# Learning-based Image Enhancement for Visual Odometry in Challenging HDR Environments

**Ruben Gomez-Ojeda**
University of Malaga, Spain
`rubengooj@gmail.com`

**Zichao Zhang**
University of Zurich, Switzerland
`zzhang@ifi.uzh.ch`

**Javier Gonzalez-Jimenez**
University of Malaga, Spain
`javiergonzalez@uma.es`

**Davide Scaramuzza**
University of Zurich, Switzerland
`sdavide@ifi.uzh.ch`

**Abstract:** One of the main open challenges in visual odometry (VO) is the robustness to difficult illumination conditions or high dynamic range (HDR) environments. The main difficulties in these situations come from both the limitations of the sensors and the inability to perform a successful tracking of interest points because of the bold assumptions in VO, such as brightness constancy. We address this problem from a deep learning perspective, for which we first fine-tune a Deep Neural Network (DNN) with the purpose of obtaining enhanced representations of the sequences for VO. Then, we demonstrate how the insertion of Long Short Term Memory (LSTM) allows us to obtain temporally consistent sequences, as the estimation depends on previous states. However, the use of very deep networks does not allow the insertion into a real-time VO framework; therefore, we also propose a Convolutional Neural Network (CNN) of reduced size capable of performing faster. Finally, we validate the enhanced representations by evaluating the sequences produced by the two architectures in several state-of-art VO algorithms, such as ORB-SLAM and DSO.

**Keywords:** Deep Learning; Visual Odometry; LSTM

## 1 Introduction

In recent years, Visual Odometry (VO) has reached a high maturity, and there are many potential applications, such as unmanned aerial vehicles (UAVs) and augmented/virtual reality (AR/VR). Despite the impressive results achieved in controlled lab environments, the robustness of VO in real-world scenarios is still an unsolved problem. While there are different challenges for robust VO (e.g., weak texture [5][9]), in this work we are particularly interested in improving the robustness in HDR environments. The difficulties in HDR environments come not only from the limitations of the sensors (conventional cameras often take over/under-exposed images in such scenes), but also from the bold assumptions of VO algorithms, such as brightness constancy. To overcome these difficulties, two recent research lines have emerged respectively: Active VO and Photometric VO. The former tries to provide the robustness by controlling the camera parameters (gain or exposure time) [27][31], while the latter explicitly models the brightness change using the photometric model of the camera [18] [6]. These approaches are demonstrated to improve robustness in HDR environments. However, they require detailed knowledge of the specific sensor and a heuristic setting of several parameters, which cannot be easily generalized to different setups.

In contrast to previous methods, we address this problem from a *Deep Learning* perspective, taking advantage of the generalization properties to achieve robust performance in varied conditions. Specifically, in this work, we propose two different Deep Neural Networks (DNNs) that enhance monocular images to more informative representations for VO. Given a sequence of images, our networks are able to produce an enhanced sequence that is invariant to illumination conditions or robust to HDR environments and, at the same time, contains more gradient information for better tracking in VO. For that, we add the following contributions to the state of the art:

○ We propose two different deep networks: a very deep model consisting of both CNNs and LSTM, and another one of small size capable of executing in real-time on a GPU. Both networks transform a sequence of RGB images into more informative ones, while also being robust to changes in illumination, exposure time, gamma correction, etc.

○ We propose a multi-step training strategy which employs the down-sampled images from synthetic datasets, which are augmented with a set of transformations to simulate different illumination conditions and camera parameters. In consequence, our DNNs are capable of generalizing the trained behavior to full resolution real sequences in HDR scenes or under difficult illumination conditions.

○ Finally, we show how the addition of Long Short Term Memory (LSTM) layers helps to produce more stable and less noisy results in HDR sequences by incorporating the temporal information from previous frames. However, these layers increase the computational burden, hence complicating their insertion into a real-time VO pipeline.

We validate the claimed features by comparing the performance of two state-of-art algorithms in monocular VO, namely ORB-SLAM [22] and DSO [6], with the original input and the enhanced sequences, showing the benefits of our proposals in challenging environments.

## 2 Related Work

To overcome the difficulties in HDR environments, works have been done to improve the image acquisition process as well as to design robust algorithms for VO.

### 2.1 Camera Parameter Configuration

The main goal of this line of research is to obtain the best camera settings (i.e., exposure, or gain) for image acquisition. Traditional approaches are based on heuristic image statistics, typically the mean intensity (brightness) and the intensity histogram of the image. For example, a method for autonomously configuring the camera parameters was presented in Neves et al. [23], where the authors proposed to setup the exposure, gain, brightness, and white-balance by processing the histogram of the image intensity. Other approaches exploited more theoretically grounded metrics. Lu et al. [19], employed the Shannon entropy to optimize the camera parameters in order to obtain more informative images. They experimentally proved a relation between the image entropy and the camera parameters, then selected the setup that produced the maximum entropy.

Closely related to our work, some researchers tried to optimize the camera settings for visual odometry. Shim et al. [27] defined an information metric, based on the gradient magnitude of the image, to measure the amount of information in it, and then selected the exposure time that maximized the metric. Recently, Zhang et al. [31] proposed a robust gradient metric and adjusted the camera setting according to the metric. They designed their exposure control scheme based on the photometric model of the camera and demonstrated improved performance with a state-of-art VO algorithm [7].

### 2.2 Robust Vision Algorithms

To make VO algorithms robust to difficult light conditions, some researchers proposed to use invariant representations, while others tried to explicitly model the brightness change. For feature-based methods, binary descriptors are efficient and robust to brightness changes. Mur-Artal et al. [22] used ORB features [25] in a SLAM pipeline and achieved robust and efficient performance. Other binary descriptors [17][2] are also often used in VO algorithms. For direct methods, Alismail et al. [1] incorporated binary descriptors into the image alignment process for direct VO, and the resulting system performed robustly in low light conditions.

To model the brightness change, the most common technique is to use an affine transformation and estimate the affine parameters in the pipeline. Jin et al. [14] proposed an adaptive algorithm for feature tracking, where they employed an affine transformation that modeled the illumination changes. More recently, photometric model, such as the one proposed by Debevec and Malik [3], is used to account for the brightness change due to the exposure
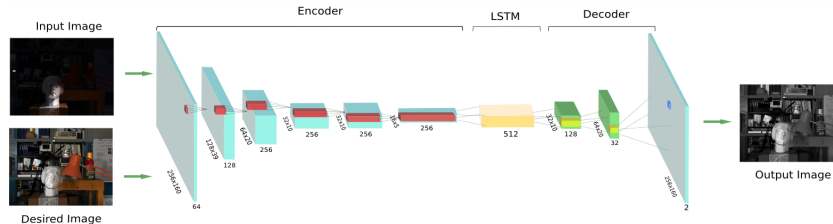
(a) DNN model used in fine-tuning.
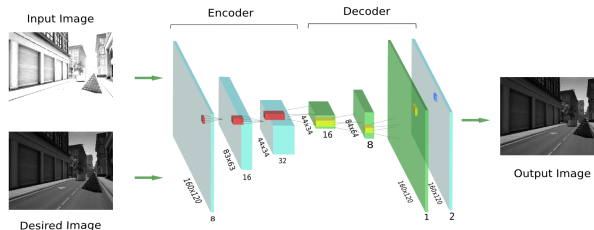


(b) Small-CNN trained from scratch.

Figure 1: Scheme of the architectures employed in this work. Both DNNs are formed by an *encoder* convolutional network, and a *decoder* that forms the enhanced output images. In the case of the fine-tuned network, we introduce a LSTM network to produce temporally consistent sequences. These figures have been adapted from [20, 21].

time variation. A method to deal with brightness changes caused by auto-exposure was published in Li et al. [18], reporting a tracking and dense mapping system based on a normalized measurement of the radiance of the image (which is invariant to exposure changes). Their method not only reduced the drift of the camera trajectory estimation, but also produced less noisy maps. Engel et al. [6] proposed a direct approach to VO with a joint optimization of both the model parameters, the camera motion, and the scene structure. They used the photometric model of the camera as well as the affine brightness transfer function to account for the brightness change. In [31], the authors also adapted a direct VO algorithm [7] with both methods and presented an experimental comparison of using the affine compensation and the photometric model of the camera.

To the best of our knowledge, there is few work on using learning-based methods to tackle the difficulties in HDR environments. In the rest of the paper, we will describe how to design networks for this task, the training strategy and the experimental results.

## 3 Network Overview

In this work, we need to perform a pixel-wise transformation from monocular RGB images in a way that the outputs are still realistic images, on which we will further run VO algorithms. For pixel-wise transformation, the most used approach is DNNs structured in the so-called *encoder-decoder* form. These type of architectures have been successfully employed in many different tasks, such as optical flow estimation [4], image segmentation [15], depth estimation [20], or even to solve the image-to-image translation problem [12]. The proposed architectures (see Figure 1), implemented in Caffe library [13], consist of an encoder, LSTM layers and a decoder, as described in the following.

### 3.1 Encoder

The encoder network consists of a set of purely convolutional layers that transform the input image, into a more reduced representation of feature vectors, suitable for a specific classification task. Due to the complexity of training from scratch [29], a standard approach is to initialize the model with the weights of a pre-trained model, known as *fine-tuning*. This has several advantages, as models trained with massive amount of natural images such as VGGNet [28], a seminal network for image classification, usually provide a good

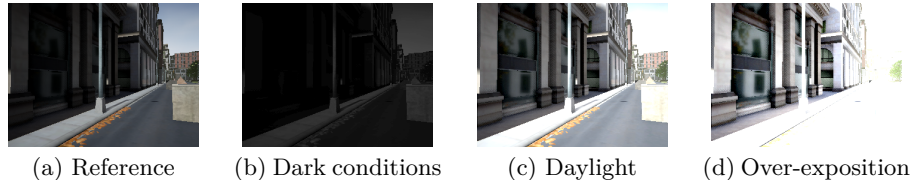| (a) Reference | (b) Dark conditions | (c) Daylight | (d) Over-exposition |

Figure 2: Some training samples extracted from the Urban dataset proposed in [21], for which we have simulated artificial illumination and exposure conditions by post-processing the dataset with different contrast and gamma levels.

performance and stability during the training. Moreover, as initial layers closer to the input image provide low-level information and final layers are more task-specific, it is also typical to employ the first layers of a well-trained CNN for different purposes, i.e. place recognition [8]. This was also the approach in [21], where authors employed the first 8 layers of VGGNet to initialize their network, keeping their weights fixed during training, while the remaining layers were trained from scratch with random initialization. Therefore, in this work, we first fine-tuned the very deep model in [21], depicted in Figure 1a.

However, since our goal is to estimate the VO with the processed sequences, a very deep network, such as the fine-tuned model, is not feasible for real-time applications. Moreover, depth estimation requires a high level of semantic abstraction because it is needed some spatial reasoning about the position of the objects in the scene. In contrast, VO algorithms are usually based on tracking regions of interest in the images, which largely relies on the gradient, i.e., the first derivatives of the images, information that it is usually present in the shallow layers of CNNs. Therefore, we also propose a smaller and less deep CNN to obtain a faster performance suitable to real-time VO, whose encoder is formed by three layers (dimensions are in Figure 1b), each one of them formed by a convolution with a $5 \times 5$ kernel, followed by a batch-normalization layer [11] and a pooling layer.

## 3.2 Long Short Term Memory (LSTM)

While it is feasible to use a feedforward neural network to increase the information in images for VO, the output sequence may not be stable in its brightness, which is likely to break the brightness constancy assumption for many VO algorithms. The input sequence may contain non-ignorable brightness variation and, more importantly, the brightness constancy is not enforced in a feedforward network. To overcome this, we can exploit the sequential information to produce more stable and temporally consistent images, i.e. reducing the impact of possible illumination change to ease the tracking of interest points. Therefore, we exploit the Recurrent Neural Networks (RNNs), more specifically, the LSTM networks first introduced in [10]. In these networks, the main difference is that, unlike in standard CNNs where the output is only a non-linear function $f$ of the current state $\mathbf{y}_t = f(\mathbf{x}_t)$, where $\mathbf{x}_t$ and $\mathbf{y}_t$ are the input and output at time $t$, the output is also dependent on the output of the previous time:

$$\mathbf{y}_t = f(\mathbf{x}_t, \mathbf{y}_{t-1}) \tag{1}$$

as the layers are capable of memorizing the previous states. We introduce two LSTM layers in the fine-tuned network between the encoder and the decoder part, in order to produce more stable results for a better odometry estimation.

## 3.3 Decoder

Finally, the decoder network is formed by three deconvolutional layers, each of them formed by an upsampling, a convolution and a batch-normalization layer, as depicted in Figure 1. The deconvolutional layers increase the size of the intermediate states and reduce the length of the descriptors. Typically, decoder networks produce an output of a proportional size of the input image containing the predicting values, which are in general blurry and noisy. Therefore, instead of directly feeding the raw output of a decoder network to a VO pipeline, we merge the output of the decoder network with the input image by first concatenating

them into a 2-channel image (one for the image and the other for the output of the decoder), then applying a final convolution with a $1 \times 1$ kernel and one channel.

# 4 Training the DNN

Our goal is to produce an enhanced image stream to increase the robustness/accuracy of visual odometry algorithms under challenging situations. Unfortunately, there is no ground-truth available for generating the optimal sequences, nor direct measurement that indicates the goodness of an image for VO. To overcome this difficulties, we observe that the majority of the state-of-art VO algorithms, both *direct* and *feature-based* approaches, actually exploit the gradient information in the image. Therefore, we aim to train our network to produce images containing more gradient information. In this section, we first introduce the dataset used for training then our training strategy.

## 4.1 Datasets

To train the network, we need images taken at **the same pose** but with different illuminations, which are unfortunately rarely available in real-world VO datasets. Therefore we employed synthetic datasets that contain changes in the illumination of the scenes. In particular, we used the well-known New University of Tsukuba dataset [24] and the Urban Virtual dataset generated by [21], consisting of several sequences from an artificial urban scenario with non-trivial 6-DoF motion and different illumination conditions. In order to increase the amount of data, we augmented the datasets 12 times by artificially simulating different camera and illumination conditions with several combinations of different Gamma and Contrast values (see Figure 2). To select the best image $\mathbf{y}^*$ (with the most gradient information), we use the following gradient information metric:

$$g(\mathbf{y}) = \sum_{u_i^*} \|\nabla \mathbf{y}(u_i)\|^2 \tag{2}$$

which is the sum of the gradient magnitude over all the pixels $u_i$ in the image $\mathbf{y}$. For training the CNN, we used RGB images of $256 \times 160$ pixels in the case of fine-tuning the model in [21] and grayscale images of $160 \times 120$ pixels for the reduced network. We trained the LSTM network with full-resolution images ($752 \times 480$), as it does not present the same properties of CNNs that allow for their application in bigger models.

## 4.2 Training the CNN

We first train without LSTM, with the aim of obtaining a good CNN (*encoder-decoder*) capable of estimating the enhanced images from individual (not sequential) inputs. This part of training consists of two stages:

### 4.2.1 Pre-training the Network

In order to obtain a good and stable initialization, we first train the CNN with pairs of images at the same pose, consisting the reference image $y^*$ and an image with different appearance. On our first attempts, we tried to optimize directly the bounded increments of the gradient information (2). The results are very noisy, due to the high complexity of the pixel-wise prediction problem. Instead, we opted to train the CNN by imposing the output to be similar to the reference image, in a pixel-per-pixel manner. For that, we employed the logarithm RMSE error, which is defined for a given reference $y^*$ and an output $y$ image as:

$$\mathcal{L}(\mathbf{y}, \mathbf{y}^*) = \sqrt{\frac{1}{N} \sum_i \|\log y_i - \log y_i^*\|^2} \, , \tag{3}$$

where $i$ is the pixel index in the images. We found this loss function much more suitable for VO applications, as it produced a smoother result than Euclidean RMSE, specially for bigger errors, hence easing the convergence process. This first part of the training was performed with the Adam solver [16], with a learning rate $l = 0.0001$ for 20 epochs of the training data, requiring about 12 hours on a NVIDIA GeForce GTX Titan.

#### 4.2.2 Imposing Invariance

Once a good performance with the previous training was achieved, we trained the CNN to obtain invariance to different appearances. The motivation is that, for images with different appearances (i.e. brightness) taken at the same pose, the CNN should be able produce the same enhanced image. For that, we selected triplets of images from the Urban dataset, by taken the reference image $\mathbf{y}^*$, and another two images $\mathbf{y}_1$ and $\mathbf{y}_2$ from the same place with two different illuminations. Then, we trained the network in a siamese configuration, for which we again imposed both outputs to be similar to the reference one. In addition, we introduced the following loss function:

$$\mathcal{L}_{SSIM}(\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}^*) = SSIM(\mathbf{y}_1, \mathbf{y}_2) \tag{4}$$

which is the structural similarity (SSIM) [30], usually employed to measure how similar two images are. This second part of the training was performed, during 10 epochs of the training data, requiring about 6 hours of training.

### 4.3 Training the LSTM network

After we obtain a good CNN, the second part of the training is designed to increase the stability of the outputs, given that we are processing sequences of consecutive images. The goal is to provide not only more meaningful images, but also fulfill the brightness constancy assumption. For that purpose, we trained the whole DNN, including the LSTM network, with sequences of two consecutive images (i.e., taken at consecutive poses on a trajectory) under slightly different illumination conditions, while the reference ones presented the same brightness. The loss function consists of the LogRMSE loss function (3) to ensure that both outputs are similar to their respective reference ones, and the SSIM loss (4) without the structural term (as images do not belong to the exact same place) between the two consecutive outputs to ensure that they have a similar appearance. The LSTM training was performed during 10 epochs of the data in about 12 hours.
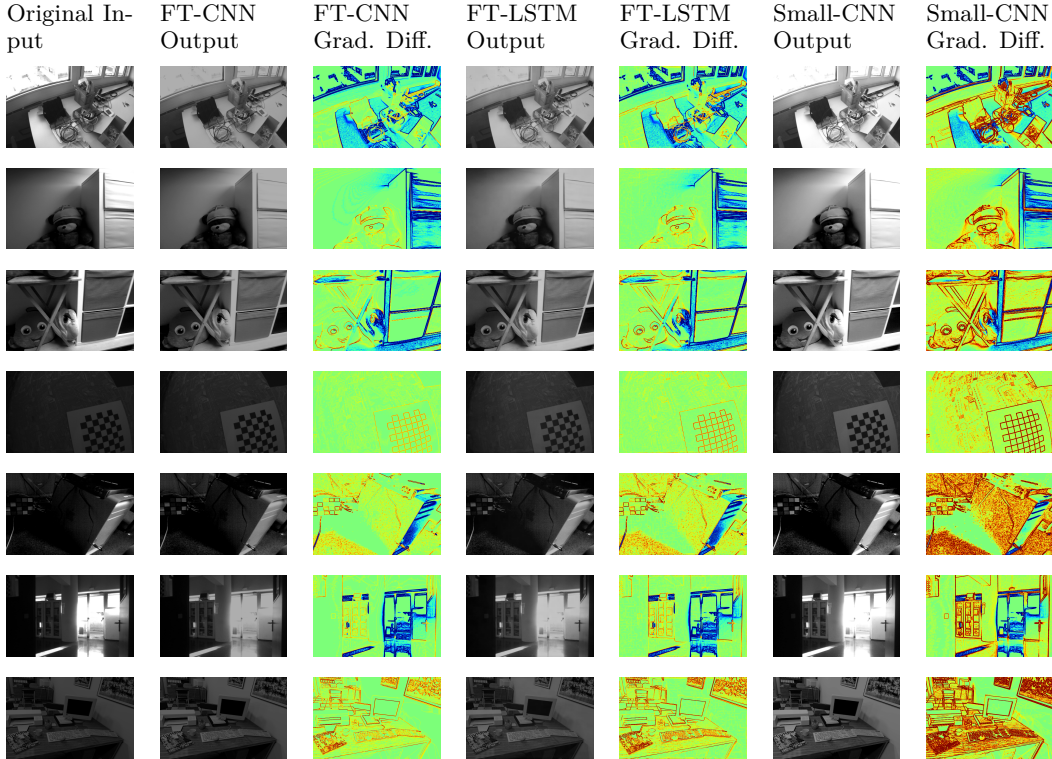


Figure 3: Outputs from the trained models and difference between the gradient images in some challenging samples extracted from the evaluation sequences (the scale for the *jet* colormap remains fixed for each row).

# 5 Experimental Validation

In this section, we evaluate the performance of our approach by measuring two different metrics: the increments of gradient magnitude in the processed images and the improvements in accuracy and performance of ORB-SLAM [22] and DSO [6], two state-of-art VO algorithms for both *feature-based* and *direct* approaches, respectively. It is worth noticing that, even though the CNN networks proposed in this paper (not the FT-LSTM) have been trained only with synthetic images with reduced size ($256 \times 180$ and $160 \times 120$ pixels for the fine-tuned and our proposal respectively), the experiments have been performed with full-resolution ($752 \times 480$ pixels) and real images.

## 5.1 Gradient Inspection

As stated before, one way of measuring the quality of an image is its amount of gradient. Unfortunately, there is no standard metric for measuring the gradient information; actually, it is highly dependent on the application. In the case of visual odometry, it is even more important, as most approaches are based on the edge information (which is directly related to the gradient magnitude image). Figure 3 presents the estimated images and the difference between the gradients of the output and the input images for several images from the trained models in different datasets. For the representation we have used the colormap *jet*, i.e. from blue to red, with $\pm 30$ units of range (negative values indicate a decrease of the gradient amount). In general, we observe a general tendency in all models to reduce the gradient amount in the most exposed parts of the camera as they are less informative due to the sensor saturation, while increasing the gradient in the rest of the image.

## 5.2 Evaluation with state-of-art VO algorithms

We also evaluate the trained models by comparing the performance of ORB-SLAM [22] and DSO [6] in challenging scenarios, with the original image sequence, those obtained as output of the DNNs trained in this work and several standard image processing approaches, i.e. Normalization (N), Global Histogram Equalization (G-HE) [26] and Adaptative Histogram Equalization (A-HE) [32]. For that, we recorded 9 sequences with a hand-held camera in a room equipped with an OptiTrack system that allows us to also record the ground-truth trajectory of the camera and evaluate quantitatively the results. Each sequence was recorded for several illumination conditions: first with $1 - 3$ lights available in the room, then without any light, and finally by switching the lights on and off during the sequence. Table 1 shows the results of ORB-SLAM in all the sequences mentioned above. Firstly, we observe the benefits of our approach as our methods clearly outperform the original input

Table 1: ORB-SLAM [22] average RMSE errors (% first row) normalized by the length of the trajectory and percentage of the sequence without loosing the tracking (second row).

| Dataset | [22] | N | G-HE | A-HE | FT-CNN | FT-LSTM | Small-CNN |
|---|---|---|---|---|---|---|---|
| *1-light* | 3.91 | 4.07 | - | - | 3.52 | 3.49 | 4.62 |
| | 24.80 | 26.98 | - | - | 23.84 | 25.32 | 80.52 |
| *2-lights* | 2.19 | 2.17 | - | 2.27 | 2.07 | 2.09 | 2.72 |
| | 68.92 | 68.76 | - | 65.88 | 70.94 | 72.98 | 68.76 |
| *3-lights* | 3.78 | 3.81 | - | 3.63 | 3.52 | 3.81 | 3.65 |
| | 100.00 | 100.00 | - | 100.00 | 100.00 | 100.00 | 100.00 |
| *switch* | 3.60 | 4.85 | - | 4.56 | 5.64 | 2.66 | 2.97 |
| | 13.76 | 24.98 | - | 8.84 | 7.32 | 31.02 | 21.62 |
| *hdr1* | 5.67 | 5.67 | 3.71 | - | 5.22 | 5.21 | 4.77 |
| | 74.30 | 76.6 | 49.36 | - | 81.54 | 81.14 | 78.76 |
| *hdr2* | 3.49 | 4.08 | 4.42 | 3.52 | 3.42 | 3.88 | 3.51 |
| | 74.86 | 70.50 | 34.12 | 25.3 | 74.52 | 71.02 | 75.22 |
| *overexposed* | 2.64 | 2.57 | 2.59 | 2.53 | 2.72 | 2.65 | 2.83 |
| | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| *bright-switch* | 3.13 | 3.08 | 2.03 | 3.10 | 1.97 | 2.02 | 1.95 |
| | 34.60 | 34.94 | 100.00 | 35.42 | 100.00 | 100.00 | 100.00 |
| *low-texture* | - | - | - | - | 5.28 | - | - |
| | - | - | - | - | 39.08 | - | - |

Table 2: DSO [6] average RMSE errors normalized by the length of the trajectory for each method and trained network when evaluating.

| Dataset | [6] | N | G-HE | A-HE | FT-CNN | FT-LSTM | Small-CNN |
|---|---|---|---|---|---|---|---|
| *1-light* | 2.39 | - | 2.37 | 2.42 | 2.36 | 2.36 | 2.40 |
| *2-lights* | 2.12 | - | 2.05 | 2.12 | 2.12 | 2.15 | 2.14 |
| *3-lights* | 2.65 | - | 2.66 | 2.66 | 2.66 | 2.69 | 2.69 |
| *switch* | - | - | - | - | 4.38 | 4.39 | 2.90 |
| *hdr1* | 2.46 | 4.80 | 2.34 | 2.52 | 2.42 | 2.17 | 2.44 |
| *hdr2* | 1.28 | - | 1.59 | 3.17 | 1.23 | 1.22 | 2.57 |
| *overexposed* | 1.61 | 1.60 | 1.64 | 1.62 | 1.58 | 1.58 | 1.60 |
| *bright-switch* | 4.51 | - | 1.49 | 1.47 | 1.93 | 1.73 | 4.43 |
| *low-texture* | 3.22 | 2.67 | 2.76 | 3.22 | 3.22 | 3.14 | 3.21 |

and the standard image processing approaches in the difficult sequences (*1-light* and *switch*), while also maintaining a similar performance in the easy ones (*2-lights* and *3-lights*). As for the different networks, we clearly observe the better performance of FT-LSTM in the difficult sequences, although the reduced approach Small-CNN reports a good performance in the scene with the switching lights.

The results obtained with DSO are represented in Table 2. Since all the methods were successfully tracked, we omit the tracking percentage. In terms of accuracy, we again observe the good performance of the reduced approach, namely the Small-CNN, with the direct approach. However, its accuracy is worse in the *bright-switch* sequence but it still performs similar to the original sequence.

## 5.3 Computational Cost

Finally, we evaluate the computational performance of the two trained networks. For that, we compare the performance of the CNN and the LSTM, for both the training and the runtime image resolutions. All the experiments were run on a Intel(R) Core(TM) i7-4770K CPU @ 3.50GHz, and an NVIDIA GeForce GTX Titan. Table 3 shows the results of each model, i.e. the fine-tuned approach (FT-CNN and FT-LSTM) and the reduced model (Small-CNN), for all possible resolutions. We first observe that while obtaining comparable results to the fine-tuned model, the small CNN has the advantage that can perform faster (a single frame processing takes 3 times less than with FT-CNN and up to 5 times less than FT-LSTM for the resolution $756 \times 480$ ), and therefore is the closest configuration to a direct application in a VO pipeline It is also worth noticing the important impact of the LSTM layers in the performance, because they not only require a high computational burden but also double the size of the encoder network (a consecutive image pair is needed).

## 6 Conclusions

In this work, we tackled the problem of improving the robustness of VO systems under challenging conditions, such as difficult illuminations, HDR environments, or low-textured scenarios. For that, we solved the problem from a deep learning perspective, for which we proposed two different architectures, a very deep model that is capable of producing temporally consistent sequences due to the inclusion of LSTM layers, and a small architecture for real time VO. We propose a multi-step training employing only reduced images from synthetic datasets, which are also augmented with a set basic transformations to simulate different illumination conditions and camera parameters, as there is no ground-truth available for our purposes. We then compare the performance of two state-of-art algorithms in monocular VO, ORB-SLAM [22] and DSO [6], when using the normal sequences and the ones produced by the DNNs, showing the benefits of our proposals in challenging environments.

Table 3: Average runtime and memory usage for each network

| DNN | Res. (pixels) | Memory | GPU |
|---|---|---|---|
| FT-CNN | $256 \times 180$ | 371 MiB | 23.80 ms |
| FT-CNN | $756 \times 480$ | 1175 MiB | 149.72 ms |
| FT-LSTM | $756 \times 480$ | 3897 MiB | 275.24 ms |
| Small-CNN | $160 \times 120$ | 135 MiB | **4.77 ms** |
| Small-CNN | $756 \times 480$ | 373 MiB | **48.4 ms** |

# References

[1] H. Alismail, M. Kaess, B. Browning, and S. Lucey. Direct Visual Odometry in Low Light using Binary Descriptors. *IEEE Robotics and Automation Letters*, 2016.

[2] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. BRIEF: Computing a Local Binary Descriptor Very Fast. 34(7):1281–1298, 2012.

[3] P. E. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *ACM SIGGRAPH 2008 classes*, page 31. ACM, 2008.

[4] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[5] E. Eade and T. Drummond. Edge landmarks in monocular SLAM. *Image and Vision Computing*, 27(5):588–596, apr 2009. ISSN 02628856. doi:10.1016/j.imavis.2008.04.012.

[6] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *arXiv preprint arXiv:1607.02565*, 2016.

[7] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22. IEEE, 2014.

[8] R. Gomez-Ojeda, M. Lopez-Antequera, N. Petkov, and J. Gonzalez-Jimenez. Training a convolutional neural network for appearance-invariant place recognition. *arXiv preprint arXiv:1505.07428*, 2015.

[9] R. Gomez-Ojeda, J. Briales, and J. Gonzalez-Jimenez. PL-SVO: Semi-direct Monocular Visual Odometry by combining points and line segments. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4211–4216. IEEE, 2016.

[10] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

[11] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[12] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[14] H. Jin, P. Favaro, and S. Soatto. Real-time feature tracking and outlier rejection with changes in illumination. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 684–689. IEEE, 2001.

[15] A. Kendall, V. Badrinarayanan, , and R. Cipolla. Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. *arXiv preprint arXiv:1511.02680*, 2015.

[16] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] S. Leutenegger, M. Chli, and R. Siegwart. BRISK: Binary Robust invariant scalable keypoints. pages 2548–2555, Nov. 2011. doi:10.1109/ICCV.2011.6126542.

[18] S. Li, A. Handa, Y. Zhang, and A. Calway. HDRFusion: HDR SLAM using a low-cost auto-exposure RGB-D sensor. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 314–322. IEEE, 2016.

[19] H. Lu, H. Zhang, S. Yang, and Z. Zheng. Camera parameters auto-adjusting technique for robust robot vision. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1518–1523. IEEE, 2010.

[20] M. Mancini, G. Costante, P. Valigi, and T. A. Ciarfuglia. Fast robust monocular depth estimation for Obstacle Detection with fully convolutional networks. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4296–4303. IEEE, 2016.

[21] M. Mancini, G. Costante, P. Valigi, T. A. Ciarfuglia, J. Delmerico, and D. Scaramuzza. Towards Domain Independence for Learning-Based Monocular Depth Estimation. *IEEE Robotics and Automation Letters*, 2017.

[22] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[23] A. J. Neves, B. Cunha, A. J. Pinho, and I. Pinheiro. Autonomous configuration of parameters in robotic digital cameras. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 80–87. Springer, 2009.

[24] M. Peris, A. Maki, S. Martull, Y. Ohkawa, and K. Fukui. Towards a simulation driven stereo vision system. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1038–1042. IEEE, 2012.

[25] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[26] J. C. Russ, J. R. Matey, A. J. Mallinckrodt, S. McKay, et al. The image processing handbook. *Computers in Physics*, 8(2):177–178, 1994.

[27] I. Shim, J.-Y. Lee, and I. S. Kweon. Auto-adjusting camera exposure for outdoor robotics using gradient information. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 1011–1017. IEEE, 2014.

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[29] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. Convolutional neural networks for medical image analysis: full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312, 2016.

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[31] Z. Zhang, C. Forster, and D. Scaramuzza. Active Exposure Control for Robust Visual Odometry in HDR Environments. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017.

[32] K. Zuiderveld. Contrast limited adaptive histogram equalization. In *Graphics gems IV*, pages 474–485. Academic Press Professional, Inc., 1994.