Self-Supervised Optical Flow Estimation by Projective Bootstrap

Stefano Alletto, Davide Abati, Simone Calderara, *Member, IEEE*, Rita Cucchiara, and Luca Rigazio

***Abstract*— Dense optical flow estimation is complex and time consuming, with state-of-the-art methods relying either on large synthetic data sets or on pipelines requiring up to a few minutes per frame pair. In this paper, we address the problem of optical flow estimation in the automotive scenario in a self-supervised manner. We argue that optical flow can be cast as a geometrical warping between two successive video frames and devise a deep architecture to estimate such transformation in two stages. First, a dense pixel-level flow is computed with a projective bootstrap on rigid surfaces. We show how such global transformation can be approximated with a homography and extend spatial transformer layers so that they can be employed to compute the flow field implied by such transformation. Subsequently, we refine the prediction by feeding a second, deeper network that accounts for moving objects. A final reconstruction loss compares the warping of frame X_t with the subsequent frame X_{t+1} and guides both estimates. The model has the speed advantages of end-to-end deep architectures while achieving competitive performances, both outperforming recent unsupervised methods and showing good generalization capabilities on new automotive data sets.**

***Index Terms*— Computer vision, image motion analysis, unsupervised learning.**

I. Introduction

IN THE last few years, we assisted to a growing interest from the computer vision and machine learning community towards autonomous and assisted driving applications. In this context, optical flow estimation represents one of the most active research fields but, despite the efforts, it is still an open problem.

This is due to two main factors. In the first place, the car ego-motion heavily affects the optical flow field. Indeed, urban scenes are mainly composed of static, still objects, whose moving patterns within the image plane are strongly correlated to the camera movement. Nevertheless, several surrounding objects can move independently (such as cars and pedestrians), and their movements are perhaps even more important. For an optical flow model to be reliable, both types of motion need to be correctly estimated.

Manuscript received March 24, 2018; revised August 24, 2018; accepted September 30, 2018. The Associate Editor for this paper was S. C. Wong. *(Corresponding authors: Stefano Alletto; Luca Rigazio.)*

S. Alletto was with the Enzo Ferrari Department of Engineering, University of Modena and Reggio Emilia, 41125 Modena, Italy. He is now with Panasonic Beta, Mountain View, CA 94043 USA (e-mail: stefano.alletto@us.panasonic.com).

D. Abati, S. Calderara, and R. Cucchiara are with the Enzo Ferrari Department of Engineering, University of Modena and Reggio Emilia, 41125 Modena, Italy.

L. Rigazio was with the Panasonic Silicon Valley Laboratory, Cupertino, CA 95014 USA. He is now with Panasonic Beta, Mountain View, CA 94043 USA (e-mail: luca.rigazio@us.panasonic.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/TITS.2018.2873980

Fig. 1. Examples of motion flows obtained through our projective-bootstrapped network. Rows from top to bottom: input example, projective bootstrap, final flow estimate.

Furthermore, existing machine learning models are trained with precise pixel-level optical flow ground-truth maps, that are extremely hard to collect. This represents a significant issue in general, and is even more critical in the automotive field. For this reason, automotive datasets typically lack optical flow ground-truth information [1], [4], [8]. The only suitable real-world labeled dataset is the popular Kitti Flow benchmark [11], that is collected by means of expensive, dedicated hardware. Indeed, ground-truth flow maps are computed by means of 2D-3D matching of point clouds acquired by a LIDAR sensor. Alternatively, computer graphics approaches [10] or videogames [24] are employed to acquire larger synthetic datasets at the expense of photorealism. On the other hand, acquiring a video from the perspective of a car is trivial. Despite the availability of unlabeled driving sequences is high, recent unsupervised approaches to optical flow estimation either struggle in achieving competitive results [35] or perform far from real-time [21].

In this paper, we design an end-to-end network that copes with both issues. Our main contribution is to bootstrap the optical flow estimation by a projective geometric transformation between consecutive image frames. We argue that the global ego-motion of a car can be coarsely approximated by the motion field implied by such transformation, and motivate our strategy within the plane2parallax framework [16]. We show that, when dealing with static scenes and in presence of camera motion, a specific implicit plane exists, that allows to embed both planar and parallax components of the flow field in the same projective matrix. This finding allows us to delegate the estimation of most flow vectors to an extremely shallow, fast deep network, and then refine its prediction accounting for depth structure and moving objects (Fig. 1). Moreover, we only rely on self-supervision to train our model. Building on

the recently proposed Spatial Transformer Layer (STL) [17], we embrace an optimization framework in which, instead of directly minimizing flow vector errors, we warp images in estimates of future frames and penalize reconstruction failures. We show that, despite being very different, such framework effectively reduces optical flow average point error during training.

To summarize, here we propose a method which is specifically designed for the challenging automotive scenario and is particularly suitable for tasks where optical flow ground-truth cannot be employed. Our self-supervised architecture results in significant differences compared to traditional supervised deep learning approaches; on the other hand we improve on recent unsupervised and self-supervised methods by introducing our idea of projective bootstrap, a computationally inexpensive step that we experimentally show to significantly boost the performance.

The paper is structured as follows: The next section discusses recent works on the topics of optical flow estimation. Sec. III describes the intuition behind the projective bootstrap, and the proposed end-to-end model for optical flow estimation. Sec. IV measures performances and generalization capabilities of our model in different settings. Eventually, conclusions and further development strategies are discussed in Sec. V.

II. Related Work

Here, we identify three main approaches to optical flow estimation: hand-crafted methods based on heuristics, composed methods and end-to-end strategies, with every category having its own strengths and drawbacks in terms of speed, performance and complexity.

A. Hand-Crafted Methods

Since its early days, optical flow estimation has been addressed from an image processing perspective, mainly adopting strategies that rely on the brightness constancy principle [13]. Models that solely rely on pixel brightness and motion smoothness suffer of a high sensitivity to outliers, that often occur due to changes in illumination condition and large displacements. To this end, Brox *et al.* [5] propose a variational approach that deals with the shortcomings of previous methods by jointly accounting for brightness constancy and brightness gradient constancy, as long as introducing piecewise smoothness. On a different note, Chen *et al.* [7] approach large displacement optical flow estimation by first computing a nearest-neighbor field for each pixel, which is shown to coarsely approximate the flow transformation. Subsequently, the prediction is refined by estimating and successively transforming dominant motion patterns. More recently, EpicFlow [23] and DeepFlow [30] have been proposed, achieving excellent performance. The first method is based on the idea of coarse to fine energy-based flow refinement that interpolates on a set of matches in an edge-preserving fashion. Similarly, DeepFlow relies on a deep matching algorithm to obtain point-wise correspondences and on an energy minimization framework that enforces smoothness among the computed flow field. These methods have the advantage of not relying on specific, dataset-dependent training and thus are better at generalizing; they suffer nonetheless from high computational complexities that result in high execution times.

B. End-to-End Methods

More recently, deep learning based models outperformed traditional approaches on public benchmarks. In their seminal work, *FlowNet*, Fischer *et al.* [9] introduce one of the first end-to-end deep architectures for dense optical flow. Using an conv-deconv network, they address the lack of large datasets by creating a synthetic image collection featuring random chairs flying over random landscapes. Authors show the surprising generalization capability of the proposed model when inference is performed over real-world sequences from a different domain. Building on their insights, a plethora of methods relying on deep neural networks to approach the problem have been proposed [2], [3], [15], [26], [28]. Some of the drawbacks of this initial method have been addressed in FlowNet 2.0, a much deeper network achieving performance that have marked the state of the art. While these methods obtain good performance in an end-to-end fashion, they rely on large synthetic datasets to obtain a ground truth to be used during their training. To deal with this issue, several works address flow estimation from an unsupervised or self-supervised perspective. Long *et al.* [21] reformulate the problem as image interpolation and matching. Training a network to learn the interpolation between two frames – i.e. to obtain image X_t from X_{t-1} and X_{t+1} – they show how performing per-pixel backpropagation at test time results into sensitivity maps from where pixel level matches can be obtained. However, this method suffers of high computational cost: despite relying on modern GPU architectures, authors show how obtaining the flow for an image pair requires n/s backward passes, where n is the total number of image pixels and s is an arbitrary stride controlling the sparseness of the resulting map. In [35] another unsupervised model is illustrated, employing a similar architecture but embedding into its loss function brightness constancy and motion smoothness constraints, first introduced in [13]. Surprisingly, this method achieves state of the art performance on the Kitti dataset when compared to other unsupervised approaches, showing the benefits of leveraging the representation power of deep learning and traditional image approaches. More recently, Liu *et al.* [20] obtain the flow field between adjacent frames by learning to interpolate adjacent frames through a differentiable volume sampling layer, that learns whether to sample from the previous or next frame in order to better deal with occluding and disappearing pixels. While traditionally supervised methods achieve better performance, all the aforementioned methods are united by their being end-to-end and hence requiring a single forward pass through a neural network (with the exception of [21]), hence being very fast.

C. Composed Strategies

Finally, methods that rely on a composition of different components hold the current state of the art

on different datasets [2], [14], [29], [31]–[34]. For example, Yamaguchi *et al.* [33] propose a method that performs superpixel segmentation, boundary and occlusion labeling, stereo estimation and computes the flow by using this information jointly. Conversely, Bai *et al.* [2] propose to exploit semantic information to compute the flow of background and foreground objects separately. They exploit a convolutional architecture to obtain a segmentation of the scene and rely on deep matching to obtain the flow field of individual objects and the background. Wulff *et al.* [31] propose instead to rely on a lower level segmentation input, namely a rigidity estimation map. By initializing their flow computation using a standard method and strong geometrical priors and estimating a binary segmentation of rigid versus non-rigid objects using a convolutional neural network, they achieve good performance thanks to their ability to adapt the flow computation to different objects. Intuitively, this class of models achieves state-of-the-art performances, but suffers from a high complexity resulting in high execution times.

In this picture, we collocate our proposal in the end-to-end category, aiming at achieving both competitive performance and the execution times of a forward only neural network, which is suitable for an automotive scenario. Furthermore, the proposed method is fully self-supervised thus not requiring extensive ground truth flow information which is an especially rare commodity in the automotive setting.

III. Model

Estimating a dense flow between two adjacent frames X_t, X_{t+1} is an inherently difficult problem. Indeed, for each pixel in X_t, two translation components u, v need to be estimated, resulting in a transformation with $2 \times h \times w$ parameters (where h, w represent the height and width of the two frames respectively). In what follows, we advocate for the existence of recurrent camera movements in the automotive scenario, according to which the motion field of a static scene can be roughly estimated through a projective transformation. This reduces the dimensionality of the output space dramatically, since the parameters characterizing this projection drop to nine.

A. Projective Bootstrap

It is well studied that the motion field in the case of rigid structures can be decomposed in two contributions, namely plane and parallax contributions [16], [19], [25], [27]. This is the case of automotive settings where the camera is roof mounted on a moving vehicle: Indeed, a global motion affects the majority of scene points, and deviations are mainly due to moving objects in the scene. The plane-parallax decomposition holds in presence of roto-translational transformations between two cameras. Specifically, we consider the case of a pure translation between two cameras C and C', described by the vector $T = \{t_x, t_y, t_z\}$. The geometrical setup is depicted in Fig. 2. In this particular setting, the projection of the motion field into the image plane of C moves along a pencil of lines originating from the epipole that is the focus of expansion of the motion direction. Given a 3D point $P = (X, Y, Z)$, we define p and

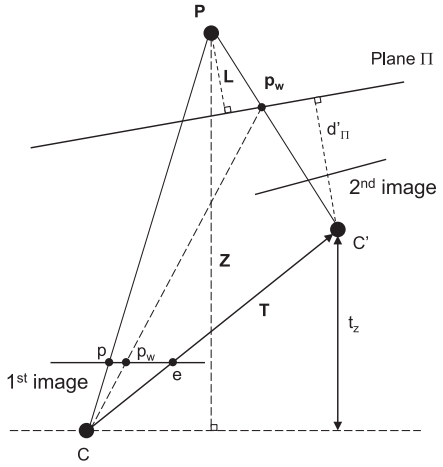


Fig. 2. The plane-parallax setting in presence of a camera translation T.

p' its projection in the image plane of C and C' respectively. By taking an auxiliary plane Π, the homography matrix $\hat{H}$ warps the point p' from C' image plane to C in p_w. The motion field projection onto C image plane can be computed in homogeneous coordinate for p as (see [16] for details):

$$p' - p = \underbrace{(p' - p_w)}_{planar} - \underbrace{\gamma \frac{t_z}{d'_\pi}(p_w - e)}_{parallax} \tag{1}$$

where d'_π is the distance of C' from plane Π, $\gamma = \frac{L}{Z}$ is the projective structure of P w.r.t. plane Π and e is the epipole location in the camera C image plane. By recalling $p_w = \hat{H}p'$ the first term of equation (1) is the planar component of the motion field since it depends only on the choice of the reference plane Π. The term $\gamma \frac{t_z}{d'_\pi}(p_w - e)$ instead, accounts for the parallax effect that is not negligible for points that do not lie on Π and it is directly modulated by L and suppressed by Z distances. When using the four points method [12] for computing the matrix $\hat{H}$ point correspondences must be observable on both image planes. This constraint is also required by the approach in [31] for optical flow initialization.

Nevertheless, a degenerate configuration for which a projective transformation can simultaneously model both the parallax and planar component can be found by considering the plane Π that pass through camera C and is parallel to its image plane. Moving the plane toward C makes the two terms $\frac{L}{Z}$ and $\frac{t_z}{d'_\pi}$ close to 1 independently from the point P. Eq. (1) then simplifies to

$$p' - p = (p' - p_w) - 2(p_w - e) \tag{2}$$

and, since $p_w = \hat{H}p'$, can be written as

$$p = 2\hat{H}p' - e. \tag{3}$$

It is important to notice that the aforementioned plane cannot be computed from image points correspondences because all P_w points are projected to the C image plane at infinite. In the case of pure translation along the Z axis, the epipole location is the image center (i.e. $e = [0, 0, 1]^T$) and can be neglected. Oppositely, in the more general setting the position of the

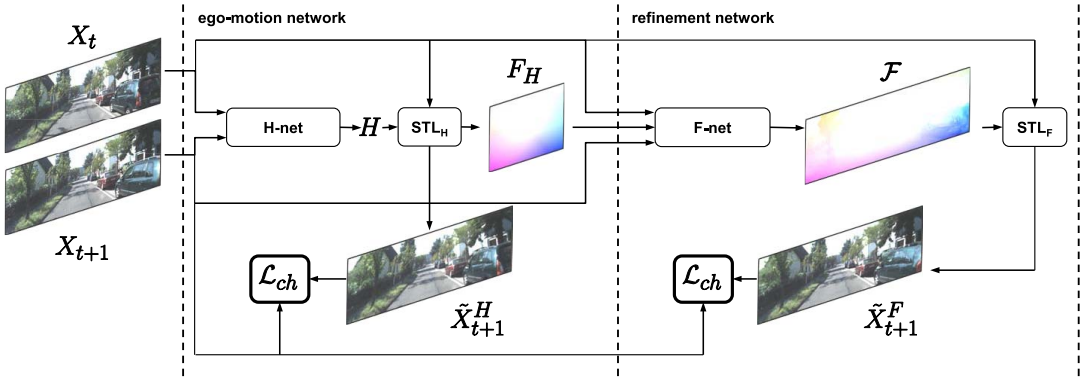


Fig. 3. Architecture of the proposed model. Frames X_t and X_{t+1} are fed into a first ego-motion network that estimates a projective transformation. STL_H then warps X_t accordingly, resulting in the estimated frame $\tilde{X}^H_{t+1}$. The bootstrapped flow is then fed to a second network along with input frames, and its refinement $\mathcal{F}$ is employed to obtain a second reconstruction, $\tilde{X}^F_{t+1}$. Both reconstructions are guided by a Charbonnier penalty $\mathcal{L}_{ch}$ to approximate X_{t+1}. During inference, the output of F-net, $\mathcal{F}$, is the predicted optical flow.

epipole needs to be estimated, and then embedded in the third column of $\hat{H}$.[1]

Eventually we can estimate the motion field by a 3×3 matrix of parameters H as

$$p' - p = p' - Hp'. \tag{4}$$

B. Architecture and Optimization

In order to leverage the projective bootstrap described in Sec. III-A, our architecture estimates the motion field in two sequential steps. First, a shallow network provides a projective transformation embedding a motion field approximating the one of a static scene in presence of camera motion (that we refer to as homographic flow). Then, a second deeper network is responsible for its refinement, and for the recovery of depth and local fields belonging to moving objects. Before introducing these two components, we briefly illustrate how a STL can be employed to train an optical flow estimator by self-supervision.

Given a geometric transformation T_ϕ estimated from two frames X_t, X_{t+1} by a generic parametric function (e.g. a neural network), $T_\phi(X_t)$ can be computed by means of a STL [17]. Being fully differentiable, this layer also allows to propagate the gradient of a given reconstruction loss to the transformation estimator. In our setting, T_ϕ embeds optical flow vectors while $\tilde{X}_{t+1} = T_\phi(X_t)$ results in the estimate of frame X_{t+1} under the transformation T_ϕ. During training, our model is tasked to optimize a Charbonnier reconstruction penalty, expressed as

$$\mathcal{L}_{ch} = \sqrt{(\tilde{X}_{t+1} - X_{t+1})^2 + \epsilon}, \tag{5}$$

where ϵ is a small regularization constant (fixed to 0.1 in our experiments). The Charbonnier penalty is a differentiable version of the l_1 norm and guides the model towards better reconstructions, i.e. better optical flow estimates (see Fig. 5). Moreover, this loss penalizes more errors within textured regions or object boundaries, while being more permissive in homogeneous portions of the reconstructed image.

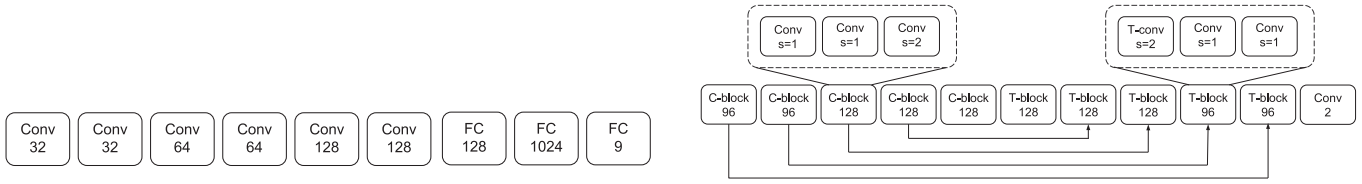
Notably, this framework requires no supervision other than the frame X_{t+1} itself, enabling the training of our model without ground-truth fields.

We design an end-to-end network, Fig. 3, composed of two main modules that, as previously introduced, first bootstraps the flow on rigid structures of the scene (ego-motion network) and subsequently refines the computation accounting for moving objects and errors in the estimation (refinement network).

1) Ego-Motion Network: The bootstrap is demanded to a shallow architecture, from now on named H-net, providing the nine coefficients of the projective transformation H (Sec. III-A) that better describes the ego-motion of the car. Fig. 4 (a) reports the details of H-net architecture. The network itself is small sized, featuring 6 convolutional layers and 3 fully connected layers resulting in 600k parameters overall. H-net is fed with two subsequent frames stacked on the channel dimension and outputs the 9 parameters of the transformation, which are input to the spatial transformer layer STL_H, that warps Xt into $\tilde{X}^H_{t+1}$. The projective transformation affects the whole image and embeds a global, coarse flow field that can be recovered from STL_H. More precisely, given a uniform sampling grid G holding column-wise pixel homogeneous coordinates, STL_H applies the input transformation to obtain the warped grid $\tilde{G} = H \cdot G$. Since $\tilde{G}$ holds the warped coordinates of each pixel, the flow field can seamlessly be recovered as $F_H = \tilde{G} - G$. As the operations on G in STL_H are differentiable, it is then possible to propagate error gradients through the layer towards H-net.

2) Refinement Network: The flow field produced by H-net cannot cope with details and moving objects. Therefore we refine it employing a second, deeper network (F-net) inspired by the model in [9]. Fig 4 (b) reports the details of F-net hourglass architecture. The encoder consists of five layer blocks, each of which is composed by three 3×3 convolutional layers with leaky ReLU activations. All convolutions in a block share unary strides except for the last one, which has stride 2. The decoder mirrors the encoder structure with

[1] those terms are multiplied by 1 when p' is expressed in projective coordinates $p' = (i, j, 1)$ thus adding a bias independent from p' location in the image



(a) (b)

Fig. 4. (a) H-net. All convolutional layers have stride $s = 2$ and all layers feature leaky ReLU activations, except for the top linearly activated fully connected module. (b) F-net. Layers in the same block share the number of output features. All activations are leaky ReLU. Convolutional layers feature a 3x3 kernel (both in H-net and F-net), whereas transposed convolutions feature 4x4 kernel. Best viewed on screen and zoomed in.

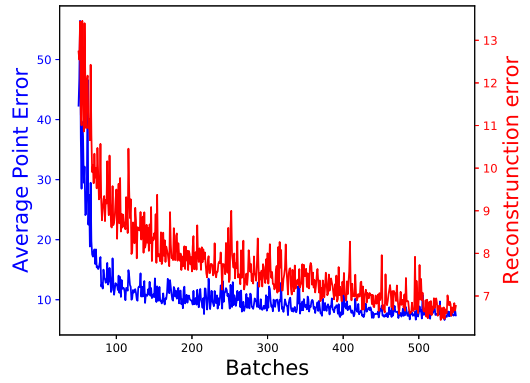


Fig. 5. Average point error and reconstruction loss during training on Virtual Kitti. Note that the first 50 batches are omitted in favor of a more readable plot scale.

transposed-convolution blocks, and a top 2-channel convolution layer produces the final estimate in the range $[-1, 1]$ using tanh as the non-linearity. F-net is fed with a channel-wise concatenation of X_t, X_{t+1} and F_H, and outputs a dense flow map $\mathcal{F} \in \mathbb{R}^{2\times w\times h}$. Hence, the spatial transformer layer STL_F warps X_t into $\tilde{X}^F_{t+1}$ and the reconstruction error w.r.t. the frame X_{t+1} is minimized by means of the Charbonnier loss (Eq. (5)). This guides F-net towards focusing on moving objects and fine details neglected by the projective bootstrap.

IV. Experimental Results

A. Datasets and Training

Due to the self-supervised nature of our method, we do not require ground truth flow information to train our network. For this reason, we are able to employ large-scale automotive datasets such as Kitti raw [11] and DR(eye)VE [1]. In particular, the Kitti raw dataset includes 44,000 frames acquired in the city of Karlsruhe, while DR(eye)VE features 555,000 frames including steep changes in image conditions due to transitions between day and night, sun, rain and a significant variance in scenarios such as highway, downtown or countryside. Notice how the size of the aforementioned datasets is suitable for training a neural network; on the other hand, the biggest real-world automotive dataset including ground truth flow information nowadays is Kitti Flow [11], that combines less that 800 annotated pairs split between training and testing in its two versions. Recently, a synthetic automotive dataset inspired by Kitti has been released [10].

TABLE I

Results Obtained by the Different Components of Our Network on Kitti Training Set

Method	Acc@5	APE	Time (s)
H-net	0.717	4.39	0.012
F-net	0.471	7.72	0.046
Joint	0.866	3.13	0.051

The Virtual Kitti dataset features more than 21,000 frames fully annotated with optical flow, semantic segmentation, depth and object bounding boxes. Due to its recent release, a state of the art on the Virtual Kitti dataset is not yet established, nonetheless we include it in our evaluation to show the generalization capabilities of our method when challenged with different automotive scenarios.

1) Training Details: To train our network, we build a set of image pairs sampled from the Kitti raw dataset; when not specifically mentioned otherwise, no fine-tuning has been performed on the individual datasets. We train the network using the Adam optimizer [18] with default parameters except for the learning rate (10^{-4} in our setting) and β_1 that was set to 0.5. Mini-batch size was set to 16, and training was stopped after 250 epochs, each of which was composed of 1000 mini-batches. The loss function of Eq. 5 is employed during the training by weighting the reconstruction errors of H-net and F-net as follows:

$$\mathcal{L}_{ch} = \mathcal{L}_{ch}(\tilde{X}^H_{t+1}, X_{t+1}) \times \alpha + \mathcal{L}_{ch}(\tilde{X}^F_{t+1}, X_{t+1}) \times \beta \quad (6)$$

where α and β are set to 0.5 and 1 respectively. Concerning the baseline proposed in Tab. II, the F-net has been trained using the same setup just described for our full architecture, but substituting the H-net with a traditional computation of the homography matrix based on SIFT+RANSAC.

In order to show the effectiveness of the proposed loss function in the context of optical flow estimation, we report in Fig. 5 the APE and reconstruction loss during training. For the purpose of this particular experiment, the Virtual Kitti dataset has been employed since it provides both enough data to train the network and dense ground-truth flow fields to compute the APE. Notably, the two metrics follow the same trend and minimizing the reconstruction error of Eq. 6 effectively decreases the flow error. All the reported times for our method are computed using a NVIDIA GTX 1080 Ti GPU for forward propagation.

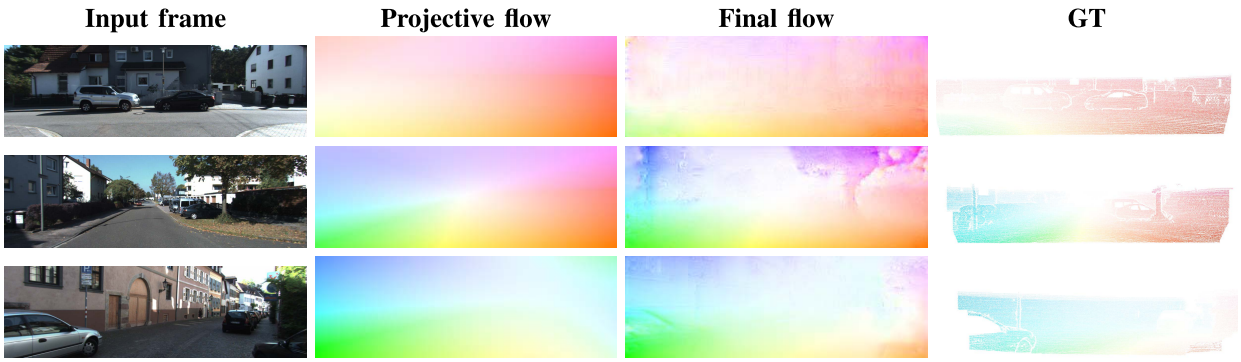


Fig. 6. Some estimated optical flow fields on the Kitti 2012 dataset.

TABLE II

PERFORMANCE COMPARISON ON THE KITTI FLOW 2012 DATASET. NOTE THAT NOT ALL THE METHODS REPORT RESULTS ON THE PUBLIC LEADERBOARDS (TESTING SET). THE TABLE IS DIVIDED IN THREE SECTIONS: HAND-CRAFTED AND SUPERVISED METHODS, UNSUPERVISED/SELF-SUPERVISED, OUR PROPOSAL. *(ft)* INDICATES THE MODEL FINE-TUNED ON KITTI FLOW

Method	Training		Testing (noc)		Testing (all)		
	Acc@5	APE	Acc@5	APE	Acc@5	APE	Time (s)
FlowNetS [9]	-	-	0.759	5.0	0.673	9.1	0.08
DeepFlow [30]	-	-	0.953	1.5	0.851	5.8	17.0
EpicFlow [23]	-	-	0.946	1.5	0.871	3.8	15.0
SpyNet [22]	0.851	4.0	0.831	4.7	0.739	10.0	0.16
SpyNet (ft) [22]	-	-	0.916	2.0	0.842	4.1	0.16
FlowNet 2.0 [15]	-	4.1	-	-	-	-	0.12
FlowNet 2.0 (ft)[15]	-	1.3	-	1.8	-	-	0.12
Long *et al.* [21]	0.716	4.7	-	-	-	-	486
Yu *et al.* [35]	-	4.3	0.779	4.6	0.681	11.3	0.03
Liu *et al.* (ft)[20]	-	-	-	-	-	9.5	-
SIFT+F-net	0.725	5.9	-	-	-	-	0.28
Ours	0.866	3.1	0.842	3.6	0.759	7.8	0.05

B. Evaluation: Kitti 2012

Following standard optical flow evaluation benchmarks, to evaluate the performance of our method we adopt the following metrics: Accuracy@5, meaning the ratio of motion vectors with end point error lower than 5 pixels, and APE which is the average point error of all motion vectors. Note that [21] and [35] only provide Accuracy@5 results instead of Accuracy@3 and do not have entries on the public benchmark, hence the use of Accuracy@5 as accuracy measure.

Tab. I reports the performance of the different components of our architecture: First, the H-net and F-net were trained independently optimizing Eq. 5, then the whole network was trained end-to-end by minimizing Eq. 6 (*Joint* entry in the table). It can be noticed how the performance of F-net is poor compared to both H-net and joint training: this is not surprising, since F-net shares similarities with the FlowNetS architecture [9], which requires large amounts of ground truth information to achieve good performance. On the other hand, due to the predominant component of homographic flow in automotive scenes, H-net in its simplicity can achieve performance comparable to unsupervised methods such as [21] and [36] and even outperform the original FlowNetS model (Tab. II) in as little as 12 ms for a forward propagation. Finally, the end-to-end training of our full architecture clearly shows an improvement over both of its two components, validating the idea of applying a geometrical bootstrap to optical flow estimation. It is also worth noticing that the computational impact of the H-net is negligible when comparing the forward prop times of F-net and joint architecture. Some qualitative results illustrating the behavior of both network modules are reported in Fig. 6.

We then evaluate our proposal against three recent self-supervised deep-learning based approaches [20], [21], [35]. We also report the results of recent methods tackling optical flow estimation using both hand-crafted and learned features [9], [23], [30]. The performance of a baseline where the homography matrix between the two frames is computed using traditional computer vision techniques (SIFT+RANSAC) instead of using the H-net is also evaluated, aiming at showing the actual contribution of the H-net. Finally, to show the effectiveness of the projective bootstrap strategy we test the flow performance of H-net and F-net separately, and then their performance when jointly trained. Tab. II reports the results of this evaluation. In particular, our proposal performs favorably against recent self-supervised approaches. It shows competitive performance and while not reaching the average point error results of DeepFlow [30] or EpicFlow [23], it is three orders of magnitude faster, requiring 0.05 sec/pair compared to the 17 sec/pair and 15 sec/pair of DeepFlow and EpicFlow respectively, which is an important factor for an automotive method. Furthermore, we compare our results to more recent deep-learning based methods who can achieve comparable running-times to ours,

TABLE III

PERFORMANCE COMPARISON ON THE VIRTUAL KITTI DATASET

Method	Acc@5	APE	Time (s)
SpyNet [22]	0.314	23.622	0.16
DeepFlow [30]	0.725	8.091	16.5
FlowNet 2.0 [15]	0.709	9.108	0.12
EpicFlow [23]	0.810	7.448	15.3
MRFlow [31]	0.833	9.284	170
Ours	0.745	7.627	0.05
Ours (ft)	0.777	6.770	0.05

TABLE IV

PERFORMANCE COMPARISON ON THE DR(EYE)VE DATASET (HIGHER IS BETTER). THE EVALUATION IS PERFORMED ON *Downtown* SEQUENCES OF THE DATASET

Method	PSNR				SSIM
	Night	Rain	Day	Avg	Avg
DeepFlow [30]	23.99	18.45	19.02	20.49	0.878
FlowNet 2.0 [15]	22.65	19.44	20.02	20.70	0.859
EpicFlow [23]	24.58	18.67	19.44	20.90	0.875
Ours	34.82	27.91	29.33	30.69	0.913

namely Flownet 2.0 [15] and SpyNet [22]. Tab. II shows that our method outperforms both of these recent approaches when no fine-tuning is performed. On the other hand, both Flownet 2.0 and SpyNet considerably improve their performance when fine-tuning on Kitti training set. This result supports our claim that our method, thanks to its self-supervised nature, should be used when ground-truth flow maps are unavailable.

Concerning the baseline performance, it can be seen how the method, while achieving acceptable results, performs worse than bootstrapping with the transformation learned by the H-net. Indeed, while the transformation learned by the H-net is inspired by this process, the experiment shows that learning the transformation instead of relying on a purely geometrical homography estimate result in an approximation that is better suited to bootstrap the subsequent flow refinement. A second advantage of our method over this baseline is that the entire process is end to end and only requires a forward propagation, thus eliminating the burdensome extraction and matching of SIFT keypoints, which results in our approach being more than 5 time faster than the baseline.

C. Evaluation: Virtual Kitti

To further evaluate our model, we assess its performance on the Virtual Kitti dataset. While still featuring the typical automotive perspective, Virtual Kitti presents some noteworthy differences from the other datasets, the most significant being the presence of the typical artifacts of computer rendered scenes. Nonetheless, it is currently the biggest dataset providing dense ground truth optical flow for automotive, obtained using the computer graphics warping of the scene and is hence guaranteed to be exact, as opposed to LIDAR based ground truth measures. Due to the dataset being very recent, no results are publicly available and we evaluate DeepFlow, EpicFlow, FlowNet 2.0, SpyNet and MRFlow [31] relying on the provided source codes and pretrained models. In all the experiments, the methods have been used with their default parameters, and no fine-tuning has been performed on the Virtual Kitti ground-truth fields. For [31], we used the publicly available code with the rigidity maps computed using semantic segmentation ground truth and default flow initializer. Tab. III provides the results of this evaluation. It can be noticed how the performance of all the methods, except for SpyNet, falls into a similar range. Nonetheless, EpicFlow and our proposal are shown to outperform the other competitors. The failure of SpyNet is this scenario is likely tied to the synthetic nature of the dataset, requiring either fine-tuning or a thorough hyper-parameter search. Finally, the last line of Tab. III reports the performance of our model after a fine-tuning step (note that our fine-tuning does not require optical flow ground truth, and is hence always possible). We obtain the best APE results compared to the other methods while still being 2.5× faster than the fastest competitor. Indeed, the self-supervised nature of our proposal does not require optical flow ground-truth information, making fine-tuning suitable for every dataset. Fig. 7 displays a qualitative evaluation of the results on the Virtual Kitti dataset.

D. Evaluation: DR(Eye)VE

To evaluate the proposed approach on the DR(eye)VE dataset, which lacks ground truth flow information, we rely on our initial assumption that the optical flow can be computed as the transformation warping two consecutive frames. It is hence possible to approximately estimate the optical flow quality by estimating the reconstruction itself. That is, given a frame X_t and the optical flow transformation T_ϕ, we use the spatial transformer layer to get the reconstructed estimate $\tilde{X}^F_{t+1}$ and measure its peak signal-to-noise ratio (PSNR) according to the protocol in [20]. Due to PSNR sensibility to small changes resulting, for example, from small flow intensity errors shifting objects by a few pixels but keeping the overall scene intact, we also evaluate the quality of $\tilde{X}^F_{t+1}$ using the Structural Similarity Index Measure (SSIM), which not only considers individual pixel intensities but also accounts for the structural similarity of small patches. Tab. IV reports the results of such evaluation where no fine-tuning has been performed. To analyze the impact of different environmental conditions on optical flow estimation, Tab. IV also reports results divided by sequence type. The slightly higher PSNR across all methods during the Night sequences is due to the lower intensity of the images, effectively resulting in a lower error. Beside that, the experiment shows that environmental conditions do not have a significant impact on optical flow estimation. SSIM, while confirming the same overall trend, shows a smaller gap between methods. As mentioned earlier, this is due to SSIM being a more stable metric where the coherence of pixel neighborhoods are also taken into account. Note that using the spatial transformer layer as warper ensures that the reconstruction pipeline is the same for every method and differences in PSNR are only due to the flow.

E. Sintel: a Non-Homographic Flow Case Study

To assess the generalization capability of our model to non-automotive contexts, we design a case study where our network is tested on a dataset featuring strong non-homographic components in its motion patterns: the MPI Sintel flow dataset [6]. It presents highly dynamic scenes

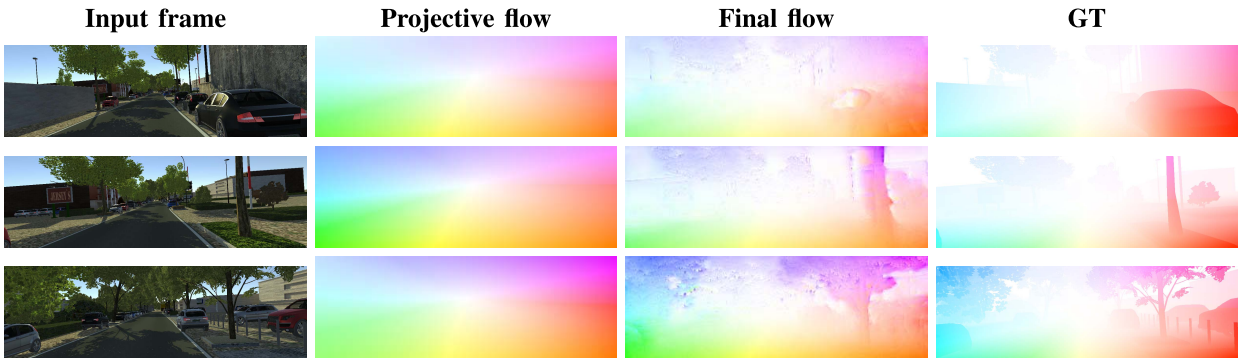


Fig. 7. Examples of flow fields estimated by the proposed model on the Virtual Kitti dataset.

TABLE V
PER-SEQUENCE APE ON SINTEL TRAINING SET. NOTE THAT *All* REPORTS THE PER-FRAME APE, AND DIFFERS FROM THE MEAN OF OTHER COLUMNS DUE TO DIFFERENT SEQUENCE LENGTHS

Alley	Ambush	Bamboo	Bandage	Cave	Market	Mountain	Shaman	Sleeping	Temple	All
5.25	28.74	4.30	5.06	24.05	15.82	7.123	4.61	4.55	9.22	11.72

Fig. 8. Examples our predicted flow in two different sequences. In each triplet, from left to right: blend between frame t and $t+1$, ground truth, prediction. First triplet: *bamboo*, ape 3.93; second triplet: *cave*, image pair ape: 19.75.

often dominated by large non-rigid objects in the foreground with very articulated motion, e.g. the protagonist fighting an assailant or a dragon. Tab. V reports the performance of our method on the Sintel training set[2] (*Final* pass) where no fine-tuning has been performed (i.e. with the model trained on the Kitti raw sequences). To provide a better picture of the results, the table reports the APE for each sequence. Notably, the proposed model is able to generalize from the automotive scenario to several Sintel sequences. Nonetheless, some sequences (such as *ambush, cave, market*) exhibit high average point errors (APE$\geq$10); this is not surprising, as these sequences feature the largest displacements and a strong component of non-homographic flow. Fig. 8 depicts two examples extracted from two different sequences: The first sequence depicts a scene where the motion is dominated by the camera motion and the network correctly identifies the flow between the input frames. In the other sequence it can be noticed how the input pair contains severe motion blur and two large non-rigid foreground objects, preventing the correct projective bootstrap of the flow and hence resulting in a failure.

V. CONCLUSIONS

In this paper, we showed a self-supervised approach to optical flow estimation that jointly accounts for the geometric cues in a scene, and for the pixel-level motion patterns of different objects. We address the complexity of self-supervised training by first estimating the global motion of the car using a lightweight network that bootstraps a more complex, pixel-level transformation. Our experimental evaluation shows how the proposed approach outperforms recent self-supervised methods while maintaining the advantages in terms of simplicity and speed of an end-to-end forward only neural network. In fact, we strongly believe that, especially in motion flow estimation, obtaining ground truth information can be prohibitive and self-supervision will in turn become a key component of any computer vision pipeline relying on flow information.

[2]Note that the submission system does not provide per-sequence information, hence the use of the training sequences

REFERENCES

[1] S. Alletto, A. Palazzi, F. Solera, S. Calderara, and R. Cucchiara, "DR(eye)VE: A dataset for attention-based tasks with applications to autonomous and assisted driving," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun./Jul. 2016, pp. 54–60.
[2] M. Bai, W. Luo, K. Kundu, and R. Urtasun, "Exploiting semantic information and deep matching for optical flow," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 154–170.
[3] C. Bailer, K. Varanasi, and D. Stricker. (Jul. 2016). "CNN-based patch matching for optical flow with thresholded hinge embedding loss." [Online]. Available: https://arxiv.org/abs/1607.08064
[4] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, 2008.
[5] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2004, pp. 25–36.
[6] D. J. Butler *et al.*, "A naturalistic open source movie for optical flow evaluation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, in Lecture Notes in Computer Science, A. Fitzgibbon, Eds. Berlin, Germany: Springer-Verlag, Oct. 2012, pp. 611–625.
[7] Z. Chen, H. Jin, Z. Lin, S. Cohen, and Y. Wu, "Large displacement optical flow from nearest neighbor fields," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2443–2450.
[8] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[9] P. Fischer *et al.* (Apr. 2015). "FlowNet: Learning optical flow with convolutional networks." [Online]. Available: https://arxiv.org/abs/1504.06852

[10] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Proc. CVPR*, 2016, pp. 4340–4349.

[11] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2012, pp. 3354–3361.

[12] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[13] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, nos. 1–3, pp. 185–203, Aug. 1981.

[14] J. Hur and S. Roth, "MirrorFlow: Exploiting symmetries in joint optical flow and occlusion estimation," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 312–321.

[15] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. (Dec. 2016). "FlowNet 2.0: Evolution of optical flow estimation with deep networks." [Online]. Available: https://arxiv.org/abs/1612.01925

[16] M. Irani and P. Anandan, "Parallax geometry of pairs of points for 3D scene analysis," in *Computer Vision—ECCV*. Berlin, Germany: Springer, 1996, pp. 17–30.

[17] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.

[18] D. Kingma and J. Ba. (Dec. 2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[19] R. Kumar, P. Anandan, and K. Hanna, "Direct recovery of shape from multiple views: A parallax based approach," in *Proc. 12th IAPR Int. Conf. Pattern Recognit., Conf., Comput. Vis. Image Process.*, vol. 1. Oct. 1994, pp. 685–688.

[20] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2017, pp 4463–4471.

[21] G. Long, L. Kneip, J. M. Alvarez, H. Li, X. Zhang, and Q. Yu, "Learning image matching by simply watching video," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 434–450.

[22] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2. Jul. 2017, pp. 4161–4170.

[23] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "EpicFlow: Edge-preserving interpolation of correspondences for optical flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1164–1172.

[24] S. R. Richter, Z. Hayder, and V. Koltun, "Playing for benchmarks," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2213–2222.

[25] H. S. Sawhney, "3D geometry from planar parallax," in *Proc. CVPR*, vol. 94. 1994, pp. 929–934.

[26] T. Schuster, L. Wolf, and D. Gadot. (Nov. 2016). "Optical flow requires multiple strategies (but only one network)." [Online]. Available: https://arxiv.org/abs/1611.05607

[27] A. Shashua and N. Navab, "Relative affine structure: Theory and application to 3D reconstruction from perspective views," in *Proc. CVPR*, vol. 94. 1994, pp. 483–489.

[28] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Deep end2end voxel2voxel prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2016, pp. 17–24.

[29] A. S. Wannenwetsch, M. Keuper, and S. Roth, "ProbFlow: Joint optical flow and uncertainty estimation," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, vol. 3. Oct. 2017, pp. 1173–1182.

[30] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "DeepFlow: Large displacement optical flow with deep matching," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Sydney, NSW, Australia, Dec. 2013, pp. 1385–1392.

[31] J. Wulff, L. Sevilla-Lara, and M. J. Black, "Optical flow in mostly rigid scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4671–4680.

[32] J. Xu, R. Ranftl, and V. Koltun. (Apr. 2017). "Accurate optical flow via direct cost volume processing." [Online]. Available: https://arxiv.org/abs/1704.07325

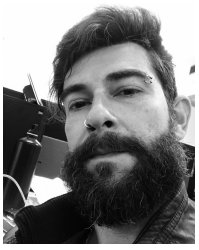
[33] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 756–771.

[34] Y. Yang and S. Soatto, "S2F: Slow-to-fast interpolator flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2087–2096.

[35] J. J. Yu, A. W. Harley, and K. G. Derpanis. (Aug. 2016). "Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness." [Online]. Available: https://arxiv.org/abs/1608.05842

Stefano Alletto received the master's degree in computer engineering and the Ph.D. degree from the University of Modena and Reggio Emilia in 2014 and 2018, respectively, where he is currently a Research Contractor. His main research interests include computer vision and machine learning applied to first person camera views, automotive and enhanced cultural experiences.

Davide Abati received the master's degree in computer engineering from the University of Modena and Reggio Emilia in 2015. He is currently pursuing the Ph.D. degree with the AImageLab Group, Modena, with a focus on computer vision and deep learning for image and video understanding.



Simone Calderara (S'04–M'05) received the master's degree in computer engineering and the Ph.D. degree from the University of Modena and Reggio Emilia, in 2005 and 2009 respectively and, where he is currently an Assistant Professor with the AImagelab Group. His current research interests include computer vision and machine learning applied to human behavior analysis, visual tracking in crowded scenarios, and time series analysis for forensic applications.

Rita Cucchiara received the master's degree in electronic engineering and the Ph.D. degree in computer engineering from the University of Bologna, Italy, in 1989 and 1992, respectively. Since 2005, she has been a Full Professor at the University of Modena and Reggio Emilia, Italy, where she is also the Head of the ImageLab Group and is also the Director of the SOFTECH-ICT Research Center. She is currently the President of the Italian Association of Pattern Recognition, affiliated with IAPR. She has authored over 300 papers on pattern recognition computer vision and multimedia, and in particular in human analysis, HBU, and egocentric-vision. The research carried out spans different application fields, such as video-surveillance, automotive and multimedia big data annotation. She is currently an Associate Editor for the IEEE TRANSACTIONS ON MULTIMEDIA and serves on the Governing Board of IAPR and on the Advisory Board of the CVF.

Luca Rigazio received the B.S. degree in electrical engineering from the Politecnico di Torino and the M.S. degree in computer engineering from Institut Eurecom, with a focus on signal processing and machine learning. He started his career at the Panasonic Speech Technology Laboratory, Santa Barbara, CA, USA, where he was involved in low complexity speech recognition models and real-time decoders. He then moved to the Panasonic Silicon Valley Laboratory, the company's premiere research center, Bay Area, CA, USA, where he started the deep learning groups and the robotics groups, with a focus on computer vision and end-to-end robotics control, with a direct application to the company's autonomous drive solutions. He has over 40 patents and over 35 publications in the areas of machine learning, deep learning, signal processing, and control for domains, such as computer vision, speech processing, digital signal processing, and robots control.