# Light Invariant Video Imaging for Improved Performance of Convolution Neural Networks

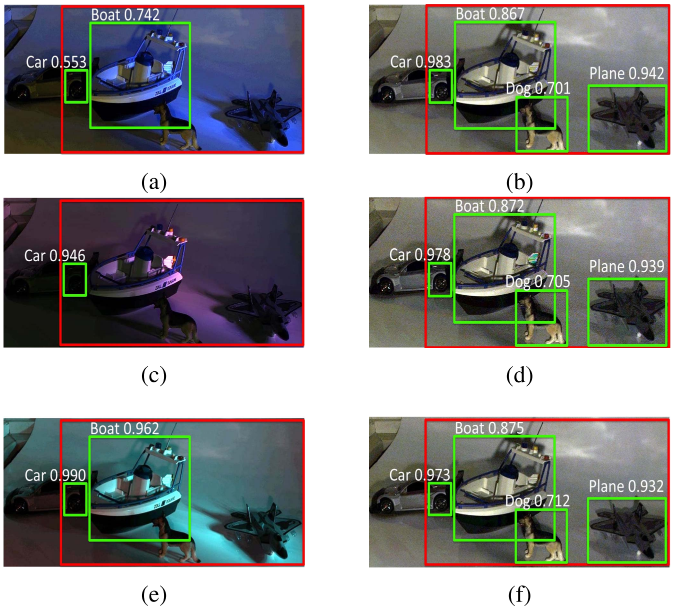
Amir Kolaman, Dan Malowany, Rami R. Hagege, and Hugo Guterman

***Abstract*—Light conditions affect the performance of computer vision algorithms by creating spatial changes in color and intensity across a scene. Convolutional neural networks (CNNs) use color components of the input image and, as a result, are sensitive to ambient light conditions. This work analyzes the influence of ambient light conditions on CNN classifiers. We suggest a method for boosting the performance of CNN-based object detection and classification algorithms by using light invariant video imaging (LIVI). LIVI neutralizes the influence of ambient light conditions and renders the perceived object's appearance independent of the light conditions. Training sets consist mainly, if not only, of objects in natural light conditions. As such, using LIVI boosts CNN performance by matching object appearance to that expected by the CNN model, which was created according to the training set. We further investigate the use of LIVI as a general self-supervised learning framework for CNN. Faster region-based CNN (Faster R-CNN) was used as a case study in order to validate the importance of light conditions on CNN performance and on how it can be improved by using LIVI as an input or feedback mechanism in a self-supervised framework. We show that LIVI enables reduced CNN size, enhanced performance and improved training.**

***Index Terms*—Computational cameras, convolutional neural networks, light invariant imaging.**

## I. Introduction

An ACCURATE and robust object detection and classification algorithm is considered the holy grail of the computer vision community. Researchers developed Convolutional Neural Networks (CNNs) with the aim of reaching human-like object detection and classification performance and improved robustness to image degradations. In recent years, CNNs have enjoyed a boost in performance, and hence popularity, thanks to the publication of databases such as ImageNet [1] and Coco [2], advances in high performance computing and improvements in open-source training models. Such factors have revolutionized the computer-vision field by achieving dramatic progress in the detection [3]–[5] and classification [6]–[8] of objects.



(a) (b) (c) (d) (e) (f)

Fig. 1. Faster R-CNN detection and classification on LIVI camera vs. standard camera. Object in LIVI images gets higher detection (green rectangles) and better classification rate (category name and rate number are shown at top left of each rectangle). (a) Blue light - standard camera. (b) Blue light - LIVI. (c) Purple light - standard camera. (d) Purple light - LIVI. (e) Green light - standard camera. (f) Green light - LIVI.

An object's appearance is affected by the light conditions at the scene. Light reaching the object can vary in color and pattern. Light variations, along with the object's reflectance, determine its appearance. Thus, the same object under different light conditions will have different appearances. Consequently, an observer might find it difficult to recognize an object under non-uniform colored light.

It is suspected that Computer vision CNNs, such as Faster Region-based Convolutional Neural Networks (Faster R-CNNs), will face the same challenges as human observers and achieve lower detection [9] and classification rates under changing light conditions (Fig. 1). This happens because all the color components of the camera form the input source for CNNs, and as such they are sensitive to changes in the object's appearance under changing light conditions. Faster R-CNN performance may thus be degraded by a non-supervised light source because such a source may create color, shadow and contrast changes in the object. As a result, when deploying a computer vision system, the computer vision community

Manuscript received February 13, 2018; revised May 31, 2018; accepted June 6, 2018. Date of publication June 12, 2018; date of current version June 4, 2019. This paper was recommended by Associate Editor C. Regazzoni. *(Corresponding author: Amir Kolaman.)*

A. Kolaman, D. Malowany, and H. Guterman are with the Laboratory of Autonomous Robotics, Electrical and Computer Engineering Department, Ben-Gurion University of the Negev, Beer-Sheva 8410501, Israel (e-mail: kolaman@post.bgu.ac.il; dan.malowany@gmail.com; hugo@ee.bgu.ac.il).

R. R. Hagege is with Lifetobot, Tel Aviv, 6296131, Israel (e-mail: rami.r.hagege@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2018.2846695

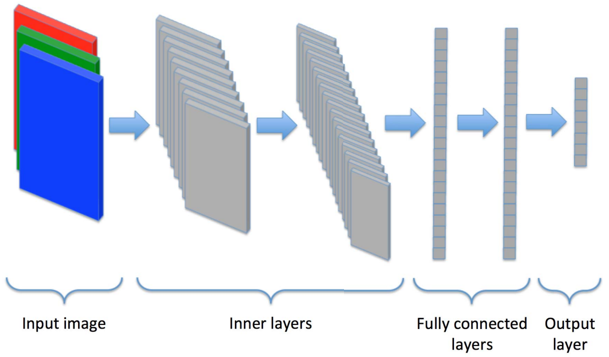


Fig. 2. The structure of a simple CNN. The input layer has three matrices, one for each color channel. The output layer is a vector whose length is defined by the number of classes.

prefers to keep object appearance constant by controlling light conditions. Unfortunately, controlling the light conditions is sometimes impractical, especially for battery powered mobile devices such as smartphones, mobile computers and outdoor video cameras. Such limitations created a need to analyze and understand the effect that different light conditions have on the performance of CNNs.

The main contributions of this work are:

1) Analyze CNNs' [1] sensitivity to changes in light conditions (section II).
2) Suggest Light Invariant Video Imaging (LIVI) as a means of eliminatingCNN sensitivity to light variations (section III).
3) Use LIVI as part of a self-supervised learning framework to improve CNN performance (section IV).

## II. Color Importance in Convolutional Neural Networks

Color is an important factor in deep neural networks. It is typically represented by three-dimensional tensor input on the implementation side. Two dimensions are the spatial width and height of the image and the third dimension is the color channels (commonly three color channels: Red, Green and Blue - RGB). The CNN itself is built of a sequence of layers that transforms one volume to another volume, where the input layer is the color image and the output layer is a vector whose length is the number of classes (Fig. 2). In general, the three main types of layers in CNN are: Convolutional Layers, Pooling Layers and Fully-Connected Layers [6]. As one of the dimensions of the input layer is the color space, it is only natural that the color space will influence the way in which these layers process the image.

The convolutional layer is the core building block of a CNN. It consists of a set of learnable kernels, which are defined during the training process of the CNN. For an input image pixel $Im_{i,j}$, a specific convolution kernel $k$ and kernel size $R \times C$, the output pixel $O_{i,j}$ can be written as:

$$O_{i,j} = \sum_{m=1}^{R}\sum_{n=1}^{C} Im_{i,j} \cdot k_{m,n}, \tag{1}$$

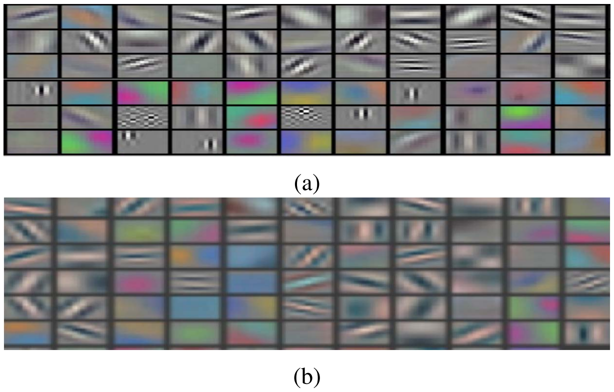


Fig. 3. Examples for first-layer kernels. (a) First-layer kernels from Krizhevsky *et al.* [1] 2012 . (b) First-layer kernels from Zeiler and Fergus [6] 2014.

where $i$, $j$, $m, n$, and $R, C$ are the image and kernel indexes and kernel dimension respectively. CNNs convolve each kernel $k$ across the width and height of the input image $Im$ and compute the dot product at each pixel $Im_{i,j}$. The convolution process produces a two-dimensional activation map that gives the responses of that filter at every spatial position $O_{i,j}$. It is evident that the kernels $k$ of the first convolutional layers $O$ are key elements in the CNN classification and detection process, as they are the first to interact with the input image $Im$. Examining CNN kernels reveals the basic features that influence CNN operation. First-layer kernels in various CNNs (Fig. 3) construct a composition of oriented edges, color blobs and color edges at different frequencies and orientations. Many researchers use this inherent CNN feature to process color information for various tasks such as the automatic coloration of gray-scale images [10], [11] and color classification [12]. CNN sensitivity to color contributes to their exceptional performance, but constitutes a flaw under light variations.

## III. Light Variations in Convolutional Neural Networks

Light variations can lead to changes in color and also create shadow in different patterns. Black-and-white CNN kernels (Fig. 3) represent light intensity variations of the viewed objects. Object appearance is affected by shadows and color cast created from direct light. Many large image data bases [2] are free of shadows and consist of objects under optimum white, bright and homogenous light conditions.

CNN sensitivity to light variation can be measured by comparing its output for two input images: One under constant light conditions and another under changing lights. It is important to note that noise was demonstrated [13], [14] to have an impact on CNN performance. One can minimize noise variability by using two images with the same noise level. Theoretically, this can be achieved by capturing two images at the exact same time with the same camera. We suggest a solution for the above impractical conditions, our solution being to produce two images – constant light and varying light – as two processing outputs for the same set of input frames.

[1] Detection and classification.

Such a solution can be implemented by using light invariant imaging.

### A. Light Invariant Imaging (LII)

Light invariant imaging (LII) creates images that are unaffected by light conditions and can be divided into passive and active solutions. Passive LII analyze existing light conditions, while active LII adds a dedicated light source to the scene. The most well-known passive LII is the color correction (white balance) stage in cameras. The best example of an active solution is the use of camera flash in low light conditions. Both methods aim to make the object appear the same under various light conditions.

Passive LII identifies the current properties of the background lights and compensates for their effect by assuming a pre-known behavior of light and objects in the scene. Examples include various White Balance (WB) (color constancy) algorithms [15] and illumination invariant imaging [16]. They assume the scene light to be one of the pre-known types. Such assumptions often break down and limit these solutions. Neural nets try to be illumination invariant by training on large data sets [17]. But data sets are limited in the size and variability of light conditions. A scene with several spatially varying light sources would challenge most passive methods and reduce their performance considerably.

Active solutions add a controlled light source in order to remove or measure the effect of other light sources on the scene. The most common active method is a standard flash light. Flash works well when it dominates the lights in the scene. Flash domination occurs when the contribution of a background light source is less than the camera's capability to measure it. Each camera signal $X$ captures the reflected flash and background lights. Therefore camera signal $X$ is comprised of $X_f$, the signal affected by the flash light, $X_b$, the signal affected by the background lights and $Z$, the noise [18].[2] When a flash light $L_f$ becomes much greater than the background light $L_b$, i.e. $L_f \gg L_b$, then the background signal is smaller than the noise levels $X_b < Z$. In this case the background light has no effect on the camera signal. This results in the flash dominating the camera signal $X = X_f + Z$, hence leading to light invariant conditions. For example, a camera with noise levels of 1% of the camera signal, $Z = 0.01X$, would require the flash light intensity relation with the background lights to be $L_b = \frac{L_f}{100}$. Such intensity requirement has made the most common time for using flash to be at night, when the intensity of the background light conditions is very low. In all the other cases, i.e. $L_b \approx L_f$ or $L_b > L_f$, the standard flash effectiveness for light invariance drops considerably.

Flash/no-flash [19] overcomes the limitations of the standard flash and achieves controlled light conditions even when the background lights are of the same magnitude as the flash. It performs biasing of the background light by subtracting the image with the flash turned on from the image with the flash turned off. Applications for flash/no-flash include improving white balance [20], shadow removal [21], foreground extraction [22], deblurring [23] and photometric stereo [24]. Flash/no-flash was used to improve saliency detection in [25]. Saliency detection can enhance object detection and classification. However, to the best of our knowledge, we could not find other scientific work investigating the connection between light invariancy (flash/no-flash) and object detection.

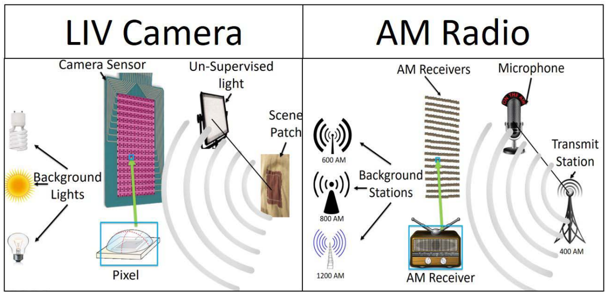


Fig. 4. LIVI vs. AM Radio.

Flash/no-flash requires precise synchronization between camera and flash, assuming there is no change in the scene between the captured frames. Such an assumption is often voided by oscillating lights in the scene, which are connected to the electric grid [26]. Light such as incandescent, fluorescent and LED oscillates in time and creates an inconsistent flash/no-flash output. This renders the flash/no-flash method inadequate for precise measurements, which require consistency. We therefore need a better alternative, which will be presented next.

### B. Light Invariant Video Imaging (LIVI)

Light Invariant Video Imaging (LIVI) [27], [28] uses a modulated light to separate itself from the dynamic background lights. This solution was inspired by AM radio, where the radio waves modulate sound and separate themselves from one another by their frequencies. A receiver tuned to a specific frequency would play a station and attenuate all the other ones (Fig. 4). Similar to radio LIVI has no need to synchronize between the light source and camera. This simplifies the system considerably and enables its usage in video. Video imaging uses alight source to modulate an object's appearance. Changing thelight intensity in time as a sine wave with a constant frequency creates an object appearance channel. An image pixel tuned to this frequency will show the influence of the modulated light and attenuate all the other ones. It sets the appearance of the object without shadow and with constant color over time.

Light sources hit the object, and then reflect back to the camera. The reflection is directly connected to the light sources in the scene. Light sources can be divided into two groups: background sources, which have an unknown behavior in time and space and a modulated light source $L_m(f_1, t)$, described by:

$$L_m(f_1, t) = a_0 + a_1 cos(2\pi f_1 t), \tag{2}$$

where $t$ represents time, $a_0$ is the constant intensity over time and $a_1$ is the amplitude of the main harmonic oscillating at $f_1 = \frac{1}{T_1}$.

[2] This is comprised of signal-dependent and signal-independent noise.

Total light in the scene is reflected by the object and generates a radiance $I(t)$ equal to:

$$I(t) = C + A_1 \cdot cos(2\pi f_1 t) + I_b(t), \tag{3}$$

where $C$ depends on the patch reflectance and constant part of all the lights (modulated and background), radiance coefficient $A_1$ depends on the patch reflectance and intensity amplitude $a_1$ from Eq. (2) and $I_b(t)$ is the reflection from the dynamic background lights $L_b(t)$.

Radiance $I(t)$ is sampled by a camera pixel at discrete times $n \in \{0, 1, ..., N-1\}$:

$$X[n] = C + I_b[nT_s] + A_1 \cdot cos[2\pi f_1 n T_s + \varphi_1] + Z_n, \tag{4}$$

,where $C$ is the measured radiance[3] of the constant part (modulated and background), $I_b[nT_s]$ is the intensity of the dynamic background radiance, $T_s = \frac{1}{f_s}$ is the sample time of the camera (the sample frequency $f_s$ is also referred to as Frames Per Second (FPS)), $A_1$ is the amplitude of the modulated radiance, $cos[2\pi f_1 n T_s + \varphi_1]$ is a discrete sample of $cos(2\pi f_1 t)$, $\varphi_1$ is the unknown phase difference between modulated light and camera and $Z_n$ is a zero mean additive noise with constant spectral intensity of $\mathbb{E}[Z_n^2]$.

The aim of the LIVI system is to reconstruct $A_1$ using pre-known information on the frequency of the modulated light $f_1$. Various methods can accomplish this, one of which is the inner product using a Finite Impulse Response (FIR) filter:

$$\hat{A_1} = \left| \frac{2}{N} \sum_{n=0}^{N-1} X[n] e^{-i2\pi f_1 T_s n)} \right|. \tag{5}$$

The filter will attenuate all the terms in Eq. (4) except for the amplitude of the oscillating part at the target frequency $f_1$, i.e., $A_1$. The purpose of the absolute value is to eliminate the phase term $e^{i\varphi_1}$, which is unknown. LIVI captures $N$ images, with background lights and modulated light, to produce one image without background lights (Fig. 5).

LIVI surpasses other light invariant methods in three ways:

1) Removal of dynamic background lights.
2) Constant level of noise.
3) Maximization of CNN performance.

Flickering light sources – such as fluorescent or tungsten – oscillate at a 100/120Hz. LIVI is capable of dynamically removing background light just as a radio receiver is able to separate one radio station from another.

The noise term of the output was empirically shown in [28] to be a constant multiplicative noise. It was also demonstrated how noise from the reconstructed signal $\hat{A_1}$ had lower noise levels than the standard camera with a more linear behavior. Lower linear noise levels are expected to give better performance in CNN output.

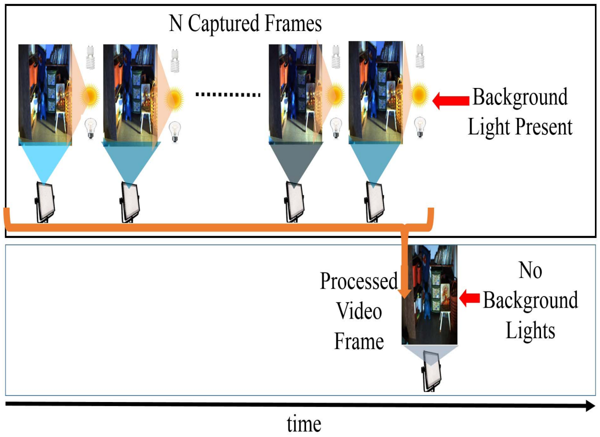
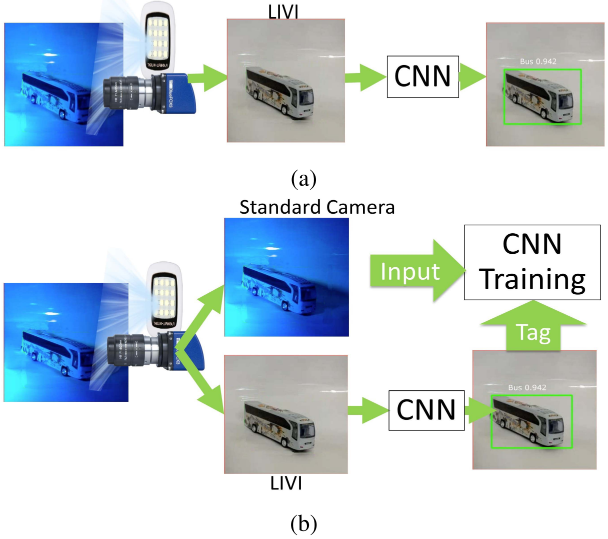


Fig. 5. LIVI analysis of N Images.



(a)

(b)

Fig. 6. Two use cases for LIVI: (a) Input to the CNN, (b) LIVI as an Auto Marker and Tagger for CNN Re-training.

## IV. Improving CNN Performance With LIVI

LIVI can be used to improve CNN performance in challenging scenerios. The most challenging cases include extreme color and shadow variations, such as in 1. CNN results in these cases can be maximized by using one of the following two methods (Fig. 6):

1) LIVI input to CNN: LIVI output is used as the input to the deployed CNN (Fig. 6a).
2) LIVI self-supervised learning: LIVI output is used as a feedback in the retraining process of the deployed CNN (Fig. 6b) .

### A. LIVI as Input to CNN

LIVI input to CNN (Fig. 6a) neutralizes the influence of the changing light conditions and transforms the test data so it can be perceived as if they are under constant normal light conditions. These constant light conditions dominate images used in training sets. Therefore keeping object appearance constant under various light conditions increases the similarity of the extracted features of the test images to the ones in the training set. Similar features give better CNN detection and classification, thus maximizing its performance.

### B. LIVI Self-Supervised Learning

LIVI self-supervised learning (Fig. 6b) uses LIVI as feedback in the CNN training process, thus maximizing

[3] Cameras can measure radiance by normalizing its measurements with Exposure Value(EV).

CNN performance for standard cameras. The standard camera is sensitive to changes in light conditions, which leads to lower success rates. Typically, to avoid these fail cases, the training process will include gathering huge amounts of data in different locations and light conditions, which hopefully will include the weak spots of the trained model. This data will need to undergo the tedious task of full manual labeling by a human user, who tags all the objects correctly and feeds them into the CNN retraining phase. This supervised learning by a human is limited by the fact that generating manual ground truth is laborious and tedious and that solving the huge imbalance between the 'normal' light conditions and the a-priori unknown problematic light conditions is extremely difficult. A faster and more efficient alternative would be self-supervised learning [29].

Our proposed self-supervised scheme uses results from LIVI-CNN as a feedback mechanism in the training process. This leverages the claims from sub-section IV-A that, under challenging light conditions, LIVI-CNN results are better than the standard camera. These superior results can be used as a feedback mechanism in the retraining process.

Consider two CNN outputs: 1. from LIVI input ($LiviCNN$) and 2. from standard camera input ($StndrdCNN$). A mismatch score $Ms$ can be given for each pair according to:

$$Ms(LiviCNN, StndrdCNN) = \frac{1}{N}\sum_{i=1}^{N} |LiviCNN_{ObjScore_i} - StndrdCNNObjScore_i|, \quad (6)$$

where $LiviCNN_{ObjScore_i}$ represents the $i$'s object score for CNN output from LIVI input and $StndrdCNN_{ObjScore_i}$ represents the $i$'s object score for CNN output from standard camera input. The above was used as a mismatch measurement to choose frames for the retraining process. Significant mismatch between two measurements was defined as a relative high score i.e. $Ms$ (Eq. 6). In the case of such mismatch, the standard frame was marked for retraining (Fig. 6). [4]

The retraining stage uses CNN LIVI results as an auto tag tool. The tool tags each object according to the CNN-LIVI detection and classification. This self auto-tagging, which replaces human tagging, is faster, more efficient and more robust. Therefore, our proposed CNN-LIVI self-supervised learning framework should accelerate and simplify the CNN retraining process.

## V. Experiments

In this section three contribution from section I are verified:

1) Color is more beneficial to CNN object classification and detection than gray-scale images.
2) Light conditions change the performance of object classification because of the response in the hidden layers of CNN.
3) LIVI can be used to improve CNN performance in a self-supervised learning framework.

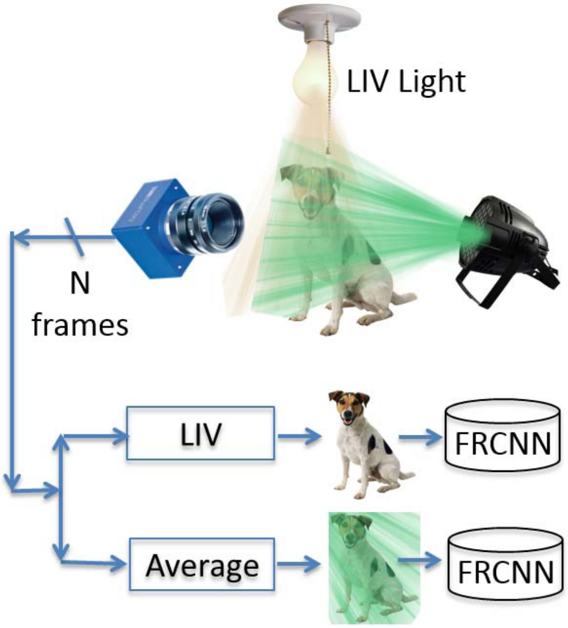


Fig. 7. Test scheme.

### A. Experimental Setup

A data set with two hundred and ten images was generated (available online [30]) and used in all of the experiments. The set presents large variations of light conditions. These variations include fifteen different locations and seven different colors of light. Two images were created for each condition: a standard image and a LIVI image. The standard image was generated by averaging $N$ input video frames. The LIVI image was generated by using the same $N$ input frames with Eq. 5. Each image in our data had five objects, giving more than seven hundred test cases. The data set was used to check the effect of light conditions on the performance of CNN detection and classification. The images were fed into a Faster R-CNN net which was trained on a Pascal VOC 2007 data set [31]. The data set contains twenty categories of objects and animals, only five of which were used in our experiment for the sake of simplicity: “airplane”, “dog”, “boat”, “bus” and “car” (Fig. 1).

A schematic description of the experiments with the proposed system can be seen in Fig. 7. The camera, modulated light source and background light were all pointed toward the objects. The background light source direction, intensity and color were changed over time, while modulated light and camera were kept in place. The camera output was connected to a computer, which created two image outputs: the standard image and the LIVI image. Both images were then fed to the CNN algorithm to detect and classify the objects. An additional reference image with flash only and no background light was captured for each scene setup. The Faster R-CNN [32] framework was chosen due to its popularity and combined detection and classification.[5] The classification stage

[4] Sub-figure (b) does not show the full explanation, as the standard image is also used as an input to the CNN. The difference in the two outputs (one with LIVI image and one with standard image) is analyzed. Most informative cases (biggest difference in output) are kept for the self-supervised images selection algorithm.

[5] This is a representative example, but we are sure that the same results can be seen in other CNN architectures.

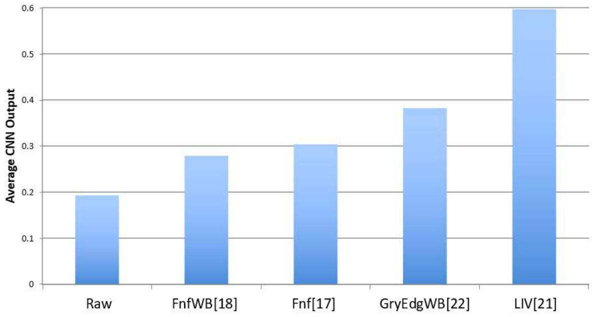


Fig. 8. Light Invariant Method Comparison. The Y-axis in the graph is the detection rate of the Faster R-CNN with matching light invariant methodologies.

uses a VGG "CNN M 1024" model (similar to AlexNet) [33] and a VGG-16 "Net D" model [8]. The mModels were pre-trained on the VOC 2007 [31] data-set. The results of the CNN under the different background light conditions (different directions and colors) were collected and analyzed.

LIVI was the best input to CNN when comparing Faster R-CNN outputs with different inputs. Five Faster R-CNN outputs were compared with matching processed camera inputs: raw data, WB using gray edge [34], flash/no flash [19], WB using flash/no-flash [20] and LIVI. The Faster R-CNN detection rate was measured as the proportion of objects that were detected, classified and located correctly.

Detection matric measured the detection rate of the Faster R-CNN. Mean Average Precision (mAP) was considered as an object detection metric due to it popularity. This is a very efficient way to quantify a model using one single number. Nevertheless, mAP measures both localization and classification and is very sensitive to the bounding box size and location resulted from the regression process. In this paper, we focus on analyzing the effect of light conditions on the activation of the different layers of the model and, as a result, on object recognition. We did not want the influence of the RPN and bounding box regression to interfere or smooth the results in any way that would interfere with our analysis. In addition, mAP is a measure that is averaged on all classes and tends to be less affected by changes in a small subset of classes (we produced change only in 5 classes out of 20). Also, the mAP is averaged on the entire precision-recall curve. As a result, it is pretty smooth, and small changes at effective working points (high f-score) might get averaged and be hard to emphasize when comparing the different results. A threshold based metric was chosen instead of mAP. A detection is considered positive if the classification score was above a 0.5 threshold and bounding box intersection of union (IOU) with ground truth bounding box was detected above a 0.3 threshold.

LIVI's output invariability, which resulted in superior CNN performance (Fig. 8), led to its use in this paper.

### B. Importance of Color in CNN

The first part of the experiment attempted to validate the importance of color components in CNN. The more kernels of color blobs and edges (explained in section II) that are included in the first-layers of the CNN, the more CNN classification degrades its performance when there is no color in the image. In order to test this, it was decided to review CNN response to color images and gray-scale images of the same scene. A set of color images was taken under different color light conditions (Fig. 1). The two sets of color images were then converted to gray-scale using Matlab's rgb2gray function, thus creating a color set and a gray set.[6]

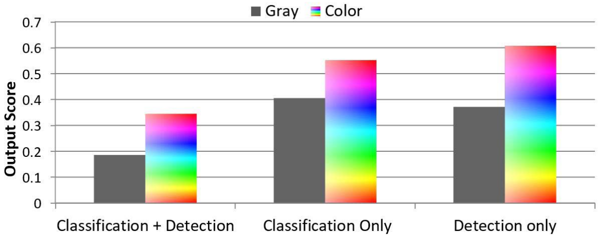


Fig. 9. Comparison between gray-scale and color images for detection+classification, classification and detection.

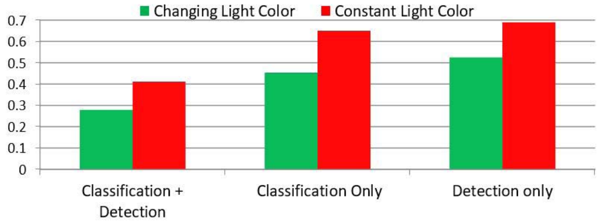


Fig. 10. Comparison between varying and constant light conditions for detection+classification, classification and detection.

CNN performance with gray scale and color images input was tested for the following: detection and classification (Fig. 9, left columns), detection rate only (Fig. 9, middle columns) and classification score only (Fig. 9, right columns). Differences in scores between gray and color images should give an indication of how important color is to CNN detection and classification. The results (Fig. 9) show that color has an important role in the different layers of Faster R-CNN, leading to better performance.

### C. Effect of Changing Light Conditions on CNN

The second part of the experiment measured the performance of the Faster R-CNN under changing light conditions. The degree of influence that varying light has on identification, classification and their combination was examined (Fig. 10). Classification was separated in Faster R-CNN by canceling the detection path and manually feeding the object's location. The detection score was extracted from the detection stage. The Intersection Of Union (IOU) detection cutoff value was set to 0.7. The confidence level cutoff value was set in this test

[6] These color images were transformed into gray-scale images. The creation of the gray-scale images out of the RGB images was done by eliminating the hue and saturation information while retaining the luminance. The resulting gray-scale image is a stack of three identical $R_{gray}$,$G_{gray}$,$B_{gray}$ matrices calculated by: $R_{gray} = G_{gray} = B_{gray} = 0.2989 * R_{color} + 0.5870 * G_{color} + 0.1140 * B_{color}$.

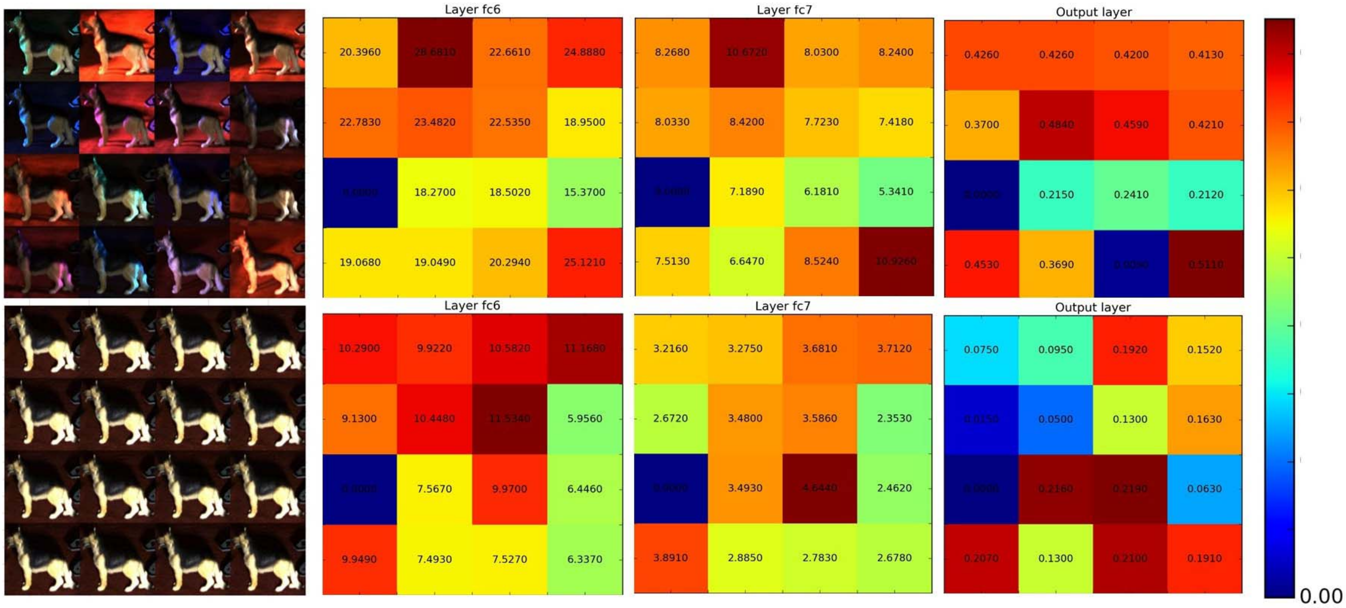
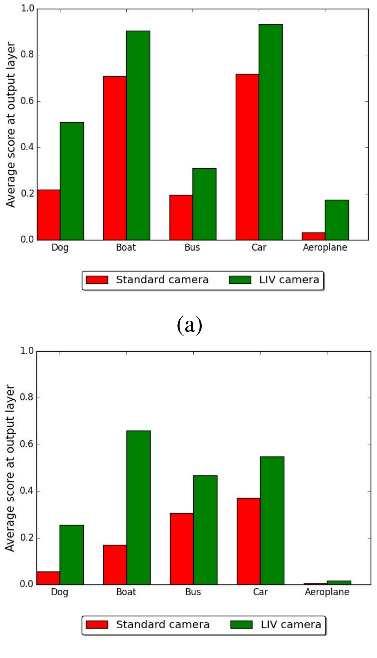


Fig. 11. Distance between "dog" images captured by a standard camera and LIVI. The distance map between images and the reference (marked in blue) is given in color and number. Distance is measured in feature space of "fc6", "fc7" and output layers.

to 0.5. Faster R-CNN governs the detection process by filtering out detections with classification probability lower than that value. The higher this parameter is set to, the lower the false alarm rate that Faster R-CNN will have and the lower the detection rate will be. Hence the goal is to set this parameter to the highest value possible while still keeping an acceptable detection rate. Coupling of the Faster R-CNN score and its detection rate level can be seen in Fig. 13. Two cases were tested: changing light conditions (blue curve, Fig. 13) and constant light conditions (green curve, Fig. 13). In addition, VGG16 and VGG1024 dependency on varying light conditions were tested for each category (Fig. 12). All of the above measurements should give an indication of whether the CNN was sensitive to changing light conditions and to what degree.

The inner layers of the CNN models were analyzed in order to better understand the connection between changing light conditions and their activation. The Faster R-CNN detection path was cancelled and the location of the objects was manually fed. Such fixation removed the detection stage dependency of the classification models.

CNN activity of inner layers "fc6", "fc7", "conv5" act as a feature vector representation of the image. These together with the output layer were measured in feature distance for the "dog" category (Fig. 11). A bounding box around the dogs was manually chosen and feature distance was extracted. The above figure shows that the distance in feature space between images under different light conditions is substantially smaller using LIVI than using a standard camera. Principal component analysis (PCA) of the information behavior in "fc6" while running the Faster R-CNN on all objects and in all light conditions shows less activation (Fig. 14). Analysis of the inner layers gives an indication of the extent that light conditions have on the CNN object classification stage.



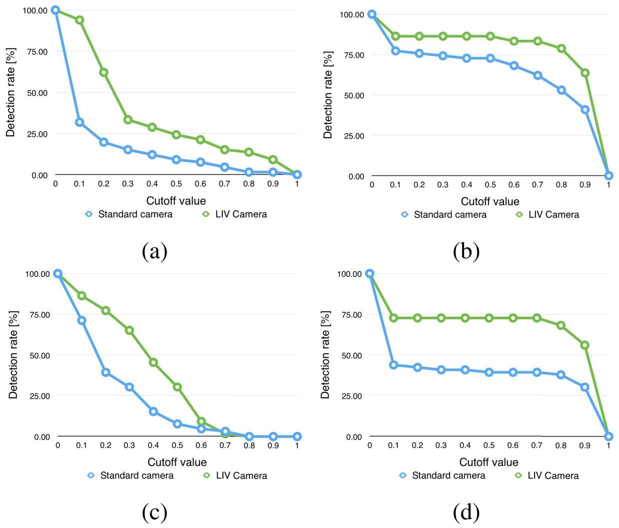
(a)

(b)

Fig. 12. Comparison between standard camera and LIVI for (a) VGG 16 and (b) VGG 1024 models.

### D. LIVI Self-Supervised Learning

The third part of the experiment used the standard pre-trained Faster RCNN VGG1024 [32], trained on the VOC2007 data set [31], to test the potential improvement when using CNN on LIVI as a feedback input (Fig. 15). The model was fine-tuned by freezing the convolutional layers and training only the RPN layers and fully connected layers on VOC2007 mixed with our data. To avoid over-fitting to our



(a)

(b)

(c)

(d)

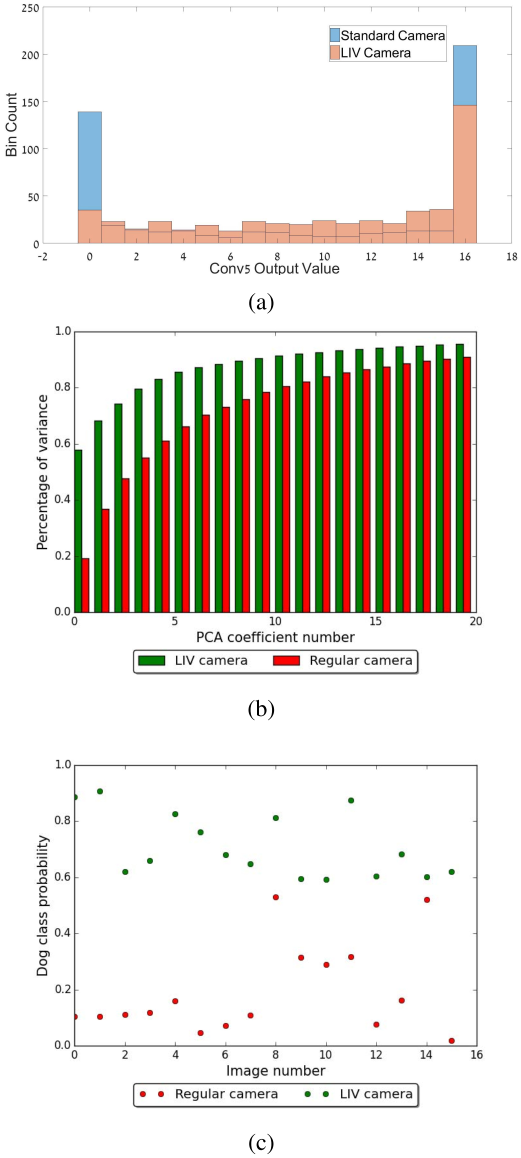
Fig. 13. Detection rates for different classes using faster R-CNN under changing light conditions. (a) "Airplane" detection rate. (b) "Boat" detection rate. (c) "Bus" detection rate. (d) "Car" detection rate.

data, the mix was 99% VOC and 1% our data. Random augmentations were used for efficient and enriched training on the data set. Affine augmentation (resize, rotation and reflection), as well as visual augmentation (blur, noise and coloring) were used. The model was fine-tuned twice: 1) Standard camera input to CNN in the retraining (Fig. 15a) process. In this case $M = 55$ images, under natural bright white lighting, were tagged manually in order to fine-tune the model. 2) Standard camera input to CNN with LIVI self-supervised retraining (Fig. 15b). The location of the objects and the location of the camera was identical in both cases to avoid any bias. Challenging frames were detected using LIVI. The above method used $M = 55$ images, under changing light conditions (color and location), to auto-mark and tag the images to fine-tune the model.

The pre-trained model was fed simultaneously with standard camera images and LIVI images. The self-supervised learning algorithm ranked the images by their potential to contribute additional information. The potential was calculated by comparing the recognition score with the LIVI camera (objects seem as if they are under bright white light) and the recognition score with the standard camera (objects suffer from changing light conditions). $L = 10$ top mismatched (Eq. 6) image pairs were chosen for the retraining process of the model. The annotations of these images was done by the algorithm, using the LIVI input and the pre-trained model. For comparison and to avoid potential bias, the same $L = 10$ images under bright white light were used also to retrain the model.

The net was retrained on a Pascal VOC2007 mixed data-set. The data-set contained a total of 5011 Pascal VOC 2007 annotated frames (mix of 2501 training + 2510 validation) and our data (10 images). The test was done on the Pascal VOC 2007 test data-set, which had 4952 frames, and our test data, which had 45 frames under various light conditions.

Results can be seen in Fig. 16a, where standard input and LIVI input are shown for comparison on the left against the two above retraining methods: standard training and LIVI as a feedback. Both re-trained models were checked on a VOC2007 data set to see whether an improvement was made compared with the pre-trained model 16b.



(a)

(b)

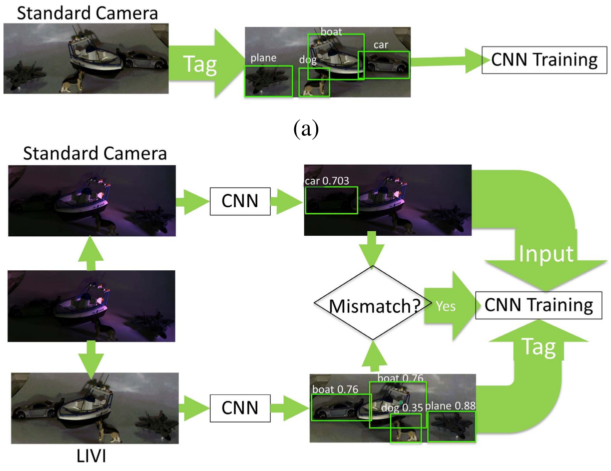
(c)

Fig. 14. Activation comparison between a standard camera and LIVI in the (a) middle layer "conv5_3", (b) fully connected layers "fc6" and (c) output layer.

## VI. Results and Discussion

Color improves detection and classification in Faster R-CNN. All parts of the Faster R-CNN (detection and classification separately, and both of them combined) performed better when the input image was colored rather than gray (Fig. 9). This therefore implies that in general, color information improves the performance of machine vision CNN.

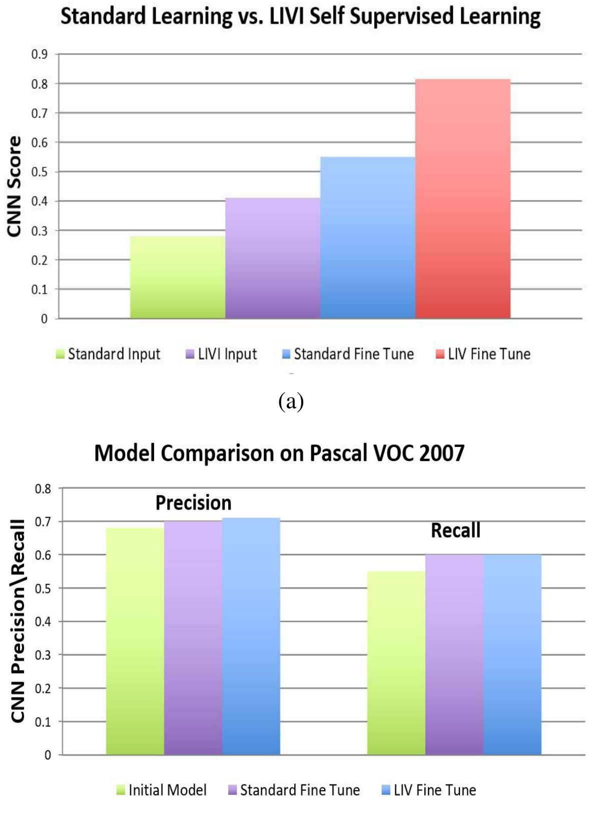
Analysis of Faster R-CNN capabilities under changing light conditions showed interesting results. Detection boxes in the example images (Fig. 1) show that only the car category was detected under all the light condition; the rest were partially detected. The classification confidence score, seen as a number



(a)

(b)

Fig. 15. Proposed CNN retraining with LIVI feedback compared with standard CNN retraining. (a) Fine tuning using a regular camera. (b) Using LIVI for auto-tagger feedback for retraining.



(a)

(b)

Fig. 16. Performance comparison of standard CNN retraining vs. proposed CNN training with LIVI feedback. (a) Retraining comparison of standard method vs. LIVI feedback (pre-tuned standard and LIVI results are shown as a reference). (b) Performance comparison of the re-trained models against the previous on the VOC2007 data set.

at the top left corner of each detection box (Fig. 1), changes between each light condition. An extensive analysis of all the images in our data base shows improved detection and classification rates using LIVI's constant light conditions (Fig. 10). Constant light conditions elevate the detection graph and bring it closer to the ideal case of one hundred percent detection for any cutoff value (Fig. 13). Improvement in the output is independent of the classification model (Fig. 12). This detection/classification divide-and-conquer analysis showed that Faster R-CNN, as a whole, is sensitive to changes in light conditions. Constant light conditions that are even in color and intensity in space enable better matching to convolutional kernels and inner CNN layers. This implies that machine vision CNN in general is also sensitive to changes in light conditions.

Deeper analysis of the impact light conditions have on the performance of Faster R-CNN was acheived by measuring the activation of the inner layers of the CNN. LIVI's input shortens the feature distance between objects in the inner "fc6" and "Fc7" layers and the output layer (Fig. 11). As a result, LIVI evokes more activations of Faster R-CNN inner layers. The mid-layer "conv5_3" analysis (Fig. 14a) shows that LIVI activates more kernels (477 vs. 373) than the standard camera. Higher Faster R-CNN activation with LIVI is also apparent in inner layers "fc6" and "fc7". Principal component analysis (PCA) of the information behavior in inner layers "fc6" and "fc7" in Faster R-CNN show sparser behavior of these layers under constant light conditions (Fig. 14b). Inner layer PCA analysis implies that using LIVI can have a major implication on the size of the net, as recent studies have shown [35], [36]. These results demonstrate how irregular light conditions result in irregular object appearance, which in turn is translated into lower activation of the kernels in the convolution layers. Therefore the inner layers of the above CNN respond more to LIVI input. They are more stable, with a constant behavior for the same objects. These results support the conclusion that LIVI manages to create a near-constant lighting environment for the CNN to process. CNN performance can be enhanced by keeping light conditions similar to the ones used in the training set or by using the LIVI.

Comparing performance of the standard passive retraining method to the active method, using LIVI as a feedback input (Fig. 15), shows improvement. The classical retraining has brought CNN results on the standard camera input very close to CNN results on LIVI input. A further improvement is seen when LIVI was used as to provide feedback in a self-supervised retraining method. This self-supervised learning was fast, due to the auto-tagging and feedback of the CNN on LIVI and robust, due to LIVI's constant light conditions. It showed that the use of LIVI for unsupervised learning can improve the performance of the CNN for camera input under changing light conditions (Fig. 16a. Further tests showed an improvement in the PASCAL VOC 2007 test data-set (Fig. 16b). Using LIVI coupled with CNN improves the retraining process for standard cameras.

## VII. Conclusions and Future Work

The state-of-the-art results with deep neural networks are largely dependent on access to large labeled data sets relevant for the intended task. Currently there are tasks for which large labeled data sets do not exist. It is therefore common practice to take an existing CNN that was pre-trained for one task and use a small labeled data set to re-train it for another task.

A fast and robust retraining, which is immune to changing light conditions, would aid CNN developers.

Our claim that CNN performance degrades under changing light conditions was verified by extensive experiments on different layers of CNN. A new CNN self-supervised training framework was suggested by coupling it with Light Invariant Video Imaging (LIVI). It was shown that this coupling improves CNN performance and enables reducing it in size. This leads to reduced retraining time for light conditions biasing.

The suggested method has many practical applications. Mobile devices would benefit from using our method, because they continuously face changing light conditions. Therefore smaller, faster and light-invariant CNN would be great for mobile devices. LIVI's modulated light can use other wavelengths, such as ultraviolet or infrared. This can benefit CNN by using multi-spectral data [37] to identify and classify objects.

Future work could enable the use of LIVI for foreground/background separation and improve classification performance. Furthermore, it can also give the classifier feedback on the areas in the scene where the light conditions are challenging. These cases would change considerations in the classification process and introduce new CNN architectures.

## REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. 2012, pp. 1097–1105.

[2] T.-Y. Lin *et al.*, "ar, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*. Springer, 2014, pp. 740–755.

[3] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. CVPR*, Jul. 2017, pp. 936–944.

[4] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 354–370.

[5] X. Wang, A. Shrivastava, and A. Gupta, "A-fast-rcnn: Hard positive generation via adversary for object detection," in *Proc. CVPR*, vol. 2, 2017.

[6] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. 13th Eur. Conf. (ECCV)*. Zurich, Switzerland: Springer, vol. 8689, 2014, pp. 818–833.

[7] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. (2013). "OverFeat: Integrated recognition, localization and detection using convolutional networks." [Online]. Available: https://arxiv.org/abs/1312.6229

[8] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[9] J. S. Kulchandani and K. J. Dangarwala, "Moving object detection: Review of recent research trends," in *Proc. IEEE Int. Conf. Pervasive Comput. (ICPC)*, Jan. 2015, pp. 1–5.

[10] T. D. Nguyen, K. Mori, and R. Thawonmas. (2016). "Image colorization using a deep convolutional neural network." [Online]. Available: https://arxiv.org/abs/1604.07904

[11] Z. Cheng, Q. Yang, and B. Sheng. (2016). "Deep colorization." [Online]. Available: https://arxiv.org/abs/1605.00075

[12] R. F. Rachmadi and I. K. E. Purnama. (2015). "Vehicle color recognition using convolutional neural network." [Online]. Available: https://arxiv.org/abs/1510.07391

[13] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 427–436.

[14] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. (2015). "DeepFool: A simple and accurate method to fool deep neural networks." [Online]. Available: https://arxiv.org/abs/1511.04599

[15] A. Gijsenij, T. Gevers, and J. van de Weijer, "Computational color constancy: Survey and experiments," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2475–2489, Sep. 2011.

[16] W. Maddern, A. D. Stewart, C. McManus, B. Upcroft, W. Churchill, and P. Newman, "Illumination invariant imaging: Applications in robust vision-based localisation, mapping and classification for autonomous vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. Vis. Place Recognit. Changing Environ. Workshop (ICRA)*, Hong Kong, China, May 2014, pp. 1–8.

[17] O. Gupta, D. Raviv, and R. Raskar. (2016). "Deep video gesture recognition using illumination invariants." [Online]. Available: https://arxiv.org/abs/1603.06531

[18] S. W. Hasinoff, F. Durand, and W. T. Freeman, "Noise-optimal capture for high dynamic range photography," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 553–560.

[19] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama, "Digital photography with flash and no-flash image pairs," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 664–672, Aug. 2004.

[20] Z. Hui, A. C. Sankaranarayanan, K. Sunkavalli, and S. Hadap, "White balance under mixed illumination using flash photography," in *Proc. IEEE Int. Conf. Comput. Photography (ICCP)*, May 2016, pp. 1–10.

[21] K. Swami, S. K. Das, G. Khandelwal, and A. Vijayvargiya, "A robust flash image shadow detection method and seamless recovery of shadow regions," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2016, pp. 2836–2841.

[22] J. Sun, J. Sun, S. B. Kang, Z.-B. Xu, X. Tang, and H.-Y. Shum, "Flash cut: Foreground extraction with flash and no-flash image pairs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2007, pp. 1–8.

[23] S. Zhuo, D. Guo, and T. Sim, "Robust flash deblurring," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 2440–2447.

[24] Z. Hui, K. Sunkavalli, S. Hadap, and A. C. Sankaranarayanan. (2017). "Illuminant spectra-based source separation using flash photography." [Online]. Available: https://arxiv.org/abs/1704.05564

[25] S. He and R. W. Lau, "Saliency detection with flash and no-flash image pairs," in *Computer Vision—ECCV*. Springer, 2014, pp. 110–124.

[26] M. Sheinin, Y. Y. Schechner, and K. N. Kutulakos, "Computational imaging on the electric grid," in *Proc. IEEE CVPR*, Jul. 2017, pp. 2363–2372.

[27] A. Kolaman, R. Hagege, and H. Guterman, "Light source separation from image sequences of oscillating lights," in *Proc. IEEE 28th Conv. Electr. Electron. Eng. Israel (IEEEI)*, Dec. 2014, pp. 1–5.

[28] A. Kolaman, M. Lvov, R. Hagege, and H. Guterman, "Amplitude modulated video camera—Light separation in dynamic scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3698–3706.

[29] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 3636–3645.

[30] *Livi Data Base*. [Online]. Available: https://www.dropbox.com/sh/6nrn9kyvmx0wztn/AAAZdD8eRsz5xsD_4bGGX9Naa?dl=0

[31] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, *The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results*.[Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html

[32] S. Ren, K. He, R. Girshick, and J. Sun. (2015). "Faster R-CNN: Towards real-time object detection with region proposal networks." [Online]. Available: https://arxiv.org/abs/1506.01497

[33] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. (2014). "Return of the devil in the details: delving deep into convolutional nets." [Online]. Available: https://arxiv.org/abs/1405.3531

[34] A. Gijsenij, T. Gevers, and J. van de Weijer, "Improving color constancy by photometric edge weighting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 5, pp. 918–929, May 2012.

[35] S. Han *et al.*, "Eie: efficient inference engine on compressed deep neural network," in *Proc. 43rd Int. Symp. Comput. Archit.* Piscataway, NJ, USA: IEEE Press, Jun. 2016, pp. 243–254.

[36] J.-H. Luo and J. Wu. (2017). "An entropy-based pruning method for CNN compression." [Online]. Available: https://arxiv.org/abs/1706.05791

[37] J. Liu, S. Zhang, S. Wang, and D. N. Metaxas. (2016). "Multispectral deep neural networks for pedestrian detection." [Online]. Available: https://arxiv.org/abs/1611.02644

**Amir Kolaman** received the B.Sc. and M.Sc. degrees in electrical and computer engineering from Ben-Gurion University of the Negev, Beersheba, Israel, in 2005 and 2011, respectively, where he is currently pursuing the Ph.D. degree with the Laboratory of Autonomous Vehicles, Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev. His research interests include real-time color video processing, quaternion image processing, application of the theory of relativity in color image processing, and light invariant imaging. He currently has eight scientific papers: presented in five conferences and three journals.

**Dan Malowany** received the B.Sc. degree in electrical and computer engineering, the M.Sc. degree in electro-optical engineering, and the Ph.D. degree in electrical and computer engineering from Ben-Gurion University of the Negev, Israel, in 1998, 2003, and 2018, respectively. His Ph.D. research with the Laboratory of Autonomous Robotics was focused at analyzing and designing an architecture that integrates mechanisms of the human visual system with the state of the art deep convolutional neural networks. During his work at the Directorate for Defense Research and Development he had significance role in managing various research and development programs in the area of unattended ground sensors for the homeland security sector. As an entrepreneur, he was involved in numerous startup ventures. He is currently the head of deep learning research at a computer vision startup. His research interests include computer vision, convolutional neural networks, reinforcement learning, the visual cortex, and robotics.

**Rami R. Hagege** received the B.Sc. degree *(summa cum laude)*, the M.Sc. degree *(summa cum laude)*, and the Ph.D. degree from Ben-Gurion University of the Negev, Beer Sheva, Israel, in 2002, 2004, and 2009, respectively, all in electrical and computer engineering. From 2009 to 2010, he was with the Laboratory of Information and Decision Systems, Massachusetts Institute of Technology. From 2010 to 2016, he was with the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev. He has been leading a computer vision startup since 2016.

**Hugo Guterman** received the degree in electronic engineering from National Technological University of Buenos Aires, Argentina, in 1978, and the M.Sc. and Ph.D. degrees in computer and electrical engineering from Ben-Gurion University of the Negev, Beer-Sheva, Israel, in 1982 and 1988, respectively. From 1988 to 1990, he was a Post-Doctoral Fellow at MIT. Since 1990, he has been with the Department of Computer and Electrical Engineering, Ben-Gurion University of the Negev. He is the Head of the Laboratory of Autonomous Robotics, the Chairman of The Paul Ivanier Center for Robotics and the Israel Section of IEEE Robotics and Automation Society. His research interests include autonomous platforms, computer architecture, control, image and signal processing, neural networks and fuzzy logic, electrochemical processes, robotics, biomedicine, biotechnology, and biosensors.