need to be invariant, or at least robust, to many different visual nuisances, including pose and illumination variations. In the following, we give an overview of the different, and sometimes complementary approaches for achieving this.

Image normalization methods. A first approach is to normalize the input image using image statistics. Several methods have been proposed, sometimes used together with Deep Networks such as SLCN and DLCN: Difference-of-Gaussians (DoG), Whitening, Subtractive and Divisive Local Contrast Normalization (SLCN and DLCN) (Jarrett et al., 2009), Local Response Normalization (LRN) (Krizhevsky et al., 2012), Histogram Equalization (HE), Contrast Limited Adaptive Histogram Equalization (CLAHE) (Pizer et al., 1987). We detail these methods in Section 3.1, and compare to them in our experiments in Section 4.

However, illumination is not necessarily uniform over an image: Applying one of these methods locally over regions of the image handles local light changes better, but unfortunately they can also become unstable on poorly textured regions. Our approach overcomes this limitation with an adaptive method that effectively adjusts the normalization according to the local appearance of the image.

Invariant features. An alternative method is to use locally invariant features. For example, Haar wavelets (Viola and Jones, 2004) and the pairwise intensity comparisons used in Local Binary Patterns (Ojala et al., 2002) are invariant to monotonic changes of the intensities. Features based on image gradients are invariant to constants added to the intensities. In practice, they are also often made invariant to affine changes by normalizing gradient magnitudes over the bins indexed by their orientations (Levi and Weiss, 2004). The SIFT descriptors are additionally normalized by an iterative process that makes them robust to saturation effects as well (Lowe, 2004). However, it is difficult to come up with features that are invariant to complex illumination changes on 3D objects, such as changes of light direction, cast or self shadows.

Intrinsic images. A third approach is to model illumination explicitly and estimate an intrinsic image or a self quotient image of the input image, to get rid of the illumination and isolate the reflectance of the scene as an invariant to illumination (Wang et al., 2004; Shen et al., 2011a,b, 2013; Zhou et al., 2015; Nestmeyer and Gehler, 2017; Fan et al., 2018; Bi et al., 2015). However, it is still difficult to get an intrinsic image from one single input image that is good enough for computer vision tasks, as our experiments in Section 4 will show for 2D object detection.

Data-driven robustness. The current trend to achieve robustness to illumination changes is to train Deep Networks with different illuminations present in the training set (Simonyan and Zisserman, 2015; He et al., 2016; Huang et al., 2017; Xie et al., 2017). This, however, requires the

acquisition of many training images under various conditions. As we will show, our approach performs better than single Deep Networks when illumination variations are limited in the training set, which can be the case in practice for some applications.

3. Adaptive local contrast normalization

In this section, we first provide an overview of the existing normalization methods, since we will compare our method against them in Section 4.2.2. We then introduce our normalization model, then we discuss how we train a CNN to predict the model parameters for a given image region and how we can efficiently extend this method to a whole image.

3.1. Overview of existing normalization methods

We describe below the main different existing methods to make object detection techniques invariant to light changes. These are also the methods we will compare to in Section 4. *In practice, in our 2D object experiments presented below, we use a detector in a sliding window fashion, and we apply these normalization methods, including our Adaptive LCN, to each window independently.*

- **Normalization by Standardization (NS)**. A common method to be robust to light changes is to replace the input image I by:

$$I^{\mathrm{NS}} = \frac{(I - \bar{I})}{\sigma_I}\,, \tag{1}$$

where $\bar{I}$ and σ_I are respectively the mean and standard deviation of the pixel intensities in I. This transformation makes the resulting image window I^{NS} invariant to affine transformation of the intensities—if we ignore saturation effects that clip the intensity values within $[0; 255]$.

- **Difference-of-Gaussians (DoG)**. The Difference-of-Gaussians is a band-pass filter often used for normalization:

$$I^{\mathrm{DoG}} = (k_2^{\mathrm{DoG}} \cdot G_{\sigma_2^{\mathrm{DoG}}} - k_1^{\mathrm{DoG}} \cdot G_{\sigma_1^{\mathrm{DoG}}}) * I\,, \tag{2}$$

where G_σ is a 2D Gaussian filter of standard deviation σ, and k_1, k_2, σ_1, σ_2 are parameters. $*$ is the 2D convolution operator. This is also a common mathematical model for the ON- and OFF-center cells of the retina (Dayan and Abbott, 2005). In practice, we use Gaussian filters of size $\lceil 6\sigma+1 \rceil$, to truncate only very small values of the Gaussian kernels.

- **Whitening**. Whitening is sometimes used for illumination normalization. It is related to DoG as learned whitening filters computed from natural image patches resemble a Difference-of-Gaussians (Rigamonti et al., 2011; Goodfellow et al., 2013; Yang et al., 2015a,b). In practice, we first compute the whitening matrix as the inverse of the square root of the covariance matrix of the image patches. The columns of the whitening matrix are all translated versions of the same patch, and we use the middle column as the whitening convolutional filter (Rigamonti et al., 2011).

- **Local Contrast Normalization (LCN)**. When working with Deep Networks, Local Contrast Normalization (LCN) (Jarrett et al., 2009) is often used. We tried its two variants. Subtractive LCN is also closely related to DoG as it subtracts from every value in an image patch a Gaussian-weighted average of its neighbors:

$$I^{\mathrm{SLCN}} = I - G_{\sigma^{\mathrm{Sub}}} * I\,, \tag{3}$$

where σ^{Sub} is a parameter. Divisive LCN, the second variant, makes the image invariant to local affine changes by dividing the intensities in I^{SLCN} by their standard deviation, computed locally:

$$I^{\mathrm{DLCN}}(\mathbf{m}) = \frac{I^{\mathrm{SLCN}}(\mathbf{m})}{\max\left(t, \left(G_{\sigma^{\mathrm{Div}}} * (I^{\mathrm{SLCN}})^2\right)(\mathbf{m})\right)}\,, \tag{4}$$

where $(I^{\mathrm{SLCN}})^2$ is an image made of the squared intensities of I^{SLCN}, and σ^{Div} is a parameter controlling the size of the region for the local standard deviation of the intensities. t is a small value to avoid singularities.

- **Local Response Normalization (LRN)**. Local Response Normalization is related to LCN, and is also used in many applications to normalize the input image, or the output of the neurons (Krizhevsky et al., 2012; Badrinarayanan et al., 2015). The normalized value at location $\mathbf{m}$ after applying kernel i can be written as:

$$I_{(i)}^{\mathrm{LRN}}(\mathbf{m}) = \frac{I_{(i)}(\mathbf{m})}{\left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (I_{(j)}(\mathbf{m}))^2\right)^\beta} \tag{5}$$

where the sum is over the n kernel maps around index i, and N is the total number of kernels in the layer. Constants k, n, α and β are then manually selected. Compared to LCN, LRN aims more at normalizing the image in terms of brightness rather than contrast (Krizhevsky et al., 2012).

- **Histogram Equalization (HE)**. Histogram Equalization aims at enhancing the image contrast by better distributing the intensities of the input image. First, a histogram $p(\lambda_i)$ of the image intensities, with λ_i any possible quantized intensity value, is built. Then, a new intensity $\tilde{\lambda}_i$ is assigned to all the pixels with intensity λ_i, with

$$\tilde{\lambda}_i = \lambda_{\min} + \mathrm{floor}\left((\lambda_{\max} - \lambda_{\min}) \sum_{j=0}^{i} p(\lambda_j)\right)\,. \tag{6}$$

- **Contrast Limited Adaptive Histogram Equalization (CLAHE)**. While Histogram Equalization does not take the spatial location of the pixels into account, CLAHE (Pizer et al., 1987) introduces spatial constraints and attempts to avoid noise amplification: It performs Histogram equalization locally, and the histograms are clipped: If $p(\lambda_i)$ is higher than a threshold $\hat{\lambda}$, it is set to $\hat{\lambda}$ and the histogram is re-normalized.

- **Intrinsic Image**. An intrinsic image of an input image I can be obtained by separating the illumination S from the reflectance R of the scene:

$$I(\mathbf{m}) = S(\mathbf{m})R(\mathbf{m})\,. \tag{7}$$

Eq. (7) is ill-posed, but can be solved by adding various constraints (Shen et al., 2011a,b; Zhou et al., 2015). Since R is supposed to be free from illumination effects, it can then be used as input instead of the original image to be invariant to illuminations. However, it is still difficult to estimate R robustly, as our experiments will show. Moreover, optimizing over S and R under constraints is computationally expensive, especially for real-time applications.

- **Self Quotient Image (SQI)**. The Self Quotient Image (Wang et al., 2004) aims at estimating the object reflectance field from a 2D image similarly to the Intrinsic Image method, but is based on the Lambertian model instead of the reflectance illumination model. The Self Quotient Image Q of image I is defined by:

$$Q = \frac{I}{G_\sigma^{\mathrm{SQI}} * I}\,. \tag{8}$$

3.2. Normalization model

As our experiments in Section 4 will show, the Difference-of-Gaussians normalization method performs best among the existing normalization methods, however, it is difficult to find the standard deviations that perform well for any input image, as we will discuss in Section 3.4. We therefore introduce the following formulation for our ALCN method:

$$\mathrm{ALCN}(I;\ w) = \left(\sum_{i=1}^{N} w_i \cdot G_{\sigma_i^{\mathrm{ALCN}}}\right) * I\,, \tag{9}$$

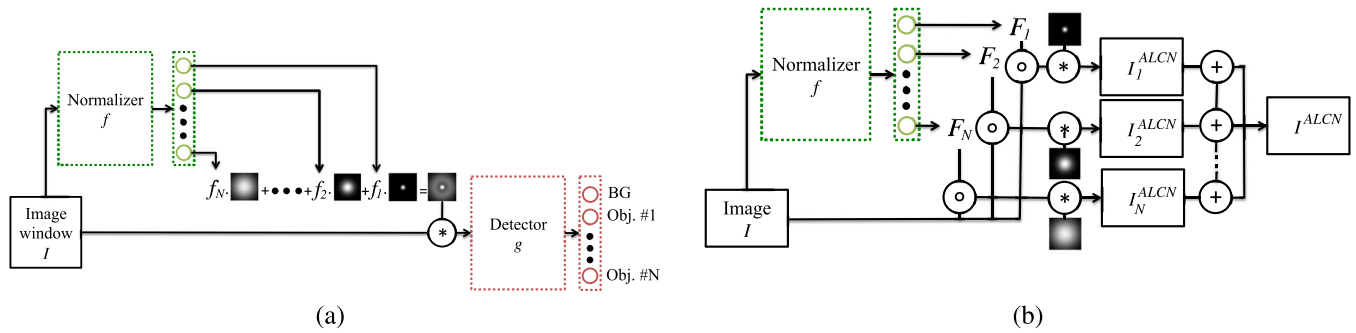

Fig. 2. Overview of our method. (a) We first train our Normalizer jointly with the Detector using image regions from the Phos dataset. (b) We can then normalize images of previously unseen objects by applying this Normalizer to predict the parameters of our normalization model.

Fig. 3. Four of the ten objects we use from the Phos dataset (Vonikakis et al., 2013) under different illuminations.

where I is an input image window, w a vector containing the parameters of the method and $\mathrm{ALCN}(I; w)$ is the normalized image; G_σ denotes a Gaussian kernel of standard deviation σ; the σ_i^{ALCN} are fixed standard deviations, and $*$ denotes the convolution product. In the experiments, we use ten different 2D Gaussian filters $G_{\sigma_i^{\mathrm{ALCN}}}$, with standard deviation $\sigma_i^{\mathrm{ALCN}} = i/2$ for $i = 1, 2, \ldots, 10$. This model is a generalization of the Difference-of-Gaussians model, since the normalized image is obtained by convolution of a linear combination of Gaussian kernels, and the weights of this linear combination are the parameters of the model.

Using fixed 2D Gaussian filters allows us to have fast running time: During training, we can perform the Gaussian convolutions on the samples of the mini-batches. It also makes training easier since the network has to predict only the weights of a linear combination. During testing, this allows us to efficiently varies the model parameters with the image locations efficiently, as will be explained in Section 3.5.

3.3. Joint training to predict the model parameters

As discussed in the introduction and shown in Fig. 2(a), we train a Convolutional Neural Network (CNN) to predict the parameters w of our model for a given image window I, jointly with an object classifier. We call this CNN the Normalizer.

Like the Normalizer, the classifier is also implemented as a CNN as well, since deep architectures perform well for such problems. This will also make joint training of the Normalizer and the classifier easy. We refer to this classifier as the Detector. Joint training is done by minimizing the following loss function:

$$(\hat{\Theta}, \hat{\Phi}) = \arg\min_{\Theta,\Phi} \sum_j \ell\left(g^{(\Theta)}\left(\mathrm{ALCN}(I_j;\ f^{(\Phi)}(I_j))\right);\ y_j\right)\ , \tag{10}$$

where Θ and Φ are the parameters of the Detector CNN $g(\cdot)$ and the Normalizer $f(\cdot)$, respectively; $\ell(\cdot; y)$ is the negative log-likelihood loss function. I_j and y_j are training image regions and their labels: We use image regions extracted from the Phos dataset (Vonikakis et al., 2013), including the images shown in Fig. 3, the labels are either background or the index of the object contained in the corresponding image region. We use Phos for our purpose because it is made of various objects under different illumination conditions, with 9 images captured under various strengths of uniform illumination and 6 images under non-uniform illumination from various directions. In practice, we use Theano (Bergstra et al., 2010) to optimize Eq. (10).

3.4. Different images need different parameters

In order to show that different images need different parameters when using previous normalization methods, we performed two studies. For each study we jointly optimizing the four DoG parameters, at the same time as the Detector:

$$(\hat{\Theta}, \hat{\Omega}) = \arg\min_{\Theta,\Omega} \sum_i \ell\left(g^{(\Theta)}\left(\mathrm{DoG}^{(\Omega)} * I_i\right); y_i\right)\ , \tag{11}$$

where Θ and Ω are the parameters of the Detector CNN $g(\cdot)$ and DoG respectively, the $\{(I_i, y_i)\}_i$ are annotated training image windows, and $\ell(\cdot; y)$ is the negative log-likelihood loss function.

Effect of brightness. In order to evaluate the effect of brightness, we split the training set into dark and bright images, by simply thresholding the mean intensity, and training two different detectors, one for each subset, We will give more details of the dataset in Section 4.1.

We set the intensity threshold to 80 in our experiments. At runtime, for each possible image location, we first tested if the image patch centered on this location is dark or bright, and apply the corresponding CNN. We also trained two more detectors, one for each subset, but this time with the optimized parameter values obtained on the other subset.

Fig. 4 shows that the optimized parameters found for one subset are only optimized for that subset and not the other one. It also shows that larger values for σ_1^{DoG} and σ_2^{DoG} perform better on the dark test images. Fig. 5 shows that our adaptive Normalizer learns to reproduce this behavior, applying larger receptive fields to darker images and vice-versa.

Different objects. In order to evaluate how different objects with different shapes and material affect on the predicted parameters, we optimize on only one object of the Phos dataset at a time. As illustrated in Fig. 6, different kernels are learned for different objects.

3.5. From window to image normalization

Once trained on windows, we apply the Normalizer to the whole input images by extending Eq. (9) to

$$\mathrm{ALCN}(\mathbf{I}) = \sum_{k=1}^{N} G_{\sigma_k^{\mathrm{ALCN}}} * \left(F_k(\mathbf{I}) \circ \mathbf{I}\right)\ , \tag{12}$$

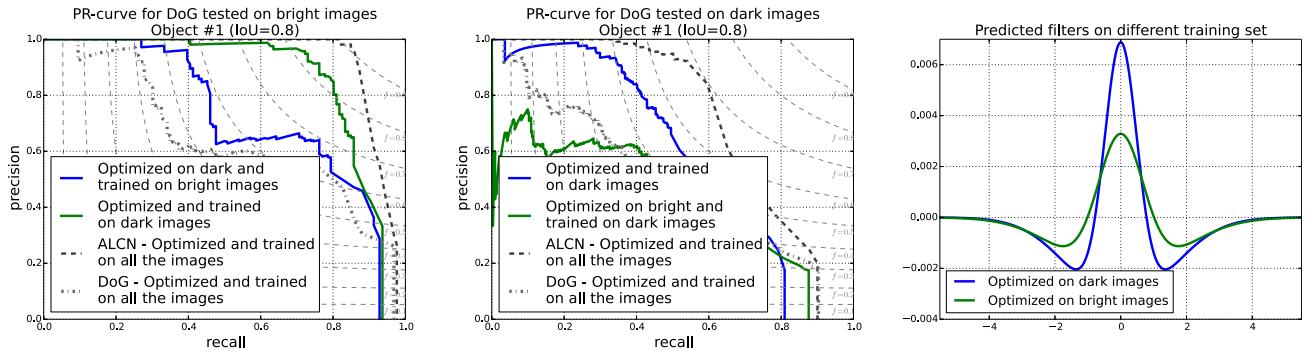

Fig. 4. Evaluating the relations between brightness and optimized normalization parameter values. We split our dataset into bright and dark images. The best performance on the dark images is obtained with larger filters than for the bright images.

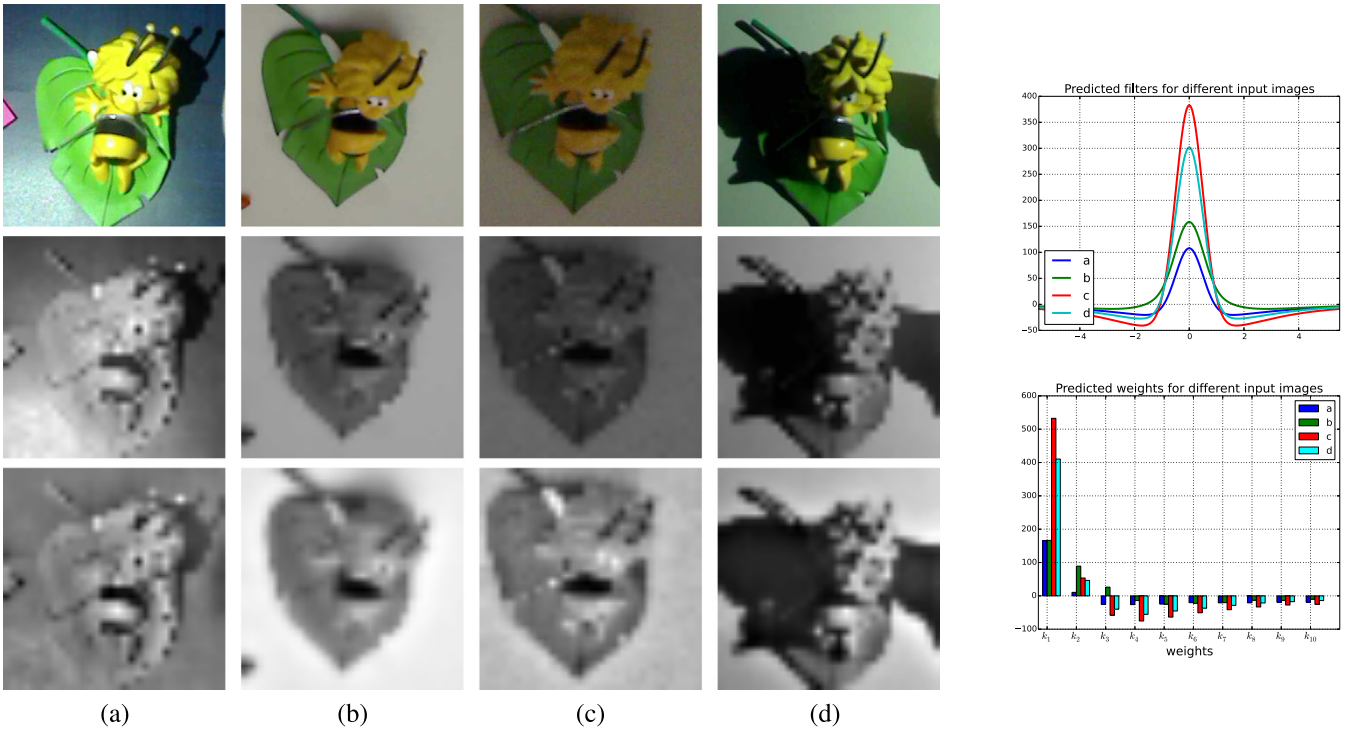

Fig. 5. Left (a–d): Four input window images of the plastic toy under different illuminations. First row: original window. Second row: window after downscaling, and used as input to the normalizer. Third row: window after normalization by the filter predicted by the Normalizer. Top-right: Predicted filters using the model of Eq. (9) for the image windows a–d, shown in 1D for clarity. The filters predicted for dark images are larger than the ones predicted for bright images. Bottom-right: The 10 predicted coefficients w_i for the image windows a–d.

where $F_k(\mathbf{I})$ is a weight matrix with the same dimension as the input image $\mathbf{I}$ for the kth 2D Gaussian filter, and $\circ$ is the Hadamard (element-wise) product. The weight matrix $F_k(\mathbf{I})$ corresponding to the kth 2D Gaussian filter is computed as $\left(F_k(\mathbf{I})\right)_{ij} = f_k(I_{ij})$, where $(\cdot)_{ij}$ is the entry in the ith row and jth column of the matrix, I_{ij} is the image window centered at (i, j) in image $\mathbf{I}$, and $f_k(\cdot)$ is the kth weight predicted by the Normalizer for the given image window. This can be done very efficiently by sharing the convolutions between windows (Giusti et al., 2013).

Normalization is therefore different for each location of the input image. This allows us to adapt better to the local illumination conditions. Because it relies on Gaussian filtering, it is also fast, taking only 50 ms for 10 2D Gaussian filters, on an Intel Core i7-5820K 3.30 GHz desktop with a GeForce GTX 980 Ti on a 128 × 128 image.

3.6. Color image normalization

For some applications, such as 3D object pose estimation, it is important to be able to normalize not only grayscale images, but also color images as well, as colors bring very valuable information. To do so, as in Zhang et al. (2016), we first transform the input color image in the CIE Lab colorspace, normalize the lightness map L with our method, and re-transform the image in the RGB space without changing the ab channels. An example of a normalized color image using this method is shown in Fig. 7(e).

3.7. Network architecture and optimization details

A relatively simple architecture is sufficient for the Normalizer: In all of our experiments, the first layer performs 20 convolutions with 5 × 5 filters with 2 × 2 max-pooling. The second layer performs 50 5 × 5 convolutions followed by 2 × 2 max-pooling. The third layer is a fully connected layer of 1024 hidden units. The last layer returns the predicted weights. In order to keep optimization tractable, we downscaled the training images of the target objects by a factor of 10. To avoid border effects, we use 48 × 48 input patches for the Normalizer, and use 32 × 32 patches as input to the Detector. We use

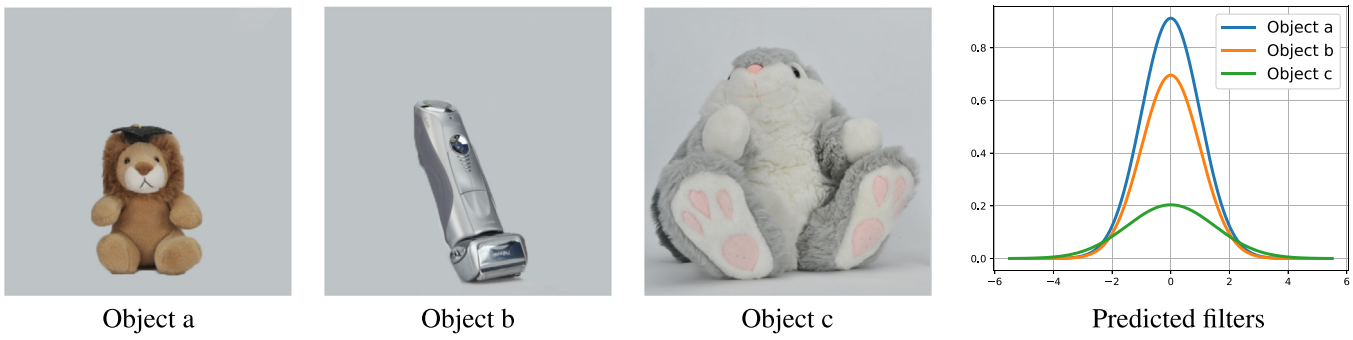

Fig. 6. Predicted filters for DoG normalization trained on different objects.

Fig. 7. (a): original RGB image. (b): ab channel in CIE Lab color space. (c): grayscaled image. (d): normalized grayscale image. (e): normalized color image.

the tanh function as activation function, as it performs better than ReLU on our problem. This difference is because a sigmoid can better control the large range of intensities exhibited in the images of our dataset, while other datasets have much more controlled illuminations.

3.8. Generating synthetic images

Since the Phos dataset is small, we augment it by applying simple random transformations to the original images. We segmented the objects manually, so that we can change the background easily.

We experimented with several methods to artificially change the illuminations, and we settled to the following formulas to generate a new image I_{new} given an original image I_{ref}. We first scale the original image, randomly replace the background, and scale the pixel intensities:

$$I_{\text{interm}} = a(\text{bg}(\text{scale}_s(I_{\text{ref}}))) + b\,, \tag{13}$$

where a, b, and s are value randomly sampled from the ranges $[1-A;1+A]$, $[-B;+B]$, and $[1-S;1+S]$ respectively. $\text{bg}(\cdot)$ is a function that replaces the background of the image by a random background, which can be uniform or cropped from an image from the ImageNet (dataset Deng et al. (2009)). $\text{scale}_s(\cdot)$ is a function that upscales or downscales the original image by a factor s.

The generated image is then taken as

$$I_{\text{new}} = \text{clip}(G(I_{\text{interm}}))\,, \tag{14}$$

where $G(\cdot)$ adds Gaussian noise, and $\text{clip}(\cdot)$ is the function that clips the intensity values to the $[0;255]$ interval. This function allows us to simulate saturation effects, and makes the transformation non-linear, even in the absence of noise. I_{interm} is an intermediate image that can influence the amount of noise: In all our experiments, we use $A = 0.5$, $B = 0.4$, and $S = 0.1$.

We generate 500,000 synthetic images, with the same number of false and negative images. Once the Normalizer is trained on the Phos dataset, we can use synthetic images created from a very small number of real images of the target objects to train a new classifier to recognize these objects: Some of our experiments presented below use only one real image. At test time, we run the Detector on all 48 × 48 image windows extracted from the test image.

4. Experiments

In this section, we first introduce the datasets we used for evaluating the methods described in the previous section, including our own. We then present the network architecture and optimization details. We perform thorough ablation studies to demonstrate our contribution, by benchmarking on 2D object detection under drastic illumination changes when only few images are available for training. We finally evaluate our normalization to improve the robustness to light changes of state-of-the-art 3D object detection and pose estimation, face recognition and semantic segmentation methods.

4.1. Datasets

Some datasets have instances captured under different illuminations, such as NORB (LeCun et al., 2004), ALOI (Geusebroek et al., 2005), CMU Multi-PIE (Gross et al., 2009) or Phos (Vonikakis et al., 2013). However, they are not suitable for our purposes: NORB has only 6 different lighting directions; the images of ALOI contain a single object only and over a black background; CMU Multi-PIE was developed for face recognition and the image is always centered on the face; Phos was useful for our joint training approach, however, it has only 15 test images and the objects are always at the same locations, which would make the evaluation dubious.

We thus created a new dataset for benchmarking object detection under challenging lighting conditions and cluttered background. We will refer to this dataset as the ALCN-2D dataset. As shown in Fig. 8, we selected three objects spanning different material properties: plastic (Object #1), velvet (Object #2) and metal (Object #3) (velvet has a BRDF that is neither Lambertian nor specular Lu et al., 1998, and the metallic object—the watch—is very specular). For each object, we have 10 300 × 300 grayscale training images and 1200 1280 × 800 grayscale test images, exhibiting these objects under different illuminations, different lighting colors, and distractors in the background. The number of test images is therefore much larger than for previous datasets. We manually annotated the ground truth bounding boxes in the test images in which the target object is present. In this first dataset, the objects are intentionally moved on a planar surface, in order to

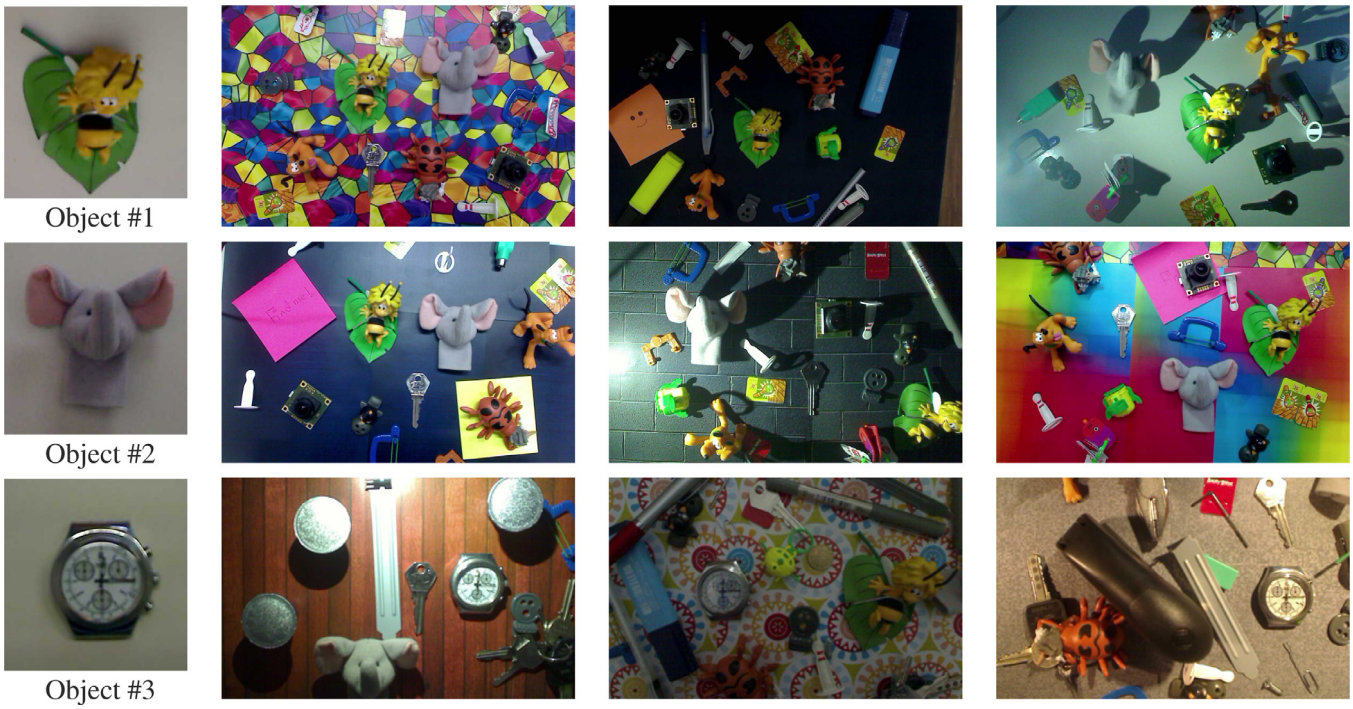

Fig. 8. The objects for our ALCN-2D dataset, and representative test images. We selected three objects spanning different material properties: plastic (Object #1), velvet (Object #2), metal (Object #3) (velvet has a BRDF that is neither Lambertian nor specular, and the metallic object – the watch – is very specular). By contrast with previous datasets, we have a very large number of test images (1200 for each object), capturing many different illuminations and background.

limit the perspective appearance changes and focus on the illumination variations.

The second dataset we consider is the BOX Dataset from the authors of Crivellaro et al. (2015), which combines perspective and light changes. It is made of a registered training sequence of an electric box under various 3D poses but a single illumination and a test sequence of the same box under various 3D poses and illuminations. Some images are shown in the second row of Fig. 1. This test sequence was not actually part of the experiments performed by Crivellaro et al. (2015) since it was too challenging in scope. The goal is to estimate the 3D pose of the box.

Finally, we introduce another dataset for 3D pose estimation. This dataset is made of a training sequence of 1000 registered frames of the Duck from the Hinterstoisser dataset (Hinterstoisser et al., 2012) obtained by 3D printing under a single illumination and 8 testing sequences under various illuminations. Some images are shown in the third row of Fig. 1. We will refer to this dataset as the ALCN-Duck dataset.

4.2. Experiments and discussion

For evaluation, we use the PASCAL criterion to decide if a detection is correct with an Intersection over Union of 0.8, with fixed box sizes of 300×300, reporting Precision–Recall (PR) curves and Areas Under Curve (AUC) in order to compare the performances of the different methods.

4.2.1. Explicit normalization vs. illumination robustness with deep learning

As mentioned in the introduction, Deep Networks can learn robustness to illumination variations without explicitly handling them, at least to some extent. To show that our method allows us to go further, we first tried to train several Deep Network architectures from scratch, without normalizing the images beforehand, by varying the number of layers and the number of filters for each layer. We use one real example of each object in the ALCN-2D dataset for this experiment. The best architecture we found performs with an AUC of 0.606. Our method, however, still performs better with an AUC of 0.787. This shows that our approach achieves better robustness to illumination than a single CNN, at least when the training set is limited, as in our scenario.

We also evaluated Deep Residual Learning Network architectures (He et al., 2016). We used the same network architectures and training parameters as in He et al. (2016) on CIFAR-10. ResNets with 20, 32, 44, 56 and 110 layers perform with AUCs of 0.456, 0.498, 0.518, 0.589 and 0.565 respectively, which is still outperformed by a much simpler network when our normalization is used. Between 56 and 110 layers, the network starts overfitting, and increasing the number of layers results in a decrease of performance.

4.2.2. Comparing ALCN against previous normalization methods

In our evaluations, we consider different existing methods described in Section 3.1. In order to assess the effects of different normalization techniques on the detection performances, we employed the same detector architecture for the normalization methods, but re-training it for every normalization method. Fig. 9 compares these methods on the ALCN-2D dataset. For DoG, Subtractive and Divisive LCN, we optimized their parameters to perform best on the training set. We tried different method of intrinsic image decomposition and they perform with similar accuracy. In this paper, we use the implementation of Shen et al. (2013), which performs slightly better on the ALCN-2D dataset, compare to other implementations. Our method consistently outperforms the others for all objects of the ALCN dataset. Most of the other methods have very different performances across the different objects of the dataset. Whitening obtained an extremely bad score for all objects, while both versions of LCN failed in detecting Object #3, the most specular object, obtaining an AUC score lower than 0.1.

4.2.3. Impact of number of real images

Once the Normalizer is trained on the Phos dataset, we freeze its weights and plug it to a detector to detect target object. To train the Detector, we use 500,000 synthetically generated images with the same way as described in Section 3.8. Some synthetic images generated are shown in Fig. 11. These 500,000 images can be generated

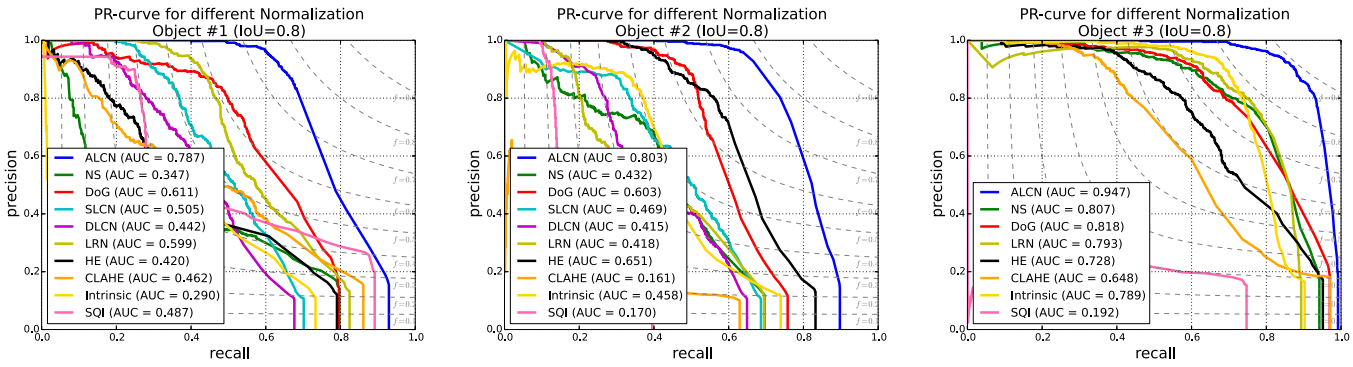

Fig. 9. Comparing different normalization methods using the best parameter values for each method for Objects #1, #2 and #3 of ALCN-2D. ALCN systematically performs best by a large margin.

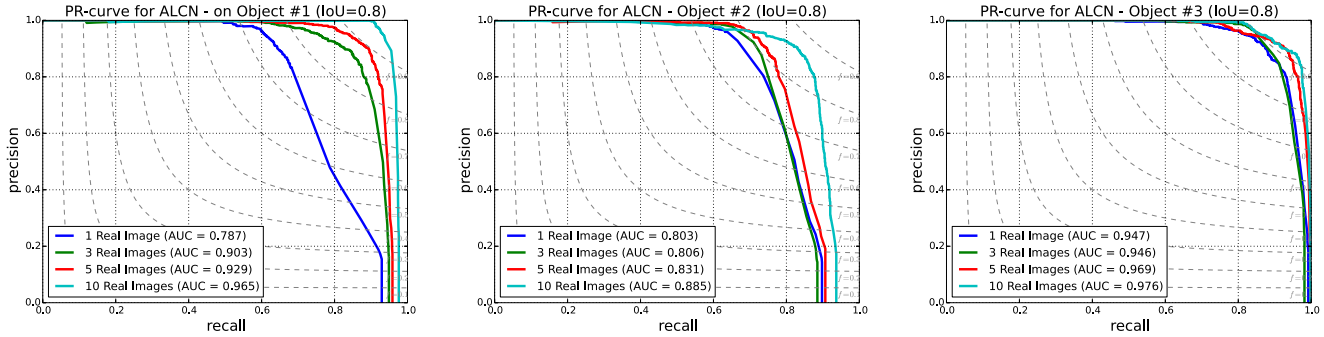

Fig. 10. Evaluating the influence of the number of real images used for training the detector on Objects #1, #2 and #3 of ALCN-2D. The detection accuracy keeps increasing when using more real images for generating the training set.

Fig. 11. Some synthetic images generated from the object #1 shown in Fig. 8.

either from only one single real image, or more. In Section 4.2.2, we showed that using only one real image to generate the whole training set already gives us good results. Fig. 10 illustrates that using more real images while keeping the total number of synthetically generated images same as before, improves the performances further, 10 real images are enough for very good performance. This shows that we can learn to detect objects under very different drastic illuminations from very few real examples augmented with simple synthetic examples.

4.2.4. Activation functions

While sigmoid functions were originally used in early neural networks and CNNs, the popular choice is now the ReLU operator, because it often eases tuning the convergence as the derivatives are constant, while special care is to be taken when using sigmoids.

However, Fig. 12 shows that using the hyperbolic tangent tanh sigmoid function yields clearly better results than using the ReLU activation functions on our problem. This difference is because a sigmoid can control better the large range of intensities exhibited in the images of our dataset, while other datasets have much more controlled illuminations.

4.3. Image normalization for other applications

In this section, we evaluate our normalization method on applications for which it was not trained for: (1) 3D object detection, (2) 3D object pose estimation, and (3) face detection and recognition.

4.3.1. 3D object pose estimation

As mentioned in the introduction, our main goal is to train Deep Networks methods for 3D pose estimation, without requiring large quantities of training data while being robust to light changes. We evaluate here ALCN for this goal on two different datasets.

BOX dataset. To evaluate ALCN for 3D object detection and pose estimation, we first applied it on the BOX dataset described in Section 4.1 using the method of Crivellaro et al. (2015), which is based on part detection: It first learns to detect some parts of the target object, then it predicts the 3D pose of each part to finally combine them to estimate the object 3D pose.

The test videos from Crivellaro et al. (2015) exhibit challenging dynamic complex background and light changes. We changed the code provided by the authors to apply ALCN before the part detection. We evaluated DoG normalization, the second best method according to our previous experiments, optimized on these training images, against our Normalizer. Fig. 13 shows the results; ALCN allows us to detect the parts more robustly and thus to compute much more stable poses.

ALCN-Duck dataset. The method proposed in Rad and Lepetit (2017) first detects the target object using a detector and then, given the image window centered on the object, predicts the 3D pose of the object using a regressor. For both, detector and regressor, Rad and Lepetit (2017) finetunes convolution and fully connected layers of VGG (Simonyan and Zisserman, 2015), and achieved very good results on the LineMOD dataset. However, this dataset does not exhibit strong light changes, and we evaluated our approach on the ALCN-Duck dataset described in Section 4.1. Here, we use color images as input to the detector and the regressor. To apply ALCN to these images, we use the method proposed in Section 3.6. We normalized color images by normalizing the L channel in CIE color space. As our experiments show even if ALCN was trained on grayscale images, we get reasonably good normalized color images. Fig. 14 shows the normalized images of the ALCN-Duck dataset.

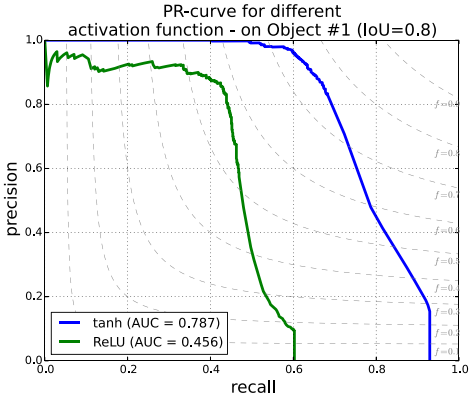

Fig. 12. Influence of the activation function. The plots show the PR curves using our normalization, applying either the ReLU operator, or the sigmoid tanh function as activation function. tanh appears to perform better, probably because it helps to control the range of intensity values in our test images.

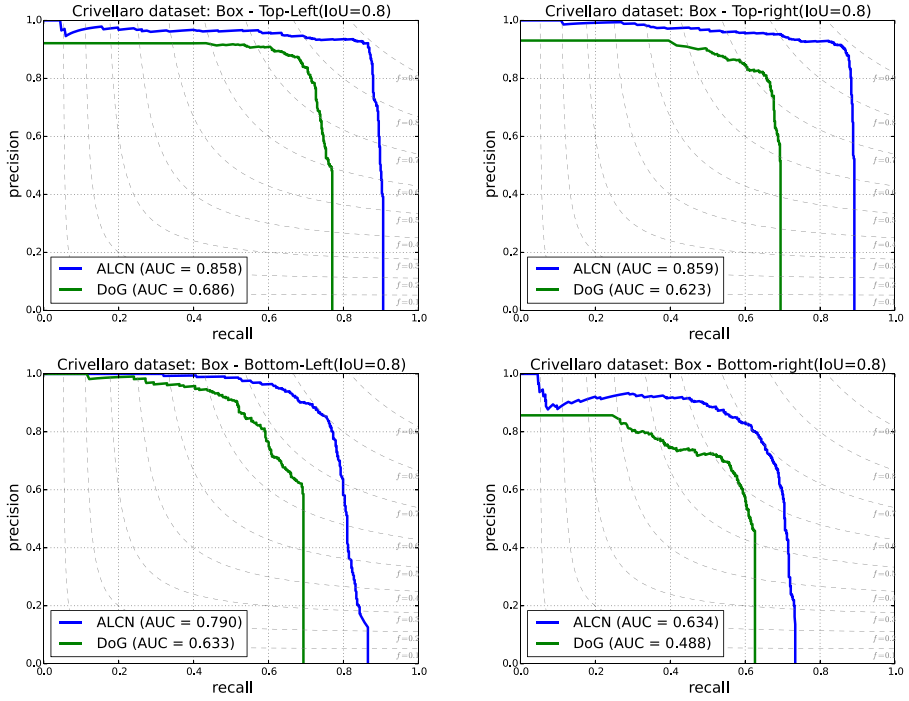

Fig. 13. Comparing ALCN and DoG on the BOX dataset — Video #3 from Crivellaro (Crivellaro et al., 2015). Our ALCN performs best at detecting the corners of the box.

Fig. 14. First row: Original color images from the ALCN-Duck dataset. Second row: Normalized color images after adding the ab channels of the images on the first row to the normalized L channel of the images by our method.

Table 1
Percentage of correctly estimated poses using the 2D Projection metric of Brachmann et al. (2016), when the method of Rad and Lepetit (2017) is applied to our ALCN-Duck sequences, with and without ALCN. Using VGG – trained to predict the 3D pose – alone is not sufficient when illumination changes. ALCN allows us to retrieve accurate poses.

Sequence	w/o illumination changes	With illumination changes						
	#1	#2	#3	#4	#5	#6	#7	#8
VGG	**100**	47.26	18.33	32.65	0.00	0.00	0.00	0.00
VGG+ALCN	**100**	**77.78**	**60.71**	**70.68**	**64.08**	**51.37**	**76.20**	**50.10**

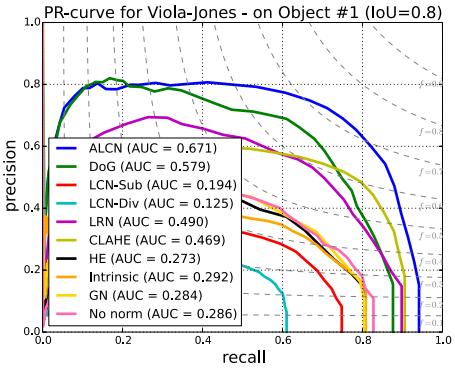

Fig. 15. Evaluating different normalization methods for the Viola-Jones detection method. By simply pre-processing the training and test images using our ALCN, we can improve the performance of Viola-Jones detection with an AUC from 0.286 to 0.671. ALCN outperforms all the other normalization methods.

Table 1 gives the percentage of correctly estimated poses using the 2D Projection metric (Brachmann et al., 2016) with and without our ALCN normalization. Rad and Lepetit (2017), with and without ALCN, performs very well on video sequence #1, which has no illumination changes. It performs much worse when ALCN is not used on Sequences #2, #3 and #4, where the illuminations are slightly different from training. For the other sequences, which have much more challenging lightening conditions, it dramatically fails to recover the object poses. This shows that ALCN can provide illumination invariance at a level to which deep networks such as VGG cannot. Some qualitative results are shown on the last row of Fig. 1.

4.3.2. Application to the Viola-Jones detector

In this experiment, we evaluate the performance of the Viola-Jones detection algorithm trained with a training set created from 10 real images, and normalized using different methods.

Fig. 15 shows that Viola-Jones (Viola and Jones, 2004) performs very poorly with an AUC of 0.286 in best cases. However, by simply normalizing the training and test images using our ALCN, Viola-Jones suddenly performs significantly better with an AUC of 0.671, while it still does not perform very well with other normalization methods. It may be surprising that Viola-Jones needs image normalization at all, as the Haar cascade image features it relies on are very robust to light changes. However, robustness comes at the price of low discriminative power. With image normalization, the features do not have to be as robust as they must be without it.

4.3.3. Application to face recognition

Finally we evaluate our normalization for face recognition to see if our normalization can improve the performance of current recognition algorithms. Hence, we test our normalization on YaleBExt (Georghiades et al., 2001) using Eigenfaces (Turk and Pentland, 1991) and Fisherfaces (Belhumeur et al., 1997), where both perform poorly with different normalization methods. Han et al. (2013) studied 13 different normalizations on face recognition. The best recognition rates among the 13 normalizations are 59.3% and 78.0% vs. 70.5% and 98.6% using our normalization, with Eigenfaces and Fisherfaces respectively.

5. Conclusion

We proposed an efficient approach to illumination normalization, which improves robustness to light changes for object detection and 3D pose estimation methods without requiring many training images. We have shown that our proposed method can bring the power of Deep Learning to applications for which large quantities of training data are not available, since it can be plugged easily to other applications.

CRediT authorship contribution statement

Mahdi Rad: Conceptualization, Methodology, Software, Writing - review & editing. **Peter M. Roth:** Writing - review & editing. **Vincent Lepetit:** Supervision, Writing - review & editing.

Acknowledgment

This work was supported by the Christian Doppler Laboratory for Semantic 3D Computer Vision, funded in part by Qualcomm Inc, United States.

References

Badrinarayanan, V., Kendall, A., Cipolla, R., 2015. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. in: arXiv Preprint.
Belhumeur, P.N., Hespanha, J., Kriegman, D.J., 1997. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. IEEE Trans. Pattern Anal. Mach. Intell. 19, 711–720.
Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y., 2010. Theano: A CPU and GPU math expression compiler. In: Python for Scientific Computing Conference.
Bi, S., Han, X., Yu, Y., 2015. An l 1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. ACM Trans. Graph. 34 (78).
Brachmann, E., Michel, F., Krull, A., Yang, M.M., Gumhold, S., Rother, C., 2016. Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In: Conference on Computer Vision and Pattern Recognition.
Crivellaro, A., Rad, M., Verdie, Y., Yi, K., Fua, P., Lepetit, V., 2015. A novel representation of parts for accurate 3D object detection and tracking in monocular images. In: International Conference on Computer Vision.
Dayan, P., Abbott, L., 2005. Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems. Triliteral.
Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. ImageNet: A large-scale hierarchical image database. In: Conference on Computer Vision and Pattern Recognition.
Fan, Q., Yang, J., Hua, G., Chen, B., Wipf, D., 2018. Revisiting deep intrinsic image decompositions. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J., 2001. From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intell..
Geusebroek, J.M., Burghouts, G.J., Smeulders, A.W.M., 2005. The amsterdam library of object images. Int. J. Comput. Vis. 61, 103–112.
Giusti, A., Ciresan, D.C., Masci, J., Gambardella, L.M., Schmidhuber, J., 2013. Fast image scanning with deep max-pooling convolutional neural networks. In: International Conference on Image Processing.
Goodfellow, I.J., Warde-Farkey, D., Mira, M., Courville, A., Bengio, Y., 2013. Maxout networks. J. Mach. Learn. Res..
Gross, R., Matthews, I., Cohn, J., Kanade, T., Baker, S., 2009. Multi-pie. Image Vis. Comput. 28, 807–813.
Han, H., Shan, S., Chen, X., Gao, W., 2013. A comparative study on illumination preprocessing in face recognition. Pattern Recognit. 46, 1691–1699.
He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Conference on Computer Vision and Pattern Recognition.

Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., Lepetit, V., 2012. Gradient response maps for real-time detection of textureless objects. IEEE Trans. Pattern Anal. Mach. Intell. 34, 876–888.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., LeCun, Y., 2009. What is the best multi-stage architecture for object recognition? In: Conference on Computer Vision and Pattern Recognition.

Krizhevsky, A., Sutskever, I., Hinton, G., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 2278–2324.

LeCun, Y., Huang, F.J., Bottou, L., 2004. Learning methods for generic object recognition with invariance to pose and lighting. In: Conference on Computer Vision and Pattern Recognition.

Levi, K., Weiss, Y., 2004. Learning object detection from a small number of examples: the importance of good features. In: Conference on Computer Vision and Pattern Recognition.

Lowe, D., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60, 91–110.

Lu, R., Koenderink, J.J., Kappers, A.M., 1998. Optical properties (bidirectional reflection distribution functions) of velvet. Appl. Opt. 37.

Nestmeyer, T., Gehler, P.V., 2017. Reflectance adaptive filtering improves intrinsic image estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6789–6798.

Ojala, T., Pietikäinen, M., Mäenpää, T., 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. 24, 971–987.

Pizer, S., Amburn, E., Austin, J., Cromartie, R., Geselowitz, A., Greer, T., Romeny, B., Zimmerman, J., 1987. Adaptive histogram equalization and its variations. Comput. Vis. Graph. Image Process. 39, 355–368.

Rad, M., Lepetit, V., 2017. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In: International Conference on Computer Vision. (accepted).

Rad, M., Roth, P.M., Lepetit, V., 2017. Alcn: Adaptive local contrast normalization for robust object detection and 3d pose estimation. In: British Machine Vision Conference.

Rigamonti, R., Brown, M., Lepetit, V., 2011. Are sparse representations really relevant for image classification? In: Conference on Computer Vision and Pattern Recognition.

Shen, J., Yang, X., Jia, Y., Li, X., 2011a. Intrinsic images using optimization. In: Conference on Computer Vision and Pattern Recognition.

Shen, J., Yang, X., Li, X., Jia, Y., 2013. Intrinsic image decomposition using optimization and user scribbles. IEEE Trans. Cybern. 43, 425–436.

Shen, L., Yeo, C., Hua, B.S., 2011b. Intrinsic image decomposition using a sparse representation of reflectance. IEEE Trans. Pattern Anal. Mach. Intell. 35, 2904–2915.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: International Conference for Learning Representations.

Turk, M., Pentland, A., 1991. Eigenfaces for recognition. J. Cogn. Neurosci. 3, 71–86.

Viola, P., Jones, M., 2004. Robust real-time face detection. Int. J. Comput. Vis. 57, 137–154.

Vonikakis, V., Chrysostomou, D., Kouskouridas, R., Gasteratos, A., 2013. A biologically inspired scale-space for illumination invariant feature detection. Meas. Sci. Technol. 24, 074024.

Wang, H., Li, S.Z., Wang, Y., 2004. Face recognition under varying lighting conditions using self quotient image. In: Automated Face and Gesture Recognition.

Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K., 2017. Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1492–1500.

Yang, B., Yan, J., Lei, Z., Li, S.Z., 2015a. Convolutional channel features. In: International Conference on Computer Vision.

Yang, B., Yan, J., Lei, Z., Li, S.Z., 2015b.. Local convolutional features with unsupervised training for image retrieval. In: International Conference on Computer Vision.

Zhang, R., Isola, P., Efros, A.A., 2016. Colorful image colorization. CoRR abs/1603.08511.

Zhou, T., Krahenbuhl, P., Efros, A., 2015. Learning data-driven reflectance priors for intrinsic image decomposition. In: International Conference on Computer Vision.