Transfer Learning from Synthetic Data in the Camera Pose Estimation Problem

Jorge L. Charco[1,2], Angel D. Sappa[1,3], Boris X. Vintimilla[1] and Henry O. Velesaca[1]
[1]*ESPOL Polytechnic University, Escuela Superior Politécnica del Litoral, ESPOL, Campus Gustavo Galindo Km. 30.5 Vía Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador*
[2]*Universidad de Guayaquil, Delta and Kennedy Av., P.B. EC090514, Guayaquil, Ecuador*
[3]*Computer Vision Center, Edifici O, Campus UAB, 08193 Bellaterra, Barcelona, Spain*

Keywords: Relative Camera Pose Estimation, Siamese Architecture, Synthetic Data, Deep Learning, Multi-view Environments, Extrinsic Camera Parameters.

Abstract: This paper presents a novel Siamese network architecture, as a variant of Resnet-50, to estimate the relative camera pose on multi-view environments. In order to improve the performance of the proposed model a transfer learning strategy, based on synthetic images obtained from a virtual-world, is considered. The transfer learning consists of first training the network using pairs of images from the virtual-world scenario considering different conditions (i.e., weather, illumination, objects, buildings, etc.); then, the learned weight of the network are transferred to the real case, where images from real-world scenarios are considered. Experimental results and comparisons with the state of the art show both, improvements on the relative pose estimation accuracy using the proposed model, as well as further improvements when the transfer learning strategy (synthetic-world data transfer learning real-world data) is considered to tackle the limitation on the training due to the reduced number of pairs of real-images on most of the public data sets.

1 INTRODUCTION

Automatic calibration of the camera extrinsic parameters is a challenging problem involved in several computer vision processes; applications such as driving assistance, human pose estimation, mobile robots, 3D object reconstruction, just to mention a few, would take advantage of knowing the relative pose between the camera and world reference system. During last decades different computer vision algorithms have been proposed for extrinsic camera parameters estimation (relative translation and rotation) (e.g., (Hartley, 1994),(Sappa et al., 2006), (Liu et al., 2009), (Dornaika et al., 2011), (Schonberger and Frahm, 2016), (Iyer et al., 2018), (Lin et al., 2019)).

Classical approaches estimate the extrinsic camera parameters from feature points (e.g., SIFT (Lowe, 1999), SURF (Bay et al., 2006), BRIEF (Calonder et al., 2012)), which are detected and described in the pair of images used to estimate the pose (relative translation and rotation) between the cameras. Theses algorithms have low accuracy when they are unable to find enough common feature points to be matched.

During last years, convolutional neural networks (CNNs) have been widely used for feature detection in tasks such as segmentation, images classification, super resolution and pattern recognition, getting better results than state-of-art (e.g., (Kamnitsas et al., 2017), (Wang et al., 2016), (Rivadeneira et al., 2019)). Within this scheme, different CNN based camera calibrations, on single and multi-view environments, have been also proposed showing appealing results (e.g., (Shalnov and Konushin, 2017), (Charco et al., 2018)). The single view approaches capture real-world environments by using a single camera that is constantly moving around the scene that may contains dynamic objects (i.e., pedestrians, cars). The challenge with these approaches lie on the scenarios with moving objects, which change their position during the acquisition and could occlude different scene's regions (features) at consecutive frames. This problem is not present in multi-view approaches, due to the fact that the scene is simultaneously captured from different positions by different cameras, considering a minimum overlap of regions between the captured scenes.

Although appealing results are obtained with learning based approaches, they have as a main limitation the size of data set used for training the net-

Charco, J., Sappa, A., Vintimilla, B. and Velesaca, H.
Transfer Learning from Synthetic Data in the Camera Pose Estimation Problem.
DOI: 10.5220/0009167604980505
In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2020) - Volume 4: VISAPP*, pages 498-505
ISBN: 978-989-758-402-2; ISSN: 2184-4321

work, which becomes an important factor. In the particular case of extrinsic camera parameter estimation, there are several data sets devoted for this task. Actually, some of them have been proposed for applications such as mapping, tracking or relocalization (e.g., Cambridge (Kendall et al., 2015) and 7-Scene (Shotton et al., 2013)). Unfortunately, most of these data sets contain just a small number of images acquired by single view approaches. Trying to overcome this limitation, in (Aanæs et al., 2016), the authors have provided a multi-view data set, acquired using a robotic arm, called DTU-Robot. Main drawback with this data set lies on the fact that an indoor and small scenario, containing few objects, has been considered; transferring knowledge from this scenario to real outdoor environments becomes a difficult task.

As mentioned above, the limited size of data sets is a common problem in learning-based solutions. In some cases this problem has been overcome by using virtual environments where data sets are acquired (e.g., 3D object recognition (Jalal et al., 2019), optical flow estimation (Onkarappa and Sappa, 2015)). These virtual environments allow generating an almost unlimited set of synthetic images by rendered many times with different conditions and actors (i.e., weather, illumination, pedestrian, road, building, vehicles). Another appealing feature of synthetic data sets is related with the lack of manual ground truth annotations required in these scenarios.

In the current work, a CNN architecture is proposed to estimate extrinsic camera parameters by using pairs of images acquired from the same scene and from different points of view at the same time. This architecture is firstly trained using synthetic images generated in a virtual world (i.e., a 3D representation of an urban environment containing a lot of buildings and different objects) and then its weights updated using a transfer learning strategy using real data. A Siamese network architecture is proposed as a variant of Resnet-50. The remainder of the paper is organized as follows. In Section 2 previous works are summarized; then, in Section 3 the proposed approach is detailed together with a description of the used synthetic data sets. Experimental results are reported in Section 4 and comparisons with a previous approach are also presented. Finally, conclusions and future work are given in Section 5.

2 RELATED WORK

During the last years, CNNs models have been used in many computer vision tasks due to their capability to extract features improving state-of-art results. In that direction, some works have been proposed for camera pose estimation from a single view approach. The authors in (Kendall et al., 2015) have proposed a CNN architecture to regress the 6-DOF camera pose from a single RGB image, being robust to indoors and outdoors environments in real time, even difficult lighting, motion blur and different camera intrinsics parameters. The previous approach was updated with a similar architecture and a new loss function to learn camera pose in (Kendall and Cipolla, 2017). In (Shalnov and Konushin, 2017) the authors have proposed a novel method for camera pose estimation based on the scene's prior-knowledge. The approach is trained on a synthetic scenario where the camera pose is estimated based on human body features. The trained network is then generalized to real environments. On the contrary to previous approaches, in (Iyer et al., 2018) a self-supervised deep network is proposed to estimate the 6-DOF, rigid body transformation, between a 3D LiDAR and a 2D camera in real-time. The approach is then used to estimate the calibration parameters.

Just few works have been proposed to solve the camera pose estimation problem in multi-view environments using CNNs. In (Charco et al., 2018), the authors have proposed to use a Siamese CNN based on a modified AlexNet architecture with two identical branches and shared weights. The training process was performed from scratch with a set of pairs of images of the same scene simultaneously acquired from different points of view. The output of each branch is concatenated to two fully connected layers to estimate the relative camera pose (translation and rotation). Euclidean distance is used as a loss function. In (En et al., 2018) the authors have also proposed a Siamese Network with two branches regressing one pose per image. GoogLeNet architecture is used to extract features and the pose regressor contains two fully connected layers with ReLU activation. The quaternion is normalized during test time. Euclidean distance and weighting term (β) are used to balance the error between translation and rotation. The authors in (Lin et al., 2019) have presented an approach based on Recurrent Convolutional Neural Networks (RCNNs). They used the first four residual blocks of the ResNet-50. The output of each consecutive monocular image is concatenated to fed the last block of the ResNet-50. Two RCNNs are used, the first is fed by the concatenated output of two consecutive images to two Long Short-Term Memory (LSTM) to find the correlations among images. The second RCNN is fed from a monocular image to LSTM and its output is reshaped to a fully connected layer. Finally, both outputs are concatenated to obtain the translation and rotation estimation.

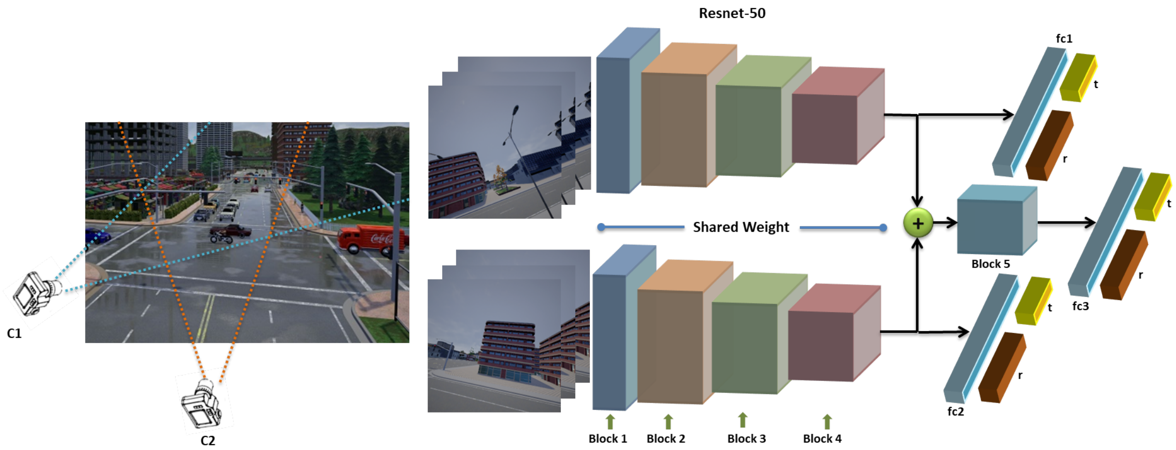

Figure 1: Siamese architecture feds with two images of the same scene captured at the same time from different points of views. The regression part contains three fully-connected layers to estimate the extrinsic camera parameters.

3 PROPOSED APPROACH

This section details the two main contributions of current work. Firstly, the Siamese network architecture proposed to estimate the relative pose between two cameras, which synchronously acquire images of the same scenario, is presented. The network takes as an input a pair of images of the same scenario, captured from different points of view (position and rotation) (see Fig. 1) and estimate the relative rotation and translation between the cameras. As mentioned above, the second contribution of current work lies on the strategy used to train the network. This strategy consists of training the proposed network using a synthetic data set (outdoor environments acquired by using CARLA simulator (Dosovitskiy et al., 2017)) and then transferring the knowledge of trained network in virtual environment to a real-world. The proposed transfer learning based strategy tackles the problem of having a large data set for the training process.

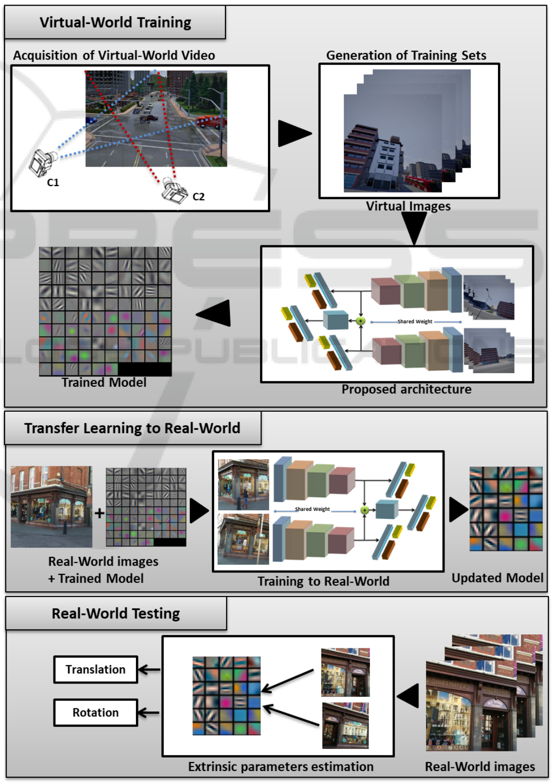

Figure 2: Training and testing processes.

3.1 Network Architecture

The proposed approach is a modified Resnet-50 (He et al., 2016), which contains two identical branches with shared weights up to the fourth residual block. The last residual (fifth block) is fed by concatenating the output of the fourth residual block from each branch. The architecture is composed with multiple residual units, bottleneck architecture consisting of convolutional layers, batch normalization, pooling and identity blocks. The standard residual block structure was modified by replacing RELUs with ELUs as activation function. According to (Clevert et al., 2015) ELU helps to speed up convergence and to reduce the bias shift neurons, avoiding the vanishing gradient. Furthermore, the last average pooling layer is replaced with a global average pooling layer. Two fully connected layers are added after the fourth residual blocks for each branch (left and right), *fc1* and *fc2*, and after the fifth residual block an additional fully connected layer is also added *fc3*. The **global pose** of each camera is predicted from the features extracted from the corresponding image, up to fourth residual block, which are used to feed the *fc1* or *fc2*. Each fc_i has a dimension of 1024 followed by two regressors, corresponding to the global translation (3×1) and rotation (4×1) respectively. Regarding the **relative camera pose**, the output of the last residual block (fifth block) is used to feed *fc3* and subsequently the two regressors used to estimate the relative translation (3×1) and rotation (4×1) between the given pair of images.

Relative camera pose is represented by two vectors: $\Delta p = [\hat{t}, \hat{r}]$, where $\hat{t}$ is a 3-dimensions vector that represents the translation and $\hat{r}$ is a 4-dimensions vector that represents the rotation (i.e., a quaternion). As the image comes in the same context is reasonable to build one model to train both components (translation and rotation) at the same time. Normally, the Euclidean distance is used to estimate them:

$$T_{Global}(I) = \left\| t - \widehat{t} \right\|_{\gamma}, \quad (1)$$

$$R_{Global}(I) = \left\| r - \frac{\widehat{r}}{\|\widehat{r}\|} \right\|_{\gamma}, \quad (2)$$

where t represents the ground truth translation and $\widehat{t}$ denotes the prediction. r is the ground truth rotation and $\widehat{r}$ denotes the prediction of the quaternion values. The estimated rotation (i.e., the estimated quaternioin values) is normalized to a unit length as $\frac{\widehat{r}}{\|\widehat{r}\|}$. γ is $L2$ Euclidean norm. The previous terms are merged according to a factor β, which is introduced due to the difference in scale between both components (translation and rotation) to balance the loss terms (Kendall et al., 2015). Hence, the general loss function, that includes both terms, is defined as:

$$Loss_{global}(I) = T_{Global} + \beta * R_{Global}, \quad (3)$$

setting the β parameter to the right value is a challenging task that depends on several factors related to the scene and cameras. In order to find the best solution (Kendall and Cipolla, 2017) propose to use two learnable variables called $\widehat{s}_x$ and $\widehat{s}_y$ that acts as weight to balance translation and rotation terms—with a similar effect to β. In the current work, a modified loss function that uses $\widehat{s}_y$ as learnable variable is used:

$$Loss_{Global}(I) = T_{Global} + (exp(\widehat{s}_y) * R_{Global} + \widehat{s}_y). \quad (4)$$

Each branch of the Siamese architecture estimates the global pose of the image. Additionally, by connecting these branches together through the fifth block the relative pose between the cameras is estimated; this relative pose is obtained as follow:

$$T_{Relative}(I) = \left\| t_{rel} - \widehat{t}_{rel} \right\|_{\gamma}, \quad (5)$$

$$R_{Relative}(I) = \left\| r_{rel} - \frac{\widehat{r}_{rel}}{\|\widehat{r}_{rel}\|} \right\|_{\gamma}, \quad (6)$$

where $T_{Relative}$ and $R_{Relative}$ estimate the difference between the ground truth (i.e., the relative one) and the prediction of trained model ($\widehat{t}_{rel}$ and $\widehat{r}_{rel}$). As $\widehat{r}_{rel}$ is directly obtained from network it has to be normalized before. Equation (7) and Eq. (8) show how to obtain t_{rel} and r_{rel}.

$$t_{rel} = t_{C1} - t_{C2}, \quad (7)$$

$$r_{rel} = r^{*}_{C2} * r_{C1}, \quad (8)$$

where Ci corresponds to the pose parameters of the (i) camera (i.e., rotation and translation) given by the CARLA simulator; these parameters are referred to a global reference system; r^{*}_{C2} is the conjugate quaternion of r_{C2}. Normalize the quaternion is an important process before using Eq. (8). Finally, the loss function used to obtain the relative pose is:

$$Loss_{Relative}(I) = T_{Rel} + (exp(\widehat{s}_y) * R_{Rel} + \widehat{s}_y). \quad (9)$$

Note that $Loss_{Global}$ in Eq. (4) and $Loss_{Relative}$ in Eq. (9) are applied for different purposes. The first one is used to predict global pose through each branch of the trained model. While the second predicts the relative pose by concatenating the Siamese Network (see Fig. 1). The proposed approach was jointly trained with Global and Relative Loss, as shown in Eq. (10):

$$L = Loss_{Global} + Loss_{Relative}. \quad (10)$$

3.2 Synthetic Data Set

This section presents the steps followed for the synthetic data set generation. Two open-source software tools were used: CARLA Simulator (Dosovitskiy et al., 2017) and OpenMVG (Moulon et al., 2016). This first one, CARLA, is used for generating synthetic images from a virtual-world. It has been developed from the ground up to support the development, training, and validation of autonomous urban driving systems. The second open-source software (OpenMVG) has been used to estimate the overlap between a given pair of images. It is designed to provide easy access to the classical problem solvers in multiple view geometry and solve them accurately.

A workstation with a Titan XP GPU was used for server execution due to the large amount of processing demanded by the CARLA simulator. The attributes and values used to configure the cameras in CARLA simulator are as follow: image_size_x = 448, image_size_y=448, sensor_tick & 1.0, FOV = 100.

Table 1: Attributes and values for the selected weathers.

Weather	Cloudyness	Precipitation	Precipitation deposits	Wind intensity	Sun azimuth (ang.)	Sun altitude (ang.)
Custom weather	0	0	0	0.00	-90	60
Clear noon	15	0	0	0.35	0	75
Cloudy noon	80	0	0	0.35	0	75
Wet noon	20	0	50	0.35	0	75
Wet cloudy noon	80	0	50	0.35	0	75
Mid rainy noon	80	30	50	0.40	0	75
Hard rainy noon	90	60	100	1.00	0	75
Soft rain noon	70	15	50	0.35	0	75
Clear sunset	15	0	0	0.35	0	15
Cloudy sunset	80	0	0	0.35	0	15
Wet sunset	20	0	50	0.35	0	15
Wet cloudy sunset	90	0	50	0.35	0	15
Mid rain sunset	80	30	50	0.40	0	15
Hard rain sunset	80	60	100	1.00	0	15
Soft rain sunset	90	15	50	0.35	0	15

Figure 3: Trajectory followed by the camera in CARLA Simulator.

The cameras were attached to moving platform to allow navigation through the virtual-world. Fifteen different types of weathers are used to generate the set of pairs of images. All selected weather and their respective parameter settings are shown in Table 1.

The trajectory followed by the cameras is shown in Fig 3. It starts from the green point and moves to left following the arrows until it ends at the same green point where it starts. The pair of cameras follow the aforementioned path while their relative pose (relative position and orientation between them) randomly changes at each time in the range $x = [-0.5, 0.5]$, $y = [-0.5, 0.5]$, and $z = [-0.5, 0.5]$, while their relative orientation changes in the range $roll = [0, 12]$, $pitch = [-5, 5]$, and $yaw = [-5, 5]$ degrees, these values were defined according to the scene and building characteristics. Values outside these ranges generate a pair of images with little overlap.

OpenMVG is then used to obtain the list of pairs of images with an overlap higher than a give threshold. The configuration options of OpenMVG has been set as presented in Table 2. The ground truth for the global camera pose has been obtained from CARLA simulator, while the relative camera pose is computed from Eq. (7) and Eq. (8).

Table 2: Options available for OpenMVG script.

Order	Option	Used
1	Intrinsics analysis	Yes
2	Compute features	Yes
3	Compute matches	Yes
4	Do Incremental/Sequential reconstruction	No
5	Colorize Structure	No
6	Structure from Known Poses (robust triangulation)	No

4 EXPERIMENTS RESULTS

As mentioned above, this paper has two main contributions. On the one hand, a new CNN based architecture is proposed for extrinsic camera parameter estimation; on the other hand, trying to overcome limitations related with the reduced amount of data provided in most of public data sets a transfer learning strategy is proposed. This strategy is based on the usage of synthetic data generated using CARLA simulator (Dosovitskiy et al., 2017). Hence, this section first presents details on the data set generation; then, experimental results by training the proposed architecture with the ShopFacade and OldHospital of Cambridge data set (Kendall et al., 2015) are depicted; and finally, results obtained when the proposed transfer learning strategy is used (i.e., first training the network on the synthetic data and then updating network weights by keep training it with real data).

The proposed approach was implemented with the TensorFlow and trained with NVIDIA Titan XP GPU and Intel Core I9 3.3GHz CPU. Adam optimizer is used to train the network with a learning rate of 10^{-4} and batch size of 32. The $\widehat{s_y}$ is initialized with -6.0 in all the experiments.

Figure 4: (*1st row*) Images from the synthetic image data set generated by the CARLA simulator (they were captured at different weather and lighting conditions. (*2nd and 3rd row*) Real-world images, ShopFacade and OldHospital of Cambridge data set respectively, used to evaluate the trained model.

4.1 Training on Real Data

Using the strategy of transfer learning, all layers were initialized up to the fourth residual block with the weights of Resnet-50 pretrained on ImageNet and the normal distribution initialization was used for the remaining layers. The network architecture was trained on ShopFacade and OldHospital of Cambridge data set. As pre-processing data set, the images were resized to 224 pixels along the shorter side; then, the mean value was computed and subtracted from the images. For the training process, random crops of 224×224 pixels have been computed on the OldHospital data set; this results in a set of 5900 pairs of images, which were used to fed to the network. The same process has been performed at 1300 pairs of images of ShopFacade data set. Both data set were used to trained the network until 500 epochs, which approximately took 7 hours and 3 hours respectively.

The pre-processing mentioned above has been also used during the evaluation phase. In the evaluation a set of 2100 pairs of images from the OldHospital and a set of 250 pairs of images from ShopFacade have been considered. On the contrary to the training stage, in this case a central crop is used instead of a random crop. Since the relationship between the pair of images is required for the training stage, both the relative and the absolute pose were estimated.

4.2 Transfer Learning Strategy

This section presents details on the strategy followed for transferring the parameters learned in a synthetic data set to real images. The proposed architecture was initialized as presented in Section 4.1. In this case the training process was performed on the synthetic data set described in Section 3.2. Since the size of synthetic images is 448×448 pixels, a resize up to 224×224 pixels is performed; like in the previous case, the average mean value is estimated and subtracted from each image. The synthetic data set contains 23384 pairs of images. The training stage took about 5 hours with 300 epochs. After the training process from the synthetic images, learned weights were used to initialize all layers of proposed architecture, which is retrained and refined in an end-to-end way with real-world images (Fig. 4 shows illustrations of images used during the training and transfer learning processes).

4.3 Results

Experimental results obtained with the proposed network and training strategy are presented. Additionally, the obtained results are compared with a state-of-the-art CNN-based method Pose-MV (Charco et al., 2018) on ShopFacade and OldHospital of Cambridge data sets. Average median error on rotation and translation for both data sets are depicted on Table 3. An-

Table 3: Comparison of average median Relative Pose Errors (extrinsic parameters) of RelPoseTL with respect to Pose-MV on ShopFacade and OldHospital of Cambridge data set.

	State-of-the-art	RelPoseTL (Ours experiments)	
Scene / Models	Pose-MV	Real data	Sythentic data (Transfer Learning)
ShopFacade	1.126m, 6.021	1,002m, 3.655	**0.834m, 3.182**
OldHospital	5.849m, 7.546	3.792m, 2.721	**3.705m, 2.583**
Average	3.487m, 6.783	2.397m, 3.188	**2.269m, 2.882**

Figure 5: Challenging scenarios, pairs of images from points of view with large relative rotation and translation, as well as moving objects.

gular error and Euclidean distance error are used to evaluate the performance of the proposed approach. The first one is used to compute the rotation error between the obtained result and the given ground truth value using the 4-dimensional vector (quaternion). The second one is used to measure the distance error between the estimated translation and the corresponding ground truth (3-dimensional translation vector). The proposed approach obtains more accurate results on both translation and rotation in both data sets. The average median translation error obtained by RelPoseTL(Real Data) improves the results of Pose-MV in about 32% (both approaches trained with the same real date); this improvements reaches up to 35% when the proposed transfer learning strategy is considered. With respect to average median rotation error, RelPoseTL (Real Data) improves by 53% the results obtained with Pose-MV; this improvements reaches up to the 58% when the proposed transfer learning strategy is considered. Large errors are generated by challenging scenarios such as those presented in Fig. 5.

5 CONCLUSIONS

This paper addresses the challenging problem of estimating the relative camera pose from two different images of the same scenario, acquired from two different points of view at the same time. A novel deep learning based architecture is proposed to accurately estimate the relative rotation and translation between the two cameras. Experimental results and comparisons are provided showing improvements on the obtained results. As a second contribution of this paper a training strategy based on transfer learning from Synthetic data is proposed. This strategy is motivated by the reduced amount of images on the data sets provided to train the network. The manuscript shows how features extracted from large amounts of synthetic images can help the estimation of relative camera pose in real-world images. The proposed architecture has been trained both, by only using real images and by using the proposed transfer learning strategy. Experimental results show that the proposed transfer learning approach helps to reduce error in the obtained results (relative rotation and translation). Future work will be focused on extending the usage the synthetic images data set, by increasing the data set with others outdoor multi-view environments, including different virtual simulators. The goal of this future work is to reach the state-of-the-art results without using a transfer learning strategy, in other words by training the network just with synthetic scenarios. Additionally, different models based on CNNs will be considered.

ACKNOWLEDGEMENTS

This work has been partially supported by the ESPOL projects EPASI (CIDIS-01-2018) and TICs4CI (FIEC-16-2018); the Spanish Government under Project TIN2017-89723-P; and the CERCA Programme/Generalitat de Catalunya". The authors acknowledge the support of CYTED Network: "Ibero-

American Thematic Network on ICT Applications for Smart Cities" (REF-518RT0559) and the NVIDIA Corporation for the donation of the Titan Xp GPU. The first author has been supported by Ecuador government under a SENESCYT scholarship contract CZ05-000040-2018.

REFERENCES

Aanæs, H., Jensen, R. R., Vogiatzis, G., Tola, E., and Dahl, A. B. (2016). Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*.

Bay, H., Tuytelaars, T., and Gool, L. J. V. (2006). SURF: Speeded Up Robust Features. In *Proceedings of the 9th European Conference on Computer Vision, Graz, Austria, May 7-13*, pages 404–417.

Calonder, M., Lepetit, V., Özuysal, M., Trzcinski, T., Strecha, C., and Fua, P. (2012). BRIEF: Computing a local binary descriptor very fast. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1281–1298.

Charco, J. L., Vintimilla, B. X., and Sappa, A. D. (2018). Deep learning based camera pose estimation in multiview environment. In *2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pages 224–228. IEEE.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.

Dornaika, F., Álvarez, J. M., Sappa, A. D., and López, A. M. (2011). A new framework for stereo sensor pose through road segmentation and registration. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):954–966.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16.

En, S., Lechervy, A., and Jurie, F. (2018). Rpnet: an end-to-end network for relative camera pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Hartley, R. I. (1994). Self-calibration from multiple views with a rotating camera. In *European Conference on Computer Vision*, pages 471–478. Springer.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Iyer, G., Ram, R. K., Murthy, J. K., and Krishna, K. M. (2018). Calibnet: Geometrically supervised extrinsic calibration using 3d spatial transformer networks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Jalal, M., Spjut, J., Boudaoud, B., and Betke, M. (2019). Sidod: A synthetic image dataset for 3d object pose recognition with distractors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0.

Kamnitsas, K., Ledig, C., Newcombe, V. F., Simpson, J. P., Kane, A. D., Menon, D. K., Rueckert, D., and Glocker, B. (2017). Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. *Medical image analysis*, 36:61–78.

Kendall, A. and Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5974–5983.

Kendall, A., Grimes, M., and Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946.

Lin, Y., Liu, Z., Huang, J., Wang, C., Du, G., Bai, J., and Lian, S. (2019). Deep global-relative networks for end-to-end 6-dof visual localization and odometry. In *Pacific Rim International Conference on Artificial Intelligence*, pages 454–467. Springer.

Liu, R., Zhang, H., Liu, M., Xia, X., and Hu, T. (2009). Stereo cameras self-calibration based on sift. In *2009 International Conference on Measuring Technology and Mechatronics Automation*, volume 1.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision, Kerkyra, Greece, September 20-27*, pages 1150–1157.

Moulon, P., Monasse, P., Marlet, R., and Others (2016). Openmvg. an open multiple view geometry library. https://github.com/openMVG/openMVG.

Onkarappa, N. and Sappa, A. D. (2015). Synthetic sequences and ground-truth flow field generation for algorithm validation. *Multimedia Tools and Applications*, 74(9):3121–3135.

Rivadeneira, R. E., Suárez, P. L., Sappa, A. D., and Vintimilla, B. X. (2019). Thermal image superresolution through deep convolutional neural network. In *International Conference on Image Analysis and Recognition*, pages 417–426. Springer.

Sappa, A., Gerónimo, D., Dornaika, F., and López, A. (2006). On-board camera extrinsic parameter estimation. *Electronics Letters*, 42(13):745–747.

Schonberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113.

Shalnov, E. and Konushin, A. (2017). Convolutional neural network for camera pose estimation from object detections. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 42.

Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., and Fitzgibbon, A. (2013). Scene coordinate regression forests for camera relocalization in rgb-d images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937.

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., and Xu, W. (2016). Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2285–2294.