Warp Consistency for Unsupervised Learning of Dense Correspondences

Supplementary Material

Prune Truong Martin Danelljan Fisher Yu Luc Van Gool
Computer Vision Lab, ETH Zurich, Switzerland
{prune.truong, martin.danelljan, vangool}@vision.ee.ethz.ch i@yf.io

In this supplementary material, we give additional details about our approach, experiment settings and results. We first give additional details about the flow-constraints derived from our introduced warp consistency graph in Sec. A. We also provide additional empirical comparisons between the corresponding regression losses. We follow by explaining the triplet image creation and the sampling process of our synthetic warps W in Sec. B. In Sec. C, we then focus on the training procedure to obtain WarpC-GLU-Net in more depth. We subsequently continue by explaining the training details of WarpC-RANSAC-Flow and WarpC-SemanticGLU-Net in respectively Sec. D and E. For completeness, in Sec. F, we also provide details about the training of all networks compared in the method analysis, corresponding to Sec. 4.1 of the main paper. In all aforementioned sections, we provide additional information about the architecture, its original training strategy, our proposed training approach comprising the sampled transformations W, as well as implementation details. We then follow by analysing the effect of the strength of the sampled warps W in Sec. G. In Sec. H, we follow by discussing the time and memory efficiency of our proposed approach during training and testing. Subsequently, we perform additional ablative and method analysis experiments in Sec. J. In Sec. I, we extensively explain the evaluation datasets and set-up. Finally we present more detailed quantitative and qualitative results in Sec. K. In particular, we show quantitative results on the pose estimation dataset YFCC100M [39] as well as the geometric matching dataset HPatches [1]. We also provide results on the semantic datasets PF-Willow [8] and SPair-71k [29]. Finally, we show the possible extension of our unsupervised approach to optical flow data.

A. Warp consistency graph regression losses

In this section, we provide additional details about the possible flow-constraints derived from our warp consistency graph (Sec. 3.3 of the main paper). We also show qualitative and quantitative comparisons between the trained networks using each possible regression loss.

A.1. Details about JI-bipath constraint

We here provide the detailed derivation of the bias insensitivity of the JI-bipath loss, which is given by (eq. (6) in the main paper) as,

$$L_{J\to I} = \left\| \widehat{F}_{J\to I'} + \Phi_{\widehat{F}_{J\to I'}}(W) - \widehat{F}_{J\to I} \right\|. \tag{1}$$

We derive an upper bound for the change in the loss $\Delta L_{J\to I}$ when a constant bias $\mathbf{b} \in \mathbb{R}^2$ is added to all flow predictions $\widehat{F}$. We have,

$$\begin{aligned}
\Delta L_{J\to I} &= \left\| \widehat{F}_{J\to I'} + \mathbf{b} + \Phi_{\widehat{F}_{J\to I'}+\mathbf{b}}(W) - (\widehat{F}_{J\to I} + \mathbf{b}) \right\| \\
&\quad - \left\| \widehat{F}_{J\to I'} + \Phi_{\widehat{F}_{J\to I'}}(W) - \widehat{F}_{J\to I} \right\| \\
&= \left\| \widehat{F}_{J\to I'} + \Phi_{\widehat{F}_{J\to I'}}(W) - \widehat{F}_{J\to I} \right. \\
&\qquad \left. + \Phi_{\widehat{F}_{J\to I'}+\mathbf{b}}(W) - \Phi_{\widehat{F}_{J\to I'}}(W) \right\| \\
&\quad - \left\| \widehat{F}_{J\to I'} + \Phi_{\widehat{F}_{J\to I'}}(W) - \widehat{F}_{J\to I} \right\| \\
&\le \left\| \widehat{F}_{J\to I'} + \Phi_{\widehat{F}_{J\to I'}}(W) - \widehat{F}_{J\to I} \right\| \\
&\quad + \left\| \Phi_{\widehat{F}_{J\to I'}+\mathbf{b}}(W) - \Phi_{\widehat{F}_{J\to I'}}(W) \right\| \\
&\quad - \left\| \widehat{F}_{J\to I'} + \Phi_{\widehat{F}_{J\to I'}}(W) - \widehat{F}_{J\to I} \right\| \\
&= \left\| \Phi_{\widehat{F}_{J\to I'}+\mathbf{b}}(W) - \Phi_{\widehat{F}_{J\to I'}}(W) \right\| .
\end{aligned} \tag{2}$$

Here we have used the triangle inequality. From the bound above, we can already see that $\Delta L_{J\to I}$ will be small if W is changing slowly. We can see this more clearly by assuming the bias $\mathbf{b}$ to be small, and doing a first order Taylor expansion,

$$\begin{aligned}
\Phi_{\widehat{F}_{J\to I'}+\mathbf{b}}(W)(\mathbf{x}) &= W\big(\mathbf{x} + \widehat{F}_{J\to I'}(\mathbf{x}) + \mathbf{b}\big) \\
&\approx W\big(\mathbf{x} + \widehat{F}_{J\to I'}(\mathbf{x})\big) + DW\big(\mathbf{x} + \widehat{F}_{J\to I'}(\mathbf{x})\big)\mathbf{b} \\
&= \Phi_{\widehat{F}_{J\to I'}}(W)(\mathbf{x}) + \Phi_{\widehat{F}_{J\to I'}}(DW\mathbf{b})(\mathbf{x}) .
\end{aligned} \tag{3}$$

Here, $DW(\mathbf{x}) \in \mathbb{R}^{2\times 2}$ is the Jacobian of W at location $\mathbf{x} \in \mathbb{R}^2$. Thus, $DW\mathbf{b}$ denotes the function obtained from the matrix-vector product between the Jacobian DW and bias $\mathbf{b}$ at every location. Inserting (3) into (2) gives an approximate bound valid for small $\mathbf{b}$,

$$\Delta L_{J \to I} \lessapprox \left\| \Phi_{\widehat{F}_{J \to I'}}(DW\mathbf{b}) \right\| . \tag{4}$$

A smooth and invertible warp W implies a generally small Jacobian DW. Since the bias $\mathbf{b}$ is scaled with DW, the resulting change in the loss will also be small. As a spacial case, it is immediately seen from (2) that the change in the loss is always zero if W is a pure translation. The bias insensitivity of the JI-bipath constraint largely explains its poor performance. As visualized in Fig. 1, the predictions of a network trained with solely the JI-bipath loss (1) suffer from a large translation bias.

A.2. Cycle constraints

Here, we provide additional details about the cycle constraints, extracted from our warp consistency graph. As explained in Sec. 3.3 of the main paper, because of the fixed direction of the known flow W which corresponds to $I' \to I$, three cycle constraints are possible, starting from either images I, I' or J and composing mappings so that the resulting composition is equal to the identity map. They are respectively formulated as follows,

$$\mathbb{I} = M_W \circ M_{J \to I'} \circ M_{I \to J} \tag{5a}$$

$$\mathbb{I} = M_{J \to I'} \circ M_{I \to J} \circ M_W \tag{5b}$$

$$\mathbb{I} = M_{I \to J} \circ M_W \circ M_{J \to I'} \tag{5c}$$

The corresponding regression losses are obtained by converting the mapping constraints (5) to flow constraints and considering only the flow W as known. We provide the expression for each of the three cycle losses in the following.

Cycle from I: By starting from image I and performing a full cycle, the resulting regression loss is expressed as,

$$L_{\text{cycle-I}} = \Big\| \widehat{F}_{I \to J} + \Phi_{\widehat{F}_{I \to J}}(\widehat{F}_{J \to I'}) + \Phi_{\widehat{F}_{I \to J} + \Phi_{\widehat{F}_{I \to J}}(\widehat{F}_{J \to I'})}(W) \Big\| \tag{6}$$

Cycle from I': Starting from image I' instead leads to the following regression loss,

$$L_{\text{cycle-I'}} = \Big\| W + \Phi_W(\widehat{F}_{I \to J}) + \Phi_{W + \Phi_W(\widehat{F}_{I \to J})}(\widehat{F}_{J \to I'}) \Big\| \tag{7}$$

Cycle from J: Finally, using image J as starting point for the cycle constraint results in this regression loss,

$$L_{\text{cycle-J}} = \Big\| \widehat{F}_{J \to I'} + \Phi_{\widehat{F}_{J \to I'}}(W) + \Phi_{\widehat{F}_{J \to I'} + \Phi_{\widehat{F}_{J \to I'}}(W)}(\widehat{F}_{I \to J}) \Big\| \tag{8}$$

	MegaDepth			**RobotCar**			**HPatches**	
	PCK-1	PCK-5	PCK-10	PCK-1	PCK-5	PCK-10	AEPE	PCK-5
Warp-supervision (3 m.p.)	35.98	57.21	63.88	2.43	33.63	54.50	28.50	76.76
$I'J$-bipath (4a m.p)	0.00	0.05	0.21	0.00	0.00	0.13	370.80	0.01
JI-bipath (4b m.p),(6 m.p)	0.00	0.06	0.21	0.00	0.05	0.21	162.50	0.04
W-bipath (4c m.p),(7 m.p)	**29.55**	**67.70**	**74.42**	**2.25**	**33.88**	**55.38**	**26.13**	**70.51**
I'-cycle (7)	25.04	64.44	71.75	2.19	32.79	54.55	27.51	66.16
I-cycle (6)	0.00	0.14	0.56	0.03	0.74	5.29	232.24	0.04
J-cycle (8)	17.91	54.95	62.81	2.05	30.96	52.06	42.67	49.06
$I'J$-bipath + warp-sup.	0.00	0.11	0.45	0.01	0.35	1.52	255.40	0.02
JI-bipath + warp-sup.	33.72	61.10	67.44	2.26	34.06	55.07	28.91	71.52
W-bipath + warp-sup.	**43.47**	**69.90**	**75.23**	2.49	35.28	56.45	**22.83**	**78.60**
I'-cycle + warp-sup.	42.11	68.84	74.28	**2.52**	**35.75**	**56.96**	24.16	78.58
I-cycle + warp-sup.	0.00	0.16	0.69	0.05	1.26	4.67	225.94	0.04
J-cycle + warp-sup.	41.56	68.33	73.85	2.37	35.20	56.36	24.69	75.35
Warp-super. + f-b	41.54	69.78	74.83	2.47	35.25	56.39	26.15	74.83

Table 1. Analysis of warp consistency graph losses (Sec. 3.3-3.4 of the main paper). 'm.p' refers to equations in the main paper.

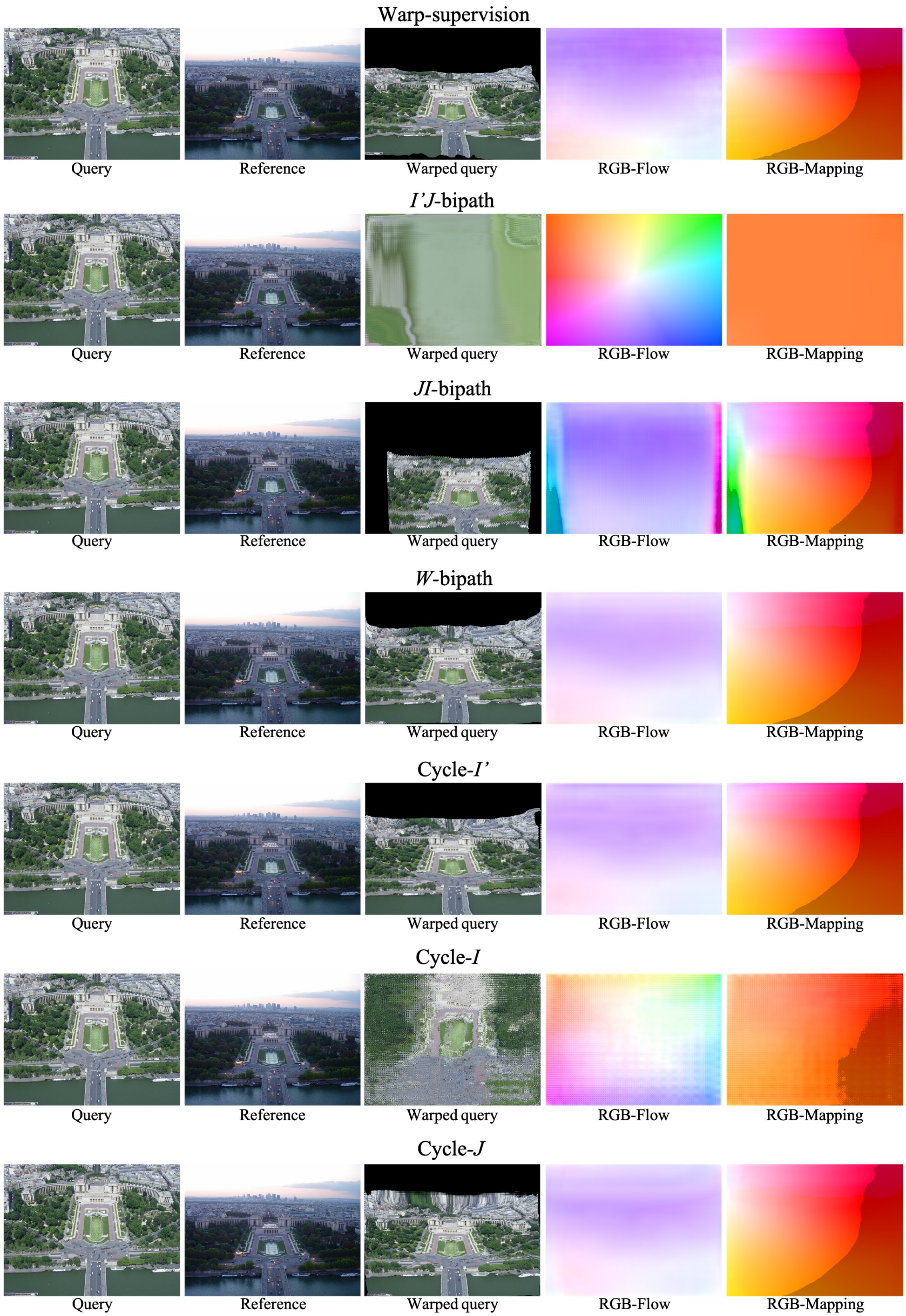
Note about warp consistency loss: Concerning the adaptive loss balancing of our final warp consistency unsupervised objective $L = L_{\text{W-vis}} + \lambda L_{\text{warp}}$ (Sec. 3.5 of the main paper), since λ is a weighting factor, we do not backpropagate gradients through it.

A.3. Quantitative and qualitative analysis

Extension of quantitative analysis: We first extend Tab. 1 of the main paper, by analysing the remaining warp consistency graph losses. Results on MegaDepth, RobotCar and HPatches are presented in Tab. 1. As in Tab. 1 of the main paper, all networks are trained following the first training stage of WarpC-GLU-Net (See Sec. 4.1 of main paper or Sec. C).

We first provide evaluation results of networks trained using the cycle losses, starting from images I and J. The cycle loss starting from I obtains very poor results. The cycle starting from J instead achieves better performance, but still lower than the cycle loss from I'. The W-bipath constraint obtains the best results overall.

We then compare combinations of the derived losses with the warp-supervision objective (eq. 3 of the main paper). Between the cycle losses, the combination of the warp-supervision with the cycle loss from I' achieves the best results compared to the combinations with the cycle losses from I and J. The combination of the warp-supervision and forward-backward losses (eq. 2 of the main paper), which are both retrieved as pair-wise constraints from the warp consistency graph (Sec. 3.3 and Fig. 4e of the main paper), leads to lower generalisation abilities on the HPatches dataset than our warp consistency loss. It also achieves substantially lower PCK-1 on MegaDepth. Moreover, because the forward-backward consistency loss leads to a degenerate trivial solution when used alone, manual tuning of a weighting hyper-parameter is required to balance the warp-supervision and the forward-backward loss terms. If it is too high, the forward-backward term gains too much importance and drives the network to zero. If it is too small instead, its contribution becomes insignificant. On the contrary, our proposed unsupervised learning objective (Sec. 3.5 of the main paper) does not require expensive



Warp-supervision

I'J-bipath

JI-bipath

W-bipath

Cycle-*I'*

Cycle-*I*

Cycle-*J*

Figure 1. Visual comparison on an image pairs of the MegaDepth dataset, of the performance of the different losses derived from the warp-consistency graph. We additionally show for each loss, the estimated mapping and flow using flow RGB representation. Note that all networks are trained following the first training stage of WarpC-GLU-Net (See Sec. 4.1 of main paper or Sec. C).

manual tuning of such hyperparameters.

Qualitative comparison: In Fig. 1, we visually compare the estimated flows by GLU-Net networks trained using each of the flow-consistency losses retrieved from the warp consistency graph. Training using the warp-supervision loss alone results in an unstable estimated flow, and corresponding warped query. It can directly be seen that the $I'J$-bipath loss results in the network learning a degenerate trivial solution, in the form of a constant predicted mapping independently of the input images. Training with the JI-bipath objective instead makes the network insensitive to an additional predicted bias. Indeed, in Fig. 1, third row, it is easily seen that the warped query is shifted towards the right and bottom, compared to the reference image. This is due to a constant predicted bias by the network. The W-bipath objective leads to a drastically better warped query. Also note that the estimated flow leads to a more accurate warped query than when trained with the I'-cycle loss. Training with the cycle loss from I leads to very poor results instead. Finally, the cycle loss derived by starting from image J results in a reasonable warped query, but it has more out-of-regions artifacts compared to the prediction of the network trained with the W-bipath loss.

B. Triplet creation and sampling of warps W

B.1. Triplet creation

Our introduced unsupervised learning approach requires to construct an image triplet (I, I', J) from an original image pair (I, J), where all three images must have training dimensions $s \times s$. We construct the triplet (I, I', J) as follows. The original training image pairs (I, J) are first resized to a fixed size $s_r \times s_r$, larger than the desired training image size $s \times s$. We then sample a dense flow W of the same dimension $s_r \times s_r$, and create I' by warping image I with W, as $I' = \Phi_W(I)$. Each of the images of the resulting image triplet (I, I', J) are then centrally cropped to the fixed training image size $s \times s$. The central cropping is necessary to remove most of the black areas in I' introduced from the warping operation with large sampled flows W as well as possible warping artifacts arising at the image borders. We then additionally apply appearance transformations to the created image I', such as brightness and contrast changes. This procedure is similar to [31], which employs solely the warp-supervision objective on (I', I).

B.2. Sampling of warps W

As mentioned in the main paper Sec. 3.6, we key question raised by our proposed loss formulation is how to sample the synthetic flows W. The analysis of the properties of the proposed W-bipath loss brought some insight into what magnitude $\|W\|$ of warps to sample during training. If the generated warps are too small, there is still a risk of biasing the prediction towards zero. Instead, using warps of roughly similar order of magnitude $\|W\|$ as the underlying transformations $\|F_{J\to I}\|$ would give equal impact to all three terms in eq. 8 of the main paper. During training, we randomly sample it from a distribution $W \sim p_W$, which we need to design.

Base transformation sampling: We construct W by sampling homography, Thin-plate Spline (TPS), or affine-TPS transformations with equal probability. The transformations parameters are then converted to dense flows of dimension $s_r \times s_r$.

Specifically, for homographies and TPS, the four image corners and a 3×3 grid of control points respectively, are randomly translated in both horizontal and vertical directions, according to a desired sampling scheme. The translated and original points are then used to compute the corresponding homography and TPS parameters. Finally, the transformations parameters are converted to dense flows. For both transformation types, the magnitudes of the translations are sampled according to a uniform or Gaussian distribution with a range or standard-deviation σ_H respectively. Note that for the uniform distribution, the sampling range is actually $[-\sigma_H, \sigma_H]$, when it is centered at zero, or similarly $[1 - \sigma_H, 1 + \sigma_H]$ if centered at 1 for example. Importantly, the image points coordinates are previously normalized to be in the interval $[-1, 1]$. Therefore σ_H should be within $[0, 1]$.

For the affine transformations, all parameters, *i.e.* scale, translations, shearing and rotation angles, are sampled from a uniform or Gaussian distribution with range or standard-deviation equal to τ, t, α and α respectively. For the affine scale parameter, the corresponding Gaussian sampling is centered at one whereas for all other parameters, it is centered at zero. Similarly, for a uniform sampling instead, the affine scale parameters is sampled within $[1 - \tau, 1 + \tau]$ with center at 1, while for all other parameters, the sampling interval is centered at zero.

Elastic transformations: To make the synthetic flow W harder for the network to estimate, we also optionally compose the base flow resulting from sampling homography, TPS and Affine-TPS transformations, with a dense elastic deformation grid. We generate the corresponding elastic residual flow $\epsilon = \sum_i \varepsilon_i$, by adding small local perturbations $\varepsilon_i \in \mathbb{R}^{s_r \times s_r \times 2}$. More specifically, we create the residual flow by first generating an elastic deformation motion field E on a dense grid of dimension $s_r \times s_r$, as described in [37]. Since we only want to include elastic perturbations in multiple small regions, we generate binary masks $S_i \in \mathbb{R}^{s_r \times s_r}$, each delimiting the area on which to apply one local perturbation ε_i. The final elastic residual flow thus take the form of $\epsilon = \sum_i \varepsilon_i$, where $\varepsilon_i = E \cdot S_i$. The final synthetic warp W is achieved by composing the base flow with the elastic residual flow ϵ.

In practise, for the elastic deformation field E, we use the implementation of [3]. The masks S_i should be between 0 and 1 and offer a smooth transition between the two, so that the perturbations appear smoothly. To create each mask S_i, we thus generate a 2D Gaussian centered at a random location and with a random standard deviation (up to a certain value) on a dense grid of size $s_r \times s_r$. It is then scaled to 2.0 and clipped to 1.0, to obtain smooth regions equal to 1.0 where the perturbations will be applied, and transition regions on all sides from 1.0 to 0.0.

B.3. Hyper-parameters

In summary, to construct our image triplet (I, I', J), the hyper-parameters are the following:

(i) s_r, the resizing image size, on which is applied W to obtain I' before cropping.

(ii) s, the training image size, which correspond to the size of the training images after cropping.

(iii) σ_H, the range or standard deviation used for sampling the homography and TPS transformations.

(iv) τ, the range or standard deviation used for sampling the scaling parameter of the affine transformations.

(v) t, the range or standard deviation used for sampling the translation parameter of the affine transformations.

(vi) α, the range or standard deviation used for sampling the rotation angle of the affine transformations. It is also used as shearing angle.

(vii) σ_{tps}, the range or standard deviation used for sampling the TPS transformations, used for the Affine-TPS compositions.

For simplicity, in all experiments including elastic deformations, we use the same elastic transformations hyper-parameters. Moreover, for all experiments and networks, we apply the same appearance transformations to image I'. Specifically, we use color transformations, by adjusting contrast, saturation, brightness, and hue. With probability 0.2, we additionally use a Gaussian blur with a kernel between 3 and 7, and a standard deviation sampled within $[0.2, 2.0]$.

C. Training details for WarpC-GLU-Net

We first provide details about the original GLU-Net architecture and the modifications we made for this work. We also briefly review the training strategy of the original work. We then extensively explain our training approach and the corresponding implementation details.

C.1. Details about GLU-Net

Architecture: We use GLU-Net as our base architecture. It is a 4 level pyramidal network, using a VGG-16 feature backbone [4], initialized with pre-trained weights on ImageNet. It is composed of two sub-networks, L-Net and H-Net which act at two different resolutions. The L-Net takes as input rescaled images to $h_L \times w_L = 256 \times 256$ and process them with a global feature correlation layer followed by a local feature correlation layer. The resulting flow is then upsampled to the lowest resolution of the H-Net to serve as initial flow, by warping the query features according to the estimated flow. The H-Net takes as input images the original images at unconstrained resolution $h \times w$, and refines the estimated flow with two local feature correlation layers. We adopt the GLU-Net architecture and simply replace the DenseNet connections [14] of the flow decoders by residual connections. We also include residual blocks in the mapping decoder. This drastically reduces the number of weights while having limited impact on performance.

Training strategy in original work: In the original GLU-Net [40], the network is trained with the warp-supervision loss (referred to as a type of self-supervised training strategy in original publication), which corresponds to equation (3) of the main paper. As for the synthetic sampled transformations W, Truong *et al*. [40] use the same 40k synthetic transformations (affine, thin-plate and homographies) than in DGC-Net [28], but apply them to images collected from the DPED [16], CityScapes [5] and ADE-20K [48] datasets.

C.2. WarpC-GLU-Net: our training strategy

We here explain the different steps of our training strategy in more depth.

Training stages: In the first training stage, we train GLU-Net using our warp consistency loss (Sec. 3.5 of the main paper) without the visibility mask. This is because the estimated flow field needs to reach a reasonable performance in order to compute the visibility mask (eq. 9 of the main paper). In the second training stage, we further introduce the visibility mask in the W-bipath loss term (eq. 8 of the main paper). In order to enhance difficulty in the second stage, we increase the transformations strengths and include additional elastic transformations for the sampled warps W. Note that the feature backbone is initialized to the ImageNet weights and not further trained.

Training dataset: For training, we use the MegaDepth dataset, consisting of 196 different scenes reconstructed from 1,070,468 internet photos using COLMAP [34]. Specifically, we use 150 scenes of the dataset and sample up to 500 random images per scene. It results in around 58k training pairs. Note that we use the same set of training pairs at each training epoch. For the validation dataset, we sample up to 100 image pairs from 25 different scenes, leading to approximately 1800 image pairs. Importantly, while we can get the corresponding sparse ground-truth correspondences from the SfM reconstructions, we do not use them during training in this work and only retrieve the image pairs.

Warps W sampling: We resize the image pairs (I, J) to $s_r \times s_r = 750 \times 750$, sample a dense flow W of the same dimension and create I'. Each of the images of the resulting image triplet (I, I', J) is then centrally cropped to $s \times s = 520 \times 520$. In the following, we give the parameters used for the sampling of the flow W in both training stages.

In the first stage, the flows W are created by sampling homographies, TPS and Affine-TPS transformations with equal probability. For homographies and TPS, we use a uniform sampling scheme with a range equal to $[-\sigma_H, \sigma_H]$, where $\sigma_H = 0.33$, which corresponds to a displacement of up to 250 pixels for the image size $s_r = 750$. For the affine transformations, we also sample all parameters, *i.e.* scale, translation, shear and rotation angles, from uniform distributions with ranges respectively equal to $\tau = 0.45$, $t = 0.25$, and $\alpha = \pi/12$ for both angles. We compose the affine transformations with TPS transformations, for which we sample the translation magnitudes uniformly with a range $\sigma_{tps} = 0.08$, thus corresponding to a displacement of up to 60 pixels. We chose a smaller range for the TPS compositions because we have found empirically that large ranges led to very drastic resulting dense Affine-TPS flows, which were not necessarily beneficial in the first training stage.

In the second stage, we also sample homographies, TPS and Affine-TPS transformations, but increase their strength. Specifically, for homography and TPS transformations, we use a range $\sigma_H = 0.4$ (displacements up to 300 pixels). The affine parameters are sampled as in the first training stage, but we increase the range of the uniform sampling for the TPS transformations to $\sigma_{tps} = 0.26$ (displacements up to 200px). To make the flows W even harder to estimate, we additionally include elastic transformations, sampled as explained in Sec. B.

Baseline comparison: For fair comparison, we retrain GLU-Net using the original training strategy, which corresponds to the warp-supervision training loss, on the same MegaDepth training images. We also use the same altered GLU-Net architecture as for WarpC-GLU-Net. Moreover, we make use of the same synthetic transformations W as in our first and second training stages. We call this version GLU-Net*.

C.3. Implementation details

Since GLU-Net is a pyramidal architecture with K levels, we employ a multi-scale training loss, where the loss at different pyramid levels account for different weights.

$$\mathcal{L}(\theta) = \sum_{l=1}^{K} \gamma_l L_l + \eta \left\| \theta \right\| , \tag{9}$$

where γ_l are the weights applied to each pyramid level and L_l is the corresponding loss computed at each level, which refers to the warp-supervision loss (eq. 3 of the main paper) for baseline GLU-Net* and our proposed warp consistency loss (Sec. 3.5 of the main paper) for WarpC-GLU-Net. The second term of the loss (9) regularizes the weights of the network. The hyper-parameters used in the estimation of our visibility mask $\widehat{V}$ (eq. 9 of the main paper) are set to $\alpha_1 = 0.025$ and $\alpha_2 = 0.5$. During training, we down-sample and scale the sampled W from original resolution $h \times w$ to $h_L \times w_L$ in order to obtain the flow field W for L-Net. For the loss computation, we down-sample the known flow field W from the base resolution to the different pyramid resolutions without further scaling, so as to obtain the supervision signals at the different levels.

For training, we use similar training parameters as in [42]. Specifically, as a preprocessing step, the training images are mean-centered and normalized using mean and standard deviation of the ImageNet dataset [21]. For all local correlation layers, we employ a search radius $r = 4$.

For our network WarpC-GLU-Net and the baseline GLU-Net*, the weights in the training loss (9) are set to be $\gamma_1 = 0.32, \gamma_2 = 0.08, \gamma_3 = 0.02, \gamma_4 = 0.01$. During the first training stage, both networks are trained with a batch size of 6 for 400k iterations. The learning rate is initially equal to 10^{-4}, and halved after 250k and 325k iterations. For the second training stage, we train for 225k iteration with an initial learning rate of 5.10^{-5}, which is halved after 100k, 150k and 200k iterations. The networks are trained using Adam optimizer [20] with weight decay of 0.0004.

D. Training details for WarpC-RANSAC-Flow

In this section, we first review the RANSAC-Flow architecture as well as their original training strategy. We then explain in more depth the different steps of our training, leading to WarpC-RANSAC-Flow.

D.1. Details about RANSAC-Flow

Architecture: RANSAC-Flow inference is divided in two steps. First, the image pairs are pre-aligned by computing the homography relating them, using multi-scale feature matching based on off-the-shelf MOCO features [11] and Ransac. As a second step, the pre-aligned image pairs are input to the trained RANSAC-Flow model, which predicts the flow and matchability mask relating them. The final flow field relating the original images is computed as a composition of the flow corresponding to the homography computed in the pre-alignment step, and the predicted flow field. RANSAC-Flow is a shallow architecture taking image pairs as input, and which regresses the dense flow field and matchability mask relating one image to the other. It relies on a single local feature correlation layer computed at one eight of the input images resolution. The local feature correlation layer is computed with a small search radius of

$r = 3$. The flow decoder and matchability branch are both fully convolutional with three convolution blocks, while the feature backbone is a modified version of ResNet-18 [12].

Training dataset: As training dataset, RANSAC-Flow uses images of the MegaDepth dataset [24], from which they selected a subset of image pairs. They pre-aligned the image pairs using their pre-processing multi-scale strategy with off-the-shelf MOCO feature [11] matching and homography estimation with Ransac. The resulting training dataset comprises 20k pre-aligned image pairs, for which the remaining geometric transformation between the frames is relatively small.

Training strategy in original work: In the original work [36], the training is separated in three stages. First, the network is trained using the SSIM loss [43], which is further combined with the forward-backward cyclic consistency loss (eq. 2 of the main paper) in the second stage. During the two first stages, only the feature backbone and the flow decoder are trained, while the matchability branch remains unchanged and unused. In the last stage, the matchability branch is also trained by weighting the previous losses with the predicted mask and including a regularization matchability loss. A disadvantage of this approach is that all losses need to be scaled with a hyper-parameter, requiring expensive manual-tuning.

D.2. WarpC-RANSAC-Flow: our training strategy

Training stages: In the first training stage, we apply our proposed loss (Sec. 3.5 of the main paper) without the visibility mask, as in the first stage of WarpC-GLU-Net. The visibility mask (eq. 8 of the main paper) is introduced in the second stage of training. As in original RANSAC-Flow, the two first stages only train the feature backbone and the flow decoder while keeping the matchability branch fixed (and unused). In the third stage, we jointly train the feature backbone, flow decoder and the matchability branch. As training loss, we use the original matchability regularization loss and further replace our visibility mask $\widehat{V}$ in the W-bipath loss (eq. 8 of the main paper) with the predicted mask, output of the matchability branch.

Warps W sampling: We resize original images (I, J) to $s_r \times s_r = 300 \times 300$. Following original RANSAC-Flow, the final training images have dimension $s \times s = 224 \times 224$. Because RANSAC-Flow uses a single local correlation layer with a search radius of 3 computed at one eight of the original image resolution, the network can theoretically only estimate geometric transformations up to $3.8 = 24$ pixels in all directions. This is a very limited compared to GLU-Net or other matching networks. It makes RANSAC-Flow architecture very sensitive to the magnitude of the geometric transformations and limited in the range of displacements that it can actually estimate. It also implies that the RANSAC-Flow pre-alignement stage (with off-the-shelf feature matching and Ransac) is crucial for the success of the matching process in general. We thus need to sample transformations W within the range of the network capabilities. As a result, we construct the warps W by sampling only homographies and TPS transformations from a Gaussian distribution. This is because the Affine-TPS transformations lead to larger geometric transformations and are more difficult to parametrize for a network very sensitive to the strength of geometric transformations. The Gaussian sampling gives more importance to transformations of small magnitudes, as opposed to the uniform sampling used for WarpC-GLU-Net.

The homography and TPS transforms are sampled from a Gaussian distribution with standard deviation $\sigma_H = 0.08$, which corresponds to a displacement of 24 pixels in an image size $s_r \times s_r = 300 \times 300$. We further integrate additional elastic transformations, which were shown beneficial to boost the network accuracy. We use the above sampling scheme for all three training stages.

D.3. Implementation details

RANSAC-Flow only estimates the flow at one eight of the original image resolution. Loss computations is performed at image resolution, *i.e.* $s \times s = 224 \times 224$, after upsampling the estimated flow field. Following the original work, we also compute training losses at the image resolution. The hyper-parameters used in the estimation of our visibility mask $\widehat{V}$ (eq. 9 of the main paper) are set to $\alpha_1 = 0.01$ and $\alpha_2 = 0.5$.

For training, we use similar training parameters as in [36]. As pre-processing, we scale the input network images to $[0, 1]$. During the first training stage, WarpC-RANSAC-Flow is trained with a batch size of 10 for 300k iterations. The learning rate is initially equal to 8.10^{-4}, and halved after 200k iterations. For the second training stage, we train for 140k iteration with a constant learning rate of 4.10^{-4}. Finally, the third training stages also uses an initial learning rate of 4.10^{-4} halved after 200k iterations, and comprises a total of 300k iterations. To weight the matchability regularization loss with respect to our warp consistency loss in the third stage, we use a constant factor of 0.6 applied to the matchability loss.

E. Training details for WarpC-SemanticGLU-Net

Here, we first review the SemanticGLU-Net architecture as well as their original training strategy. We then provide additional details about our training strategy, resulting in WarpC-SemanticGLU-Net.

E.1. Details about SemanticGLU-Net

Architecture: SemanticGLU-Net is derived from GLU-Net [42], with two architectural modifications, making it more suitable for semantic data. Specifically, the global feature correlation layer is followed by a consensus network [33]. The features from the different levels in the L-Net are also concatenated, similarly to [17].

Training strategy in original work: SemanticGLU-Net was originally trained using the same procedure as GLU-Net [42]. It is explained in Sec. C.

E.2. WarpC-SemanticGLU-Net: our training strategy

Training procedure: We only finetune on semantic data, from the original pretrained SemanticGLU-Net model, initialized with the weights provided by the authors. The VGG-16 feature backbone is initialized to the ImageNet weights and not further finetuned. We use our warp consistency loss (Sec. 3.5 of the main paper), where the visibility mask $\widehat{V}$ is directly included. Note that since SemanticGLU-Net is trained using solely the warp-supervision objective, the overall training of WarpC-SemanticGLU-Net does not use any flow annotations.

Training dataset: We use the PF-Pascal [9] images as training dataset. Following the dataset split in [10], we partition the total 1351 image pairs into a training set of 735 pairs, validation set of 308 pairs and test set of 308 pairs, respectively. The 735 training images are augmented by mirroring, random cropping and exchanging the images in the pair. It leads to a total of 2940 training image pairs.

Warps W sampling: We resize the image pairs (I, J) to $s_r \times s_r = 500 \times 500$, sample a dense flow W of the same dimension and create I'. Each of the images of the resulting image triplet (I, I', J) is then centrally cropped to $s \times s = 400 \times 400$. The flows W are created by sampling homographies, TPS and Affine-TPS transformations with equal probability. For homographies and TPS, we use a uniform sampling scheme with a range equal to $[-\sigma_H, \sigma_H]$, where $\sigma_H = 0.2$, which corresponds to a displacement of 100px, in image size $s_r = 500$. For the affine transformations, we also sample all parameters, *i.e.* scale, translation, shear and rotation angles, from uniform distributions with ranges respectively equal to $\tau = 0.4$, $t = 0.25$, and $\alpha = \pi/12$ for both angles. We compose the affine transformations with TPS transformations, for which we sample the translation magnitudes uniformly with a range $\sigma_{tps} = 0.2$, thus corresponding to a displacement of 100px.

Implementation details: For our network WarpC-SemanticGLU-Net, the weights in the training loss (9) are set to $\gamma_1 = 0.32, \gamma_2 = 0.08, \gamma_3 = 0.02, \gamma_4 = 0.01$. We train with a batch size of 5, for a total of 7k iterations. The learning rate is initially equal to 8.10^{-5}, and halved after 4k, 5k and 6k iterations. The network is trained using Adam optimizer [20] with weight decay of 0.0004.

F. Training details for method analysis

For the method analysis corresponding to Sec. 4.1 of the main paper, we use as base network GLU-Net [42]. Architecture description and implementation details are explained in Sec. C. In this section, for completeness we provide additional details about the training procedure used for each of the compared networks, when necessary.

Warp consistency graph analysis: All networks are trained following the first WarpC-GLU-Net training stage, *i.e.* without including the visibility mask in the bipath or cycle losses. We employ the same warps W for all networks, which correspond to the sampling distribution used in the first training stage, as detailed in Sec. C.

Ablation study: Networks in ablation study are trained according to the stages described in Sec. C.

Comparison to alternative losses: We provide implementation details for networks trained with alternative losses. For all unsupervised learning objectives, we train the network in two stages. First, we use solely the evaluated loss, without visibility or occlusion mask. In the second stage, we further finetune the resulting model, extending the evaluated loss with the visibility mask, estimated as in [27]. For the objectives including our warp consistency loss (WarpC) or the warp-supervision loss, we use the same synthetic warp W distribution than introduced in Sec. C. In the following, we give details about each training using an alternative loss.

Warp-supervision + forward-backward: Selecting a hyper-parameter is necessary to weight the forward-backward loss with respect to the warp-supervision objective. After manual tuning, we weight the forward-backward term with a constant factor equal to 0.05. It ensures that the forward-backward term accounts for about half of the magnitude of the warp-supervision loss. For further implementation details, refer to Sec. C.

Census: The implementation details are the same than explained in Sec. C. Particularly, we found that downsampling the images to the flow resolution at each level for loss computation gave better results than upsampling the estimated flows to image resolution.

SSIM: To compute the loss, we upsample the estimated flow from each level to image resolution, *i.e.* $h \times w = 520 \times 520$ for the HNet and $h_L \times w_L = 256 \times 256$ for the LNet. This strategy led to significantly better results than downsampling the images instead. As a result, because GLU-Net is a multi-scale architecture and the loss is computed using the flow from each resolution, the weights of the final training loss (9) are set to $\gamma_1 = 0.08, \gamma_2 = 0.08, \gamma_3 = 0.01, \gamma_4 = 0.01$. This gives equal contribution to all levels, since estimated flows at levels of L-Net and H-Net are upsampled to respectively $h_L \times w_L$ and $h \times w$. SSIM is computed with a window size of 11 pixels, following RANSAC-Flow [36].

SSIM + forward-backward: The model trained using the SSIM loss is further finetuned with the combination of photometric SSIM and forward-backward consistency losses (eq. 2 of the main paper). Both loss terms are balanced with a constant factor equal to 0.1, applied to the forward-backward consistency term. It ensures that the forward-backward term accounts for about half of the magnitude of the SSIM loss. Implementation details are the same than when training with the SSIM loss only.

SSIM + WarpC: For the WarpC loss, we follow the training procedure and implementation details provided in Sec. C, *i.e.* we compute the loss at estimated flow resolution. For the SSIM loss term, we instead follow the training strategy explained above, *i.e.* we compute the loss at image resolution. For the WarpC term, the different levels weights of the final training loss (9) are set to be $\gamma_1 = 0.32, \gamma_2 = 0.08, \gamma_3 = 0.02, \gamma_4 = 0.01$, while for the SSIM loss term they are set to $\gamma_1 = 0.08, \gamma_2 = 0.08, \gamma_3 = 0.01, \gamma_4 = 0.01$. Each loss term, *i.e.* WarpC and SSIM, is computed independently and the final loss is the sum of both.

Sparse ground-truth data: Since the ground-truth is sparse, it is inconvenient to down-sample the ground-truth to different resolutions. We thus instead up-sample the estimated flow fields to the ground-truth resolution and compute the loss at this resolution. As for SSIM, we therefore use $\gamma_1 = 0.08, \gamma_2 = 0.08, \gamma_3 = 0.01, \gamma_4 = 0.01$ for the level weights of the final training loss (9).

G. Analysis of transformations W

In this section, we analyse the impact of the sampled transformations' strength on the performance of the corresponding trained WarpC networks. As explained in Sec. B, the strength of the warps W is mostly controlled by the standard-deviation or range σ_H, used to sample the base homography and TPS transformations. We thus analyse the effect of the sampling range σ_H on the evaluation results of the corresponding WarpC networks, particularly WarpC-GLU-Net and WarpC-SemanticGLU-Net. We do not provide such analysis for WarpC-RANSAC-Flow because as mentioned in Sec. D, RANSAC-Flow architecture is limited to a small range of displacements that it can estimate, which also limits the range σ_H over which we can sample the warps W.

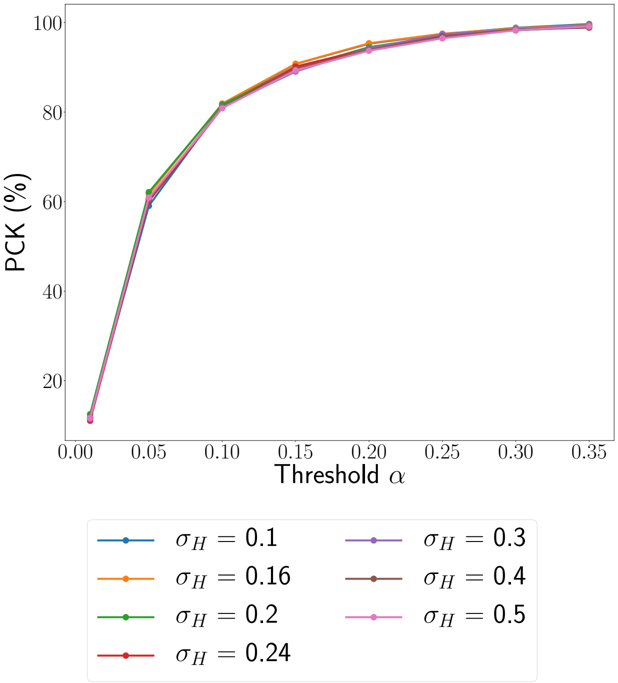


Figure 2. PCK curves obtained on the PF-Pascal [9] images by WarpC-SemanticGLU-Net, for different sampling ranges σ_H used to create the synthetic transformations W during training.

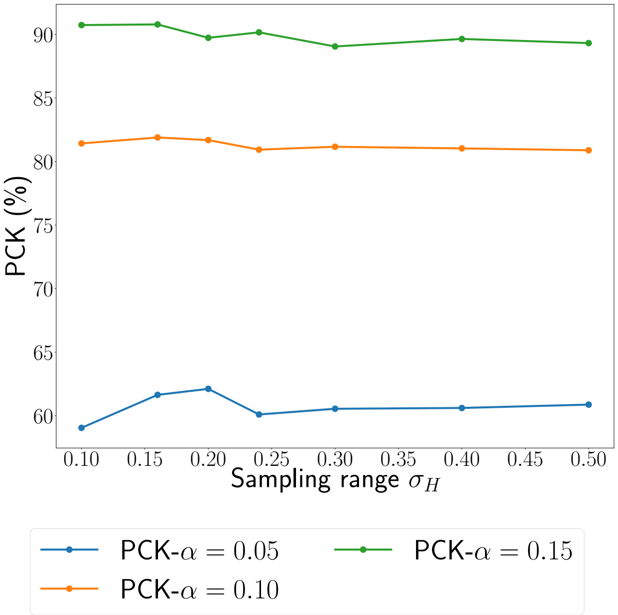
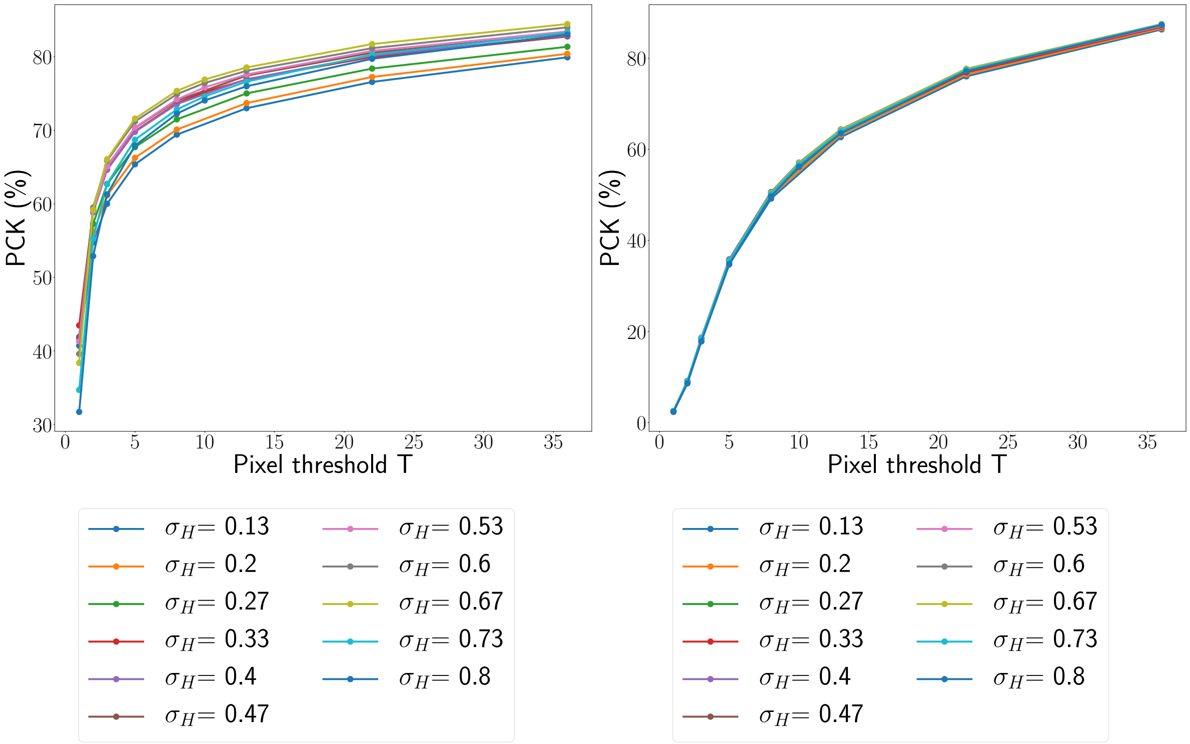


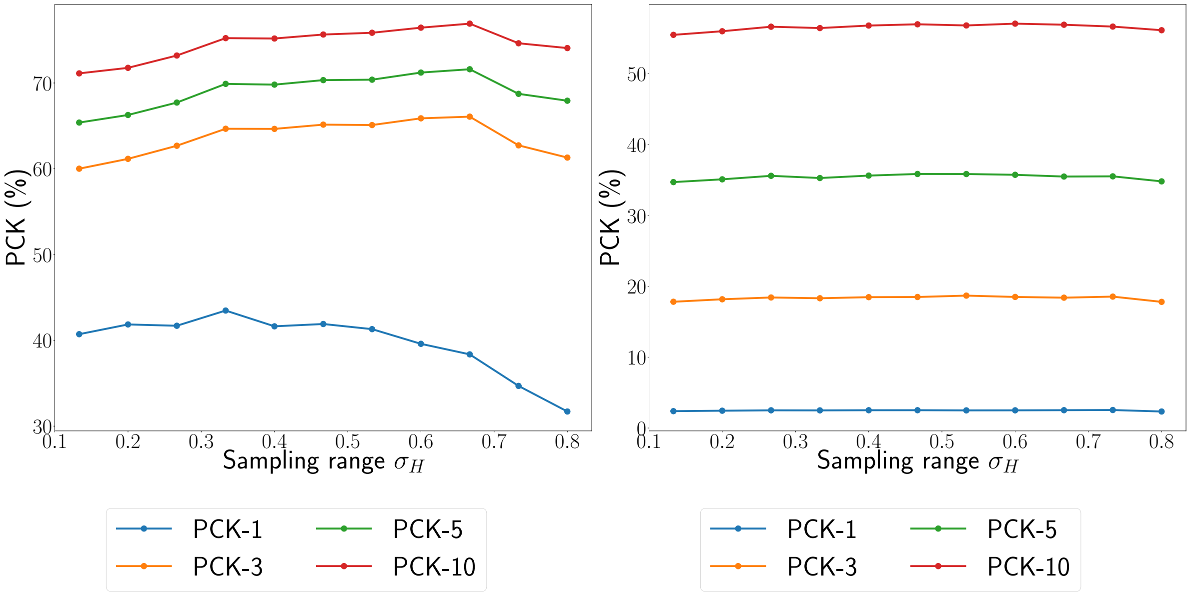
Figure 3. PCK for α thresholds in $\{0.05, 0.1, 0.15\}$ obtained on the PF-Pascal [9] images by WarpC-SemanticGLU-Net, for different sampling ranges σ_H used to create the synthetic transformations W during training.



(a) MegaDepth

(b) RobotCar

Figure 4. PCK curves obtained by GLU-Net based networks trained using our warp consistency loss, for different sampling ranges σ_H. Transformations W are sampled according to the first training stage.



(a) MegaDepth

(b) RobotCar

Figure 5. PCK for pixel thresholds in $\{1, 3, 5, 10\}$ obtained by GLU-Net based networks trained using our warp consistency loss, for different sampling ranges σ_H. Transformations W are sampled according to the first training stage.

While we choose a specific distribution p_W to sample the transformations parameters used to construct the flow W, our experiments show that the performance of the trained networks according to our proposed warp consistency loss (Sec. 3.5 of the main paper) is relatively insensitive to the strength of the transformations W, if they remain in a reasonable bound. We present these experiments below.

WarpC-GLU-Net: Specifically, we analyze the PCK curves obtained by GLU-Net based models, trained following our first training stage (Sec. C), for varying ranges σ_H used to sample the TPS and homography transformations. Note that for all networks, the sampling distributions of the affine-tps transformations are the same. We plot in Fig. 4 the resulting curves, computed on the MegaDepth and RobotCar datasets. For completeness, we additionally plot the PCK values for fixed pixel thresholds in $\{1, 3, 5, 10\}$ versus the sampling range σ_H in Fig. 5. On MegaDepth, increasing the sampling range σ_H from 0.13 to 0.67 leads to an improvement of the resulting network's robustness to large geometric transformations, *i.e*. an increase in PCK-3, 5 and 10. Further increasing σ_H up to 0.8 leads to a decrease in these PCK values. For PCK-1 however, networks trained with sampling ranges within $[0.13, 0.53]$ obtain similar accuracy. The accuracy starts dropping for larger sampling ranges. We select $\sigma_H = 0.33$ because it obtains the best PCK-1 and good PCK-3, 5 and 10. Nevertheless, note that networks trained using sampling ranges within $[0.2, 0.53]$ lead to relatively similar PCK metrics, within 2-3 %. Moreover, on RobotCar, all networks obtain similar PCK metrics, independently of the sampling range σ_H.

WarpC-SemanticGLU-Net: As for WarpC-GLU-Net, we show that the performance of WarpC-SemanticGLU-Net is relatively insensitive to the strength of the transformations W, if they remain in a reasonable bound. Specifically, we analyze the PCK curves obtained by WarpC-SemanticGLU-Net based models, for varying ranges σ_H used to sample the TPS and homography transformations of W during training. Note that for all networks, the sampling distributions of the affine-tps transformations are the same. We plot in Fig. 2 the resulting curves evaluated on the test set of PF-Pascal and in Fig. 3 the results for specific PCK values. For sampling ranges σ_H within $[0.1, 0.5]$, the results of the corresponding trained WarpC-SemanticGLU-Net are all very similar overall. Particularly, the gap between all networks for $\alpha > 0.05$ is very small, within 1 %. For $\alpha < 0.05$, differences amount to 4%. We selected $\sigma_H = 0.2$ because it led to a slightly better PCK for the low threshold $\alpha = 0.05$.

H. Time and memory efficiency

We here discuss the time and memory efficiency of our unsupervised approach during training and testing. Our loss formulation does not impact inference time and memory requirements, which remain at the baseline. For WarpC-GLU-Net, on an RTX 2080Ti GPU with 11GB, we fit 10 image pairs for warp-supervision during training and 6 image triplets for our objective. We use the same image size 520 × 520. The training times per pair/triplet are 190 ms for the former and 240 ms (×1.26) for ours. We visualize the convergence speed (validation performance) as a function of training time for both approaches in Fig. 6. While each iteration is slower, our warp consistency strategy leads to faster convergence and vastly better final performance. The overall training times of our models WarpC-GLU-Net, WarpC-RANSAC-Flow and WarpC-SemanticGLU-Net are 7, 4 and 1 days respectively.

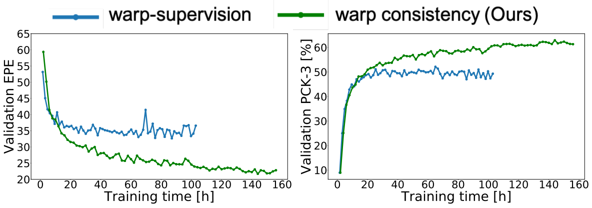


Figure 6. Validation metrics w.r.t. training time for GLU-Net.

I. Experimental setup and datasets

In this section, we first provide details about the evaluation datasets and metrics. We then explain the experimental set-up in more depth.

I.1. Evaluation metrics

AEPE: AEPE is defined as the Euclidean distance between estimated and ground truth flow fields, averaged over all valid pixels of the reference image.

PCK: The Percentage of Correct Keypoints (PCK) is computed as the percentage of correspondences $\tilde{\mathbf{x}}_j$ with an Euclidean distance error $\|\tilde{\mathbf{x}}_j - \mathbf{x}_j\| \leq T$, w.r.t. to the ground truth $\mathbf{x}_j$, that is smaller than a threshold T.

I.2. Evaluation datasets and set-up

HPatches: The HPatches dataset [1] is a benchmark for geometric matching correspondence estimation. It depicts planar scenes, with transformations restricted to homographies. As in DGC-Net [28], we only employ the 59 sequences labelled with `v_X`, which have viewpoint changes, thus excluding the ones labelled `i_X`, which only have illumination changes. Each image sequence contains a query image and 5 reference images taken under increasingly larger viewpoints changes, with sizes ranging from 450 × 600 to 1613 × 1210.

MegaDepth: The MegaDepth dataset [24] depicts real scenes with extreme viewpoint changes. No real ground-truth correspondences are available, so we use the result of SfM reconstructions to obtain sparse ground-truth correspondences. We follow the same procedure and test images

	MegaDepth			RobotCar			HPatches	
	PCK-1	PCK-5	PCK-10	PCK-1	PCK-5	PCK-10	AEPE	PCK-5
Stage1, pretrained VGG16	43.47	**69.90**	**75.23**	**2.49**	**35.28**	**56.45**	22.83	78.60
Stage1, from scratch	**43.74**	68.21	73.07	2.36	34.14	54.76	**22.75**	**81.73**
Stage2, pretrained VGG16	50.61	**78.61**	**82.94**	**2.51**	**35.92**	**57.44**	**21.00**	83.24
Stage2, from scratch	**51.16**	77.64	81.86	2.43	35.12	56.53	21.22	**83.83**

Table 2. Feature backbone training for both training stages of WarpC-GLU-Net.

than [36], spanning 19 scenes. More precisely, 1600 pairs of images were randomly sampled, that shared more than 30 points. The test pairs are from different scenes than the ones we used for training and validation. Correspondences were obtained by using 3D points from SfM reconstructions and projecting them onto the pairs of matching images. It results in approximately 367K correspondences. During evaluation, following [36], all the images and ground-truths are resized to have minimum dimension 480 pixels.

RobotCar: Images in RobotCar depict outdoor road scenes, taken under different weather and lighting conditions. While the image pairs show similar view-points, they are particularly challenging due to their many textureless regions. For evaluation, we use the correspondences originally introduced by [26]. Following [36], all the images and ground-truths are resized to have minimum dimension 480 pixels.

TSS: The TSS dataset [38] contains 400 image pairs, divided into three groups: FG3DCAR, JODS, and PASCAL, according to the origins of the images. The dense flow fields annotations for the foreground object in each pair is provided along with a segmentation mask. Evaluation is done on 800 pairs, by also exchanging query and reference images. Evaluation is done by computing PCK for a pixel threshold computed with respect to query image size.

PF-Pascal: The PF-PASCAL [9] benchmark is built from the PASCAL 2011 keypoint annotation dataset [2]. It consists of 20 diverse object categories, ranging from chairs to sheep. Sparse manual annotations are provided for 300 image pairs. Evaluation is done by computing PCK for a pixel threshold computed with respect to query image size.

PF-Willow: The PF-WILLOW dataset consists of 900 image pairs selected from a total of 100 images [8]. It spans four object categories. Sparse annotations are provided for all pairs. For evaluation, we report the PCK scores with multiple thresholds (α = 0.05, 0.10, 0.15) with respect to bounding box size in order to compare with prior methods.

J. Additional method analysis experiments

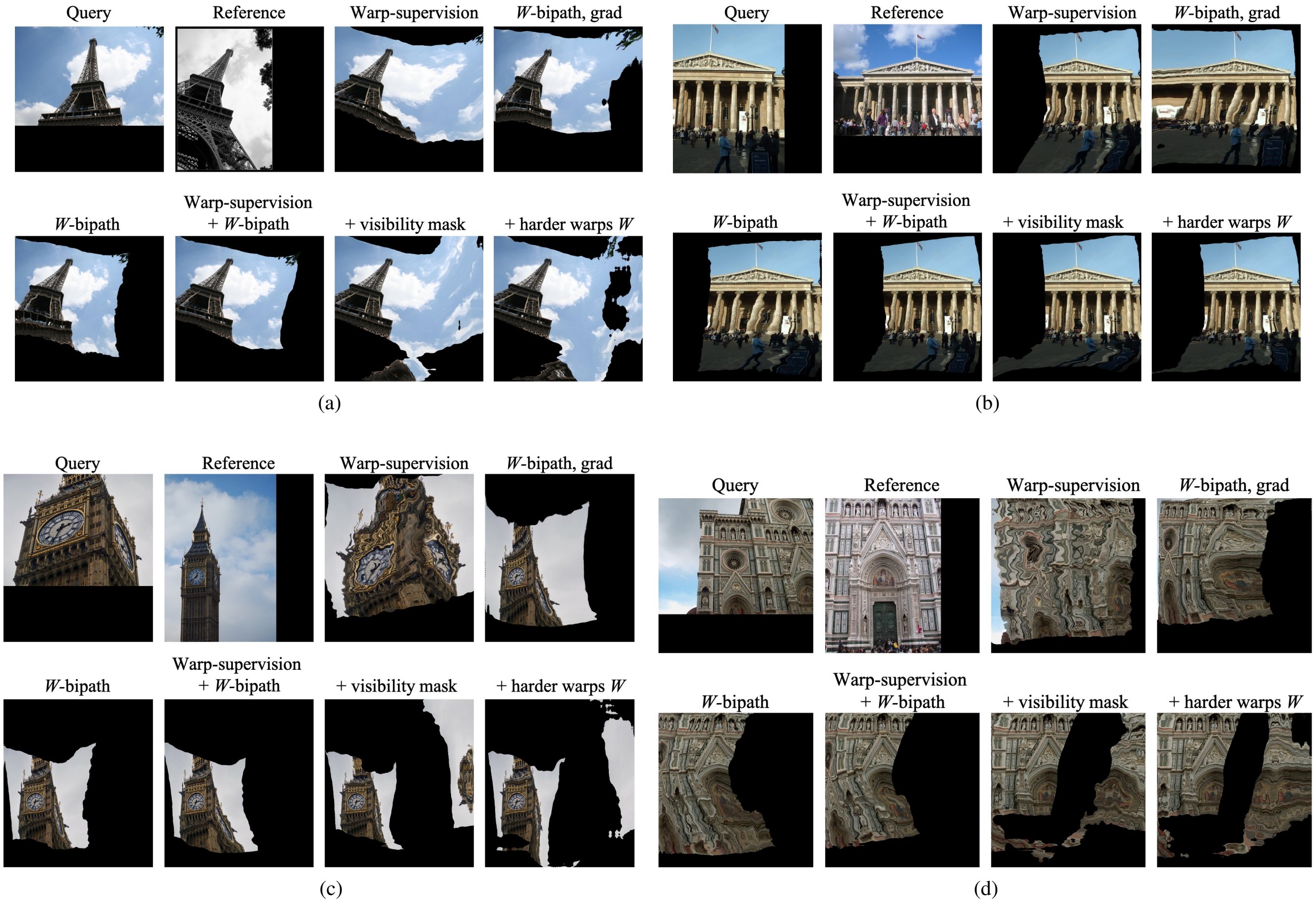
In this section, we extend the method analysis corresponding to Sec. 4.1 of the main paper.

Feature backbone training: We train WarpC-GLU-Net from scratch, including the VGG-16 feature backbone, without initializing it with the pre-trained VGG-16 weights on ImageNet. We compare this version to the one using the pre-trained VGG-16 weights, which are fixed during both training stages. In general, both networks achieve similar results, allowing us to use ImageNet initialization to reduce overall training time. In the first training stage, training the feature backbone from scratch leads to a slightly better accuracy on MegaDepth (PCK-1) as compared to the pre-trained VGG-16 version. However, the resulting model is somewhat less robust to large displacements, evidenced by the lower PCK-10 results. On RobotCar, the network with feature training also obtains slightly worst results, which could be due to the fact that the version using pre-trained weights on ImageNet saw more image diversity. However, training from scratch leads to better performance on HPatches, with a significant $+3\%$ in PCK-5. In the second training stage, the trend is the same but the gap between the two network trainings further reduces.

Additional comparison to alternative unsupervised losses: We here provide additional comparison to other unsupervised losses. We also evaluate more combinations of losses. Results are reported in Tab. 3, which extends Tab. 3 of the main paper. For completeness, we train a version of GLU-Net using standard unsupervised losses in the literature, *i.e*. photometric (SSIM [43] here), forward-backward (eq. 2 of the main paper) and smoothness loss in (III). For the smoothness loss, we use a first order smoothness constraint, following [46]. Adding the smoothness loss leads to an improvement compared to only SSIM (I) or the combination of SSIM and forward-backward (II), particularly on the RobotCar dataset. Nevertheless, it is still significantly lower than our proposed unsupervised approach (VI) for all metrics and datasets, except for PCK-1 on MegaDepth. Moreover, it also obtains lower metrics than the combination of our warp consistency loss with the photometric SSIM loss (VII) on the MegaDepth and HPatches dataset. The RobotCar dataset depicts scenes with little geometric transformations but large appearance variations in the form of seasonal or day-time changes for example (see Fig. 9). As a result, the photometric consistency is strongly violated on those images, while a smoothness loss is beneficial, which explains that the combination of the three classical unsupervised losses (III) leads to slightly better results than the combination of our proposed approach and the photometric SSIM loss (VII). Nevertheless, our proposed warp consistency approach (VI) alone outperforms all other methods on RobotCar.

		MegaDepth			RobotCar			HPatches	
		PCK-1	PCK-5	PCK-10	PCK-1	PCK-5	PCK-10	AEPE	PCK-5
I	SSIM (1)	51.93	69.58	71.58	2.18	31.48	51.65	38.62	62.61
II	SSIM + f-b (1)+(2)	52.59	70.78	72.78	2.12	31.86	52.13	35.79	64.48
III	SSIM + smoothness + f-b	55.00	71.24	73.10	2.42	34.76	55.75	38.13	66.46
IV	Warp-superv. (3)	38.51	60.33	66.57	2.30	33.21	54.19	26.88	78.07
V	Warp-superv. + SSIM (3)+(1)	**56.58**	73.81	75.69	2.27	33.05	53.97	29.50	71.34
VI	**WarpC** (3)+(8)	50.61	**78.61**	**82.94**	**2.51**	**35.92**	**57.44**	**21.00**	**83.24**
VII	**WarpC** + SSIM	55.82	74.89	77.08	2.38	34.56	55.50	26.04	72.44

Table 3. Additional comparison and combination of alternative losses. All equations numbers refer to the main paper.



(a) (b) (c) (d)

Figure 7. Qualitative ablation study of our unsupervised training approach, using GLU-Net as base network. We employ images of the MegaDepth dataset. Note that in the dense estimation settings, the network must predict a match for every pixels in the reference, even in obviously occluded regions. Only correspondences found in overlapping regions are relevant nevertheless.

We also train the combination of warp-supervision loss and SSIM (V). It leads to an improvement compared to SSIM (I) for all dataset and metrics. Nevertheless, on RobotCar and HPatches, the improvement brought by the warp-supervision loss in (V) is significantly lower than the increase brought by combining our proposed warp consistency loss (WarpC) with SSIM in (VII). On MegaDepth, WarpC combined with SSIM (VII) achieves better performance than warp-supervision and SSIM (V) for PCK-5 and PCK-10, for a slightly lower performance on sub-pixel accuracy (PCK-1). This confirms that the warp consistency loss enables to handle the large appearance and geometric changes present between real image pairs, while the warp-supervision loss mostly focuses on getting accuracy to small displacements. Moreover, note that both combinations of SSIM with warp-supervision (V) or WarpC (VII) obtain worse results than our warp consistency loss (WarpC) in (VI) for all datasets and thresholds, except for PCK-1 on MegaDepth.

Qualitative ablation study: Next, we show qualitative results of our ablation study, corresponding to Tab. 2 of the main paper. We warp the queries according to the flows estimated by GLU-Net networks trained with different losses, and present the corresponding qualitative results in Fig. 7. Note that in the dense estimation settings, the network is obligated to predict a match for every pixels in the reference, even in obviously occluded regions. Occluded regions can be filtered out using *e.g.* a forward-backward consistency mask [27], or by letting the network predict a visibility mask as in [36, 28].

On image pairs (c) and (d), the superiority of the W-bipath loss as opposed to the warp-supervision loss is obvious. The warp-supervision loss, solely relying on synthetic training image pairs, is not equipped to handle the large and complex 3D motions present in these example pairs. On the contrary, the W-bipath constraint benefits from direct supervision to improve the predictions between real image pairs during training.

The benefit of not back-propagating the gradients through the estimated flow used in the warping operation ('W-bipath, grad' refers to version with back-propagation) is best illustrated in examples (b) and (c). It leads to a gen-

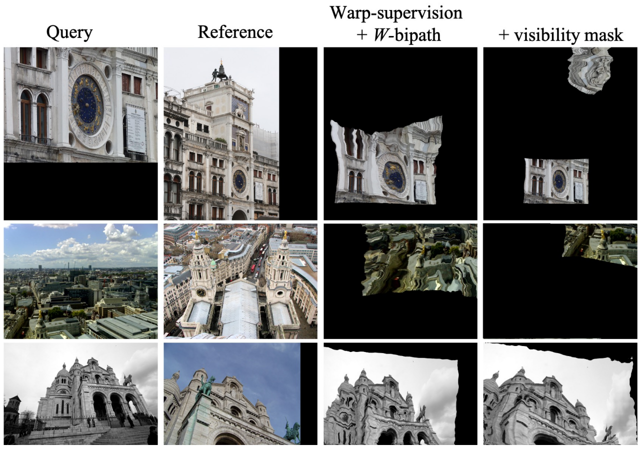


Figure 8. Impact of including the visibility mask (eq. 9 of the main paper) in the W-bipath training loss (eq. 8 of the main paper).

erally more accurate and stable flow predictions. The W-bipath loss version without back-propagation is used as default for the rest of the section, unless otherwise stated.

Combining the warp-supervision with the W-bipath loss drives the network to be more accurate. It is particularly visible on image pairs (a) and (b). On both these examples, combining the warp-supervision with the W-bipath loss results in a more stable and accurate estimated flow.

The impact of extending the W-bipath objective with our visibility mask (eq. 8 of the main paper) is well illustrated on examples (c) and (d). In the former, training with the visibility mask permits to 'clean' the estimated flow and produces a much more accurate prediction. In (d), it allows to get the correct overall geometric transformations and removes most of the shakiness present for previous networks. In general, introducing the visibility mask is a crucial step, which enables the network to better deal with very large geometric variations, such as drastic scale or view-point changes, as visualised in Fig. 8.

Finally, training using harder warps W leads to improved accuracy to small details, as evidenced in example (b), where the columns in the warped query appear straighter.

Method analysis on semantic data: For completeness, we also empirically analyze and decompose our proposed warp consistency loss, when trained and evaluated on semantic data. We follow the training procedure detailed in Sec. E. In Tab. 4, we show results on the TSS [38], PF-Pascal [9] and PF-Willow [8] datasets. From the pretrained SemanticGLU-Net, further finetuning on PF-Pascal using solely the warp-supervision objective improves upon SemanticGLU-Net on all datasets and metrics. Using the W-bipath loss instead leads to slightly better performance on TSS images, but drastic improvement on PF-Willow and PF-Pascal. This is because image pairs of those two datasets are generally much harder and ambiguous than for TSS images, and therefore benefit more from the supervision on real image pairs provided by our warp consistency objective during training. Further combining both objectives (WarpC-SemanticGLU-Net) leads to a substantial improvement on TSS for similar performances than solely the W-bipath loss on PF-Pascal. The combination with the warp-supervision objective also results in an additional boost in accuracy for the lowest threshold $\alpha = 0.05$ on PF-Willow. For reference, we visualize image examples and the performance of WarpC-SemanticGLU-Net on TSS in Fig. 13, 14, 15, and on PF-Pascal in Fig. 16, 17 , 18. Also note that training with the photometric SSIM loss diverges, because it cannot handle the radical appearance and shape variations between multiple instances of the same object class.

	TSS	PF-Pascal		PF-Willow	
	Avg.	α=0.05	α=0.1	α=0.05	α=0.1
SemanticGLU-Net	82.8	46.0	70.6	36.4	63.8
Warp-superv. (eq. 3 of m.p)	85.2	48.8	72.4	39.7	67.5
W-bipath (eq. 8 of m.p)	85.5	**62.9**	**82.0**	47.1	**75.4**
W-bipath + Warp-sup. (**WarpC**)	**87.2**	62.1	81.7	**49.0**	75.1

Table 4. Warp consistency graph analysis on semantic data.

K. Detailed results

In this section, we provide additional results for pose estimation using RANSAC-Flow as base network. We also further evaluate WarpC-GLU-Net on the HPatches dataset [1] and compare it to state-of-the-art methods. We then provide more detailed results on the PF-Pascal semantic dataset [9]. We additionally give evaluation results on the PF-Willow dataset [8] and the SPair-71K dataset [29]. We also show the possible extension of our unsupervised training approach to the optical flow task. Finally, we present extensive qualitative results on multiple geometric and semantic matching datasets.

K.1. Results on pose estimation

Since RANSAC-Flow predicts a matchability mask along with the dense correspondences, we additionally evaluate both jointly for the task of pose estimation. Specifically, we follow the standard set-up of [47] and evaluate on 4 scenes of the YFCC100M dataset [39], each comprising 1000 image pairs.

YFCC100M: The YFCC100M dataset represents touristic landmark images. The ground-truth poses were created by generating 3D reconstructions from a subset of the collections [13]. We use the evaluation procedure introduced in RANSAC-Flow [36]. In particular, the original images and ground-truths are resized to have a minimum dimension of 480.

mAP: For the task of pose estimation, we use mAP as the evaluation metric, following [47]. The absolute rotation error $|R_{err}|$ is computed as the absolute value of the rotation angle needed to align ground-truth rotation matrix R with estimated rotation matrix $\hat{R}$, such as

$$R_{err} = cos^{-1}\frac{Tr(R^{-1}\hat{R}) - 1}{2} , \tag{10}$$

where operator Tr denotes the trace of a matrix. The translation error T_{err} is computed similarly, as the angle to align the ground-truth translation vector T with the estimated translation vector $\hat{T}$.

$$T_{err} = cos^{-1}\frac{T \cdot \hat{T}}{\|T\| \left\|\hat{T}\right\|} , \tag{11}$$

where $\cdot$ denotes the dot-product. The accuracy Acc-κ for a threshold κ is computed as the percentage of image pairs for which the maximum of T_{err} and $|R_{err}|$ is below this threshold. mAP is defined according to original implementation [47], *i.e.* mAP @5° is equal to Acc-5, mAP @10° is the average of Acc-5 and Acc-10, while mAP @20° is the average over Acc-5, Acc-10, Acc-15 and Acc-20.

Results: RANSAC-Flow infers the dense flow field relating an image pair, coupled with a predicted matchability mask, both trained unsupervised. Pose estimation on YFCC100M evaluates the performance of the predicted flow and mask jointly. Indeed, for pose estimation, confidence or mask prediction is *crucial* in order to select the accurate matches from the dense flow output and further use them to compute the pose. Results on YFCC100M are presented in Tab. 5. In the original RANSAC-Flow work, results are only reported when using an additional semantic segmentation network (SegNet) to better filter unreliable correspondences, in *e.g.* sky. However, using a segmentation networks makes the overall method supervised. We therefore present results without any segmentation network, purely relying on RANSAC-Flow outputs. Without this additional segmentation, the performance of RANSAC-Flow is drastically reduced. In contrast, WarpC-RANSAC-Flow, trained with our unsupervised approach (Sec. D), can directly estimate highly robust and generalizable matchability masks. The predicted masks of RANSAC-Flow and our approach WarpC-RANSAC-Flow are visually compared in Fig. 12, in yellow and red respectively. Crucially, training with a photometric objective does not permit to identify unreliable matching regions such as the sky, that fit the brightness constancy assumption of the photometric objective. These regions, when included for pose estimation computation, will drastically reduce the performance of the network. On the other hand, our proposed unsupervised objective enables to identify accurate matching regions and to filter out outliers or unreliable regions, leading to drastically better results.

	mAP @5°	mAP @10°	mAP @20°
Superpoint [6]	30.50	50.83	67.85
SIFT [25]	46.83	68.03	80.58
D2D [44]	55.58	66.79	-
RANSAC-Flow [36] (SegNet)	**63.48**	**72.93**	**81.59**
WarpC-RANSAC-Flow (SegNet)	62.90	72.48	81.56
RANSAC-Flow [36]	30.93	38.20	46.88
WarpC-RANSAC-Flow	**61.85**	**71.24**	**79.86**

Table 5. Two-view geometry estimation on YFCC100M [39]. Including an additional segmentation network (SegNet) makes the overall training supervised.

	AEPE ↓	PCK-1 (%) ↑	PCK-5 (%) ↑
DGC-Net [28]	33.26	12.00	58.06
GLU-Net [42]	25.05	39.55	78.54
GLU-Net-GOCor [41]	**20.16**	**41.55**	81.43
GLU-Net*	25.04	39.37	78.60
WarpC-GLU-Net	21.00	41.00	**83.24**

Table 6. HPatches homography dataset [1].

K.2. Results on HPatches

We here present results of WarpC-GLU-Net against baseline GLU-Net* and state-of-the-art methods on the geometric matching homography dataset HPatches [1] in Tab. 6. Our approach WarpC-GLU-Net scores second in AEPE and PCK-1, after the recent GLU-Net-GOCor, which uses the GOCor module [41] in replacement to the feature correlation layer. We could also use our unsupervised learning approach to train GLU-Net-GOCor and further benefit from the improvement brought by GOCor. WarpC-GLU-Net is nevertheless significantly better than GLU-Net and baseline GLU-Net*, which both use a warp-supervision training loss.

K.3. Additional results on semantic matching

In this section, we present additional results on semantic data. While our proposed warp supervision objective offers a general training approach, applicable to multiple tasks such as semantic as well as geometric matching, we here compare it to methods specifically and exclusively designed for semantic data.

PF-Pascal and PF-Willow: In Tab. 7, we extend Tab. 5 of the main paper, by showing PCK results for the threshold $\alpha = 0.15$ on the PF-Pascal dataset. We additionally report evaluation results on the PF-Willow dataset [8].

On the PF-Pascal data, our proposed approach WarpC-SemanticGLU-Net ranks first for the lowest threshold $\alpha = 0.05$. For the second and third thresholds, it is marginally behind the current state-of-the-art DCC-Net [15] and DHPF [30] (0.6 % for $\alpha = 0.1$ and 1.4 % for $\alpha = 0.15$). Note however, that these networks use a much stronger and deeper pre-trained feature backbone, namely ResNet-101, while we employ a VGG-16 backbone. On PF-Willow, WarpC-SemanticGLU-Net ranks second for all thresholds, shortly after the recent DHPF [30]. Nevertheless, it obtains

Supervision	Methods	Features	TSS PCK @ α_{img} FG3DCar	 JODS	 Pascal	 Avg.	PF-Pascal PCK @ α_{img} $\alpha = 0.05$	 $\alpha = 0.10$	 $\alpha = 0.15$	PF-Willow PCK @ α_{bbox} $\alpha = 0.05$	 $\alpha = 0.10$	 $\alpha = 0.15$
segmentation mask	SF-Net [23]	ResNet-101	-	-	-	-	53.6	*81.9*	*90.6*	46.3	74.0	84.2
warp-supervision (synthetic pairs)	CNNGeo(S) [31]	ResNet-101	90.1	76.4	56.3	74.4	41.0	69.5	80.4	36.9	69.2	77.8
	GLU-Net [42]	VGG-16	93.2	73.3	71.1	79.2	42.2	69.1	83.1	30.4	57.7	72.9
	GLU-Net-GOCor [41]	VGG-16	94.6	77.9	77.7	83.4	36.6	56.8	-	-	-	-
	A2Net [35]	ResNet-101	-	-	-	-	42.8	70.8	83.3	36.3	68.8	84.4
image-level labels	WeakAlign [32]	ResNet-101	90.3	76.4	56.5	74.4	49.0	74.8	84.0	38.2	71.2	85.8
	RTNs [18]	ResNet-101	90.3	76.4	56.5	74.4	55.2	75.9	85.2	41.3	71.9	86.2
	PARN [17]	ResNet-101	89.5	75.9	*71.2*	78.8	-	-	-	-	-	-
	NC-Net [33]	ResNet-101	94.5	81.4	57.1	77.7	54.3	78.9	86.0	33.8	67.0	83.7
	DCCNet [15]	ResNet-101	93.5	*82.6*	57.6	77.9	55.6	**82.3**	90.5	43.6	73.8	86.5
	DHPF [30]	ResNet-101	-	-	-	-	56.1	82.1	**91.1**	**50.2**	**80.2**	**91.1**
	SAM-Net [19]	VGG-19	*96.1*	82.2	67.2	81.8	*60.1*	80.2	86.9	-	-	-
warp-supervision image-level labels	Semantic-GLU-Net [42]	VGG-16	94.4	75.5	78.3	*82.8*	46.0	70.6	83.3	36.4	63.8	78.4
	WarpC-SemanticGLU-Net (Ours)	VGG-16	**97.1**	**84.7**	**79.7**	**87.2**	**62.1**	81.7	89.7	*49.0*	*75.1*	*86.9*

Table 7. PCK [%] obtained by different state-of-the-art unsupervised methods on the TSS [38], PF-Pascal [9] and PF-Willow [8] datasets for the task of semantic matching. Results from [31, 35, 32, 18, 33, 15] are from [30]. Best results are highlighted in red, while second best are in blue. We compare our approach WarpC-SemanticGLU-Net to methods specifically and exclusively designed for semantic data, trained unsupervised. On the contrary, our proposed warp consistency loss (Sec. 3.5 of the main paper) offers a general formulation, applicable to multiple tasks, including geometric and semantic matching. In the last section of the table, we highlight the improvement brought by our unsupervised finetuning.

substantially better results than all other methods excluding DHPF, especially for the lowest threshold $\alpha = 0.05$ with a notable improvement of $+2.7\%$ compared to next best approach. We also note that DHPF predicts a cost volume as final output whereas we infer the dense flow field relating an image pair. On the TSS dataset, where dense ground-truth flows are available, our approach outperforms all previous approaches by a large margin.

Importantly, as highlighted in the last section of the table, our unsupervised finetuning leads to an impressive improvement compared to original SemanticGLU-Net: $+16.1\%$, $+11.1\%$ and $+6.4\%$ for thresholds $\alpha = \{0.05, 0.1, 0.15\}$ on the PF-Pascal dataset, and $+12.6\%$, $+11.3\%$ and $+8.5\%$ on the PF-Willow dataset for the same thresholds. As a result, while we chose a relatively weak baseline on these datasets, our unsupervised finetuning makes the resulting model very competitive, achieving first or second best metrics overall on four PCK thresholds out of six. Moreover, any other baseline could be used instead, benefiting from our unsupervised warp consistency finetuning.

SPair-71k: We also evaluated our unsupervised approach on the SPair-71k benchmark [29]. It includes 70,958 image pairs of 18 object categories from PASCAL 3D+ [45] and PASCAL VOC 2012 [7], providing 12,234 pairs for testing. This benchmark is more challenging than other datasets [9, 8, 38] for semantic correspondence evaluation, as it covers significantly large variations of viewpoint, scale, truncation and occlusion. For the evaluation metric, we used the PCK with respect to the object bounding box and $\alpha = 0.1$. We finetuned our WarpC-Semantic-GLU-Net with our unsupervised warp consistency strategy on the training set of SPair-71K. We compare to other unsupervised approaches trained or finetuned on the same data in Tab. 8. For comparison, we also further finetuned the original Semantic-GLU-Net [42] on SPair-71k with the warp-supervision objective. Our WarpCSemantic-GLU-Net is competitive with other unsupervised approaches. As before, while the Semantic-GLU-Net baseline architecture appears quite weak on the SPair-71K images, note the significant improvement (+9.2 %) brought by our unsupervised warp consistency finetuning, as opposed to simple warp-supervision. We believe that training another stronger baseline would further improve results.

K.4. Extension to optical flow data

Here, we show the generalization capabilities of our unsupervised approach for the optical flow task. We report results on KITTI and MPI Sintel in Tab. 9 by comparing our approach (WarpC-GLU-Net) with the baseline (GLU-Net*) trained using only warp-supervision. We evaluate according to the standard metrics, namely AEPE and Fl for KITTI and AEPE and PCKs for Sintel. Note that we use the *same weights* as in the paper, which are trained for the dense geometric matching task on the MegaDepth training set, and therefore not well suited for the optical flow setting. Still, WarpC-GLU-Net obtains significantly better results than GLU-Net*, showing the benefit of our unsupervised

Table 8. PCK for $\alpha = 0.1$ with respect to object bounding box on SPair-71k [29]. We compare to unsupervised approaches trained or finetuned on the training set of Spair-71k [29].

Methods	Feature backbone	PCK @ α_{bbox} [%]
CNNGeo [31]	ResNet-101	20.6
WeakAlign [32]	ResNet-101	20.9
A2Net [35]	ResNet-101	22.3
NC-Net [33]	ResNet-101	20.1
DHPF [30]	ResNet-101	**27.7**
SF-Net [23]	ResNet-101	26.5
SemanticGLU-Net [42] (warp-sup.)	VGG-16	14.3
WarpCSemanticGLU-Net	VGG-16	23.5

Table 9. Results on the training splits of KITTI and Sintel.

	KITTI-2012		KITTI-2015		Sintel Clean		Sintel Final	
	AEPE	F1 (%)	AEPE	Fl (%)	AEPE	PCK-1 (%)	AEPE	PCK-1 (%)
GLU-Net*	3.37	17.38	10.90	36.06	5.74	69.47	6.60	61.33
WarpC-GLU-Net	**3.09**	**16.32**	**9.35**	**33.65**	**5.23**	**70.86**	**6.30**	**62.83**

training, even when trained for a different domain. We believe that unsupervised training on road-scene videos, such as KITTI raw, would further improve the results.

K.5. Qualitative results

Finally, we provide extensive qualitative visual examples of the performance of our WarpC models. We first qualitatively compare baseline GLU-Net* and our approach WarpC-GLU-Net on images of MegaDepth and RobotCar in Fig. 10, 11 and 9 respectively. WarpC-GLU-Net is significantly more accurate than GLU-Net*. It can also handle very drastic scale and view-point changes, where GLU-Net* often completely fails. This is thanks to our W-bipath objective, which provides supervision on the network predictions between the real image pairs, as opposed to the warp-supervision objective. Also note that in the dense estimation settings, the network must predict a match for every pixels in the reference, even in obviously occluded regions. Only correspondences found in overlapping regions are relevant nevertheless. Moreover, occluded regions can be filtered out using *e.g.* a forward-backward consistency mask [27], or by letting the network predict a visibility mask as in [36, 28]. This is particularly important for MegaDepth images, in which some image pairs have overlapping ratios below 10%. On RobotCar images in Fig. 9, our approach WarpC-GLU-Net better handles large appearance variations, such as seasonal or day-night changes.

We further show qualitative results of RANSAC-Flow and WarpC-RANSAC-Flow on YFCC100M images in Fig. 12. Contrary to RANSAC-Flow, the masks predicted by WarpC-RANSAC-Flow correctly filter out unreliable, homogeneous or ambiguous regions, such as the sky or field.

We then show the performance of WarpC-SemanticGLU-Net compared to SemanticGLU-Net on images of TSS in Fig. 13, 14 and 15. Our unsupervised finetuning brings visible robustness to the large appearance changes and shape variations inherent to the semantic matching task. Finally, we also qualitatively compare both networks on images of the PF-Pascal dataset in Fig. 16, 17 and 18. The PF-Pascal dataset shows more diverse object categories than TSS images. WarpC-SemanticGLU-Net manages to accurately align challenging image pairs, such as the chair examples which are particularly cluttered.

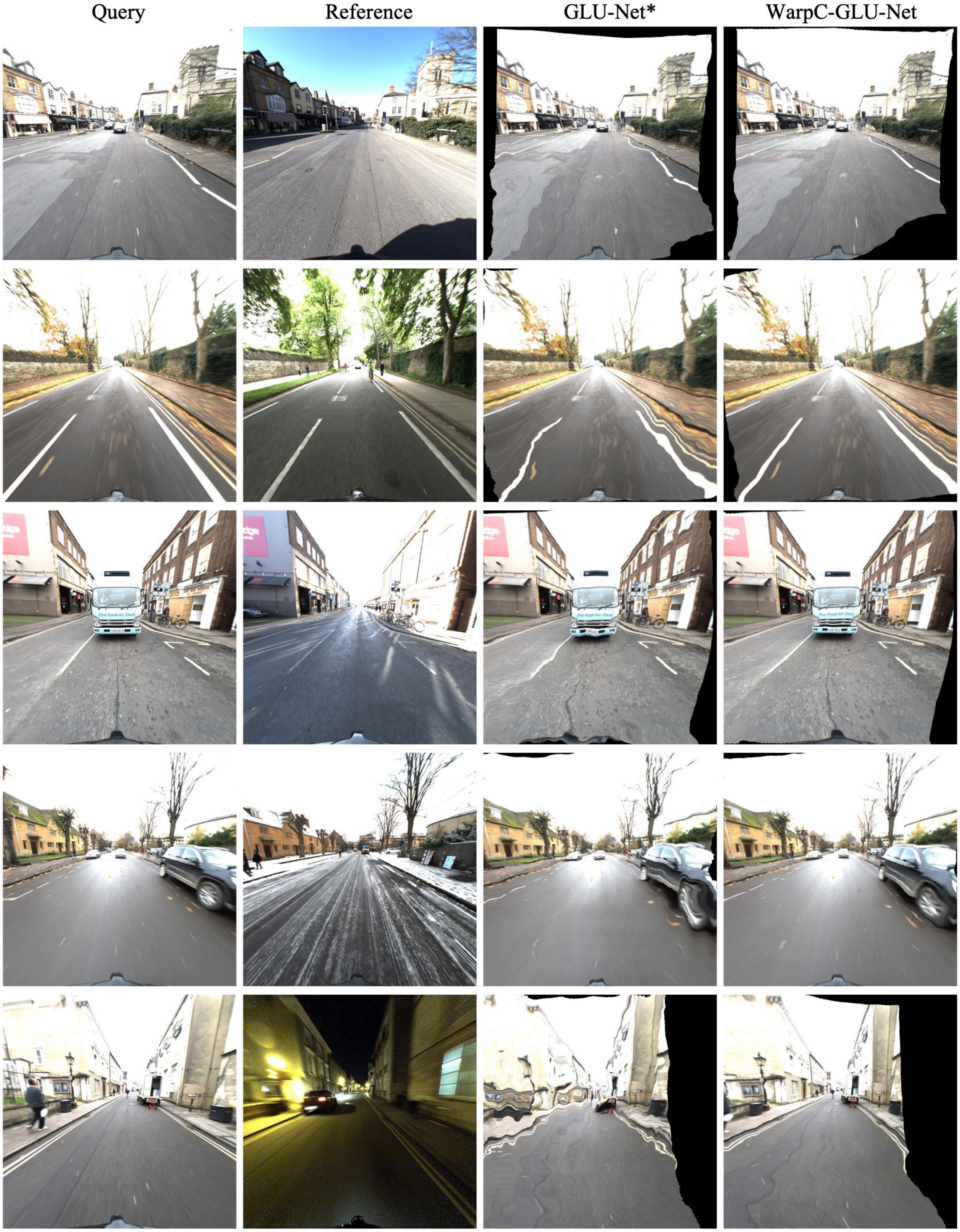


Figure 9. Visual comparison on image pairs of the RobotCar dataset [22], of GLU-Net* and WarpC-GLU-Net. We visualize the query images warped according to the flow fields estimated by both networks. The warped queries should align with the reference images.

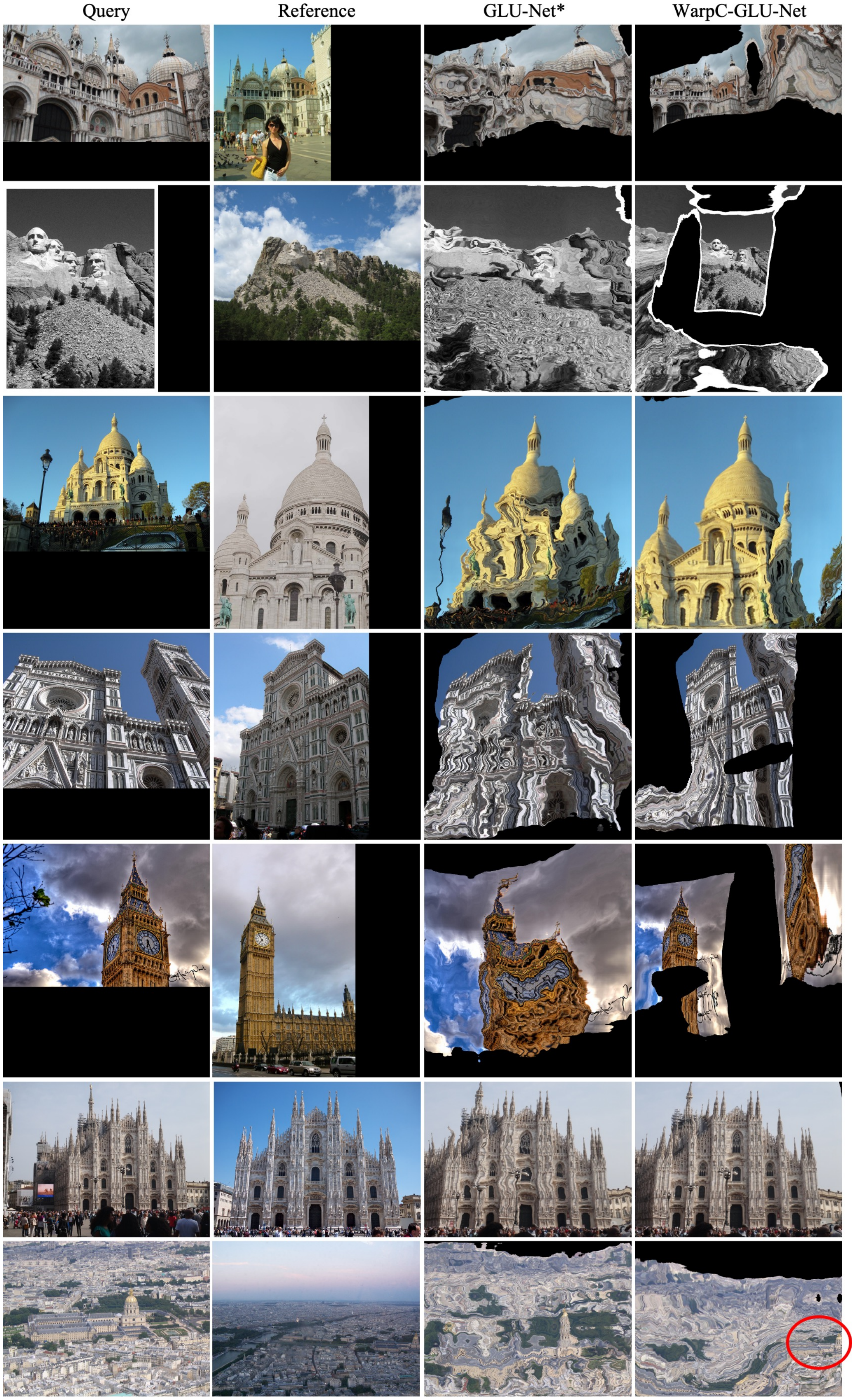


Figure 10. Visual comparison on image pairs of the MegaDepth dataset [24], of GLU-Net* and WarpC-GLU-Net. We visualize the query images warped according to the flow fields estimated by both networks. The warped queries should align with the reference images in overlapping regions. Note that in the dense estimation settings, the network is obligated to predict a match for every pixels in the reference, even in obviously occluded regions. Only correspondences found in overlapping regions are relevant nevertheless. In the last row, we highlight the overlapping region in red, because it is particularly difficult to see.

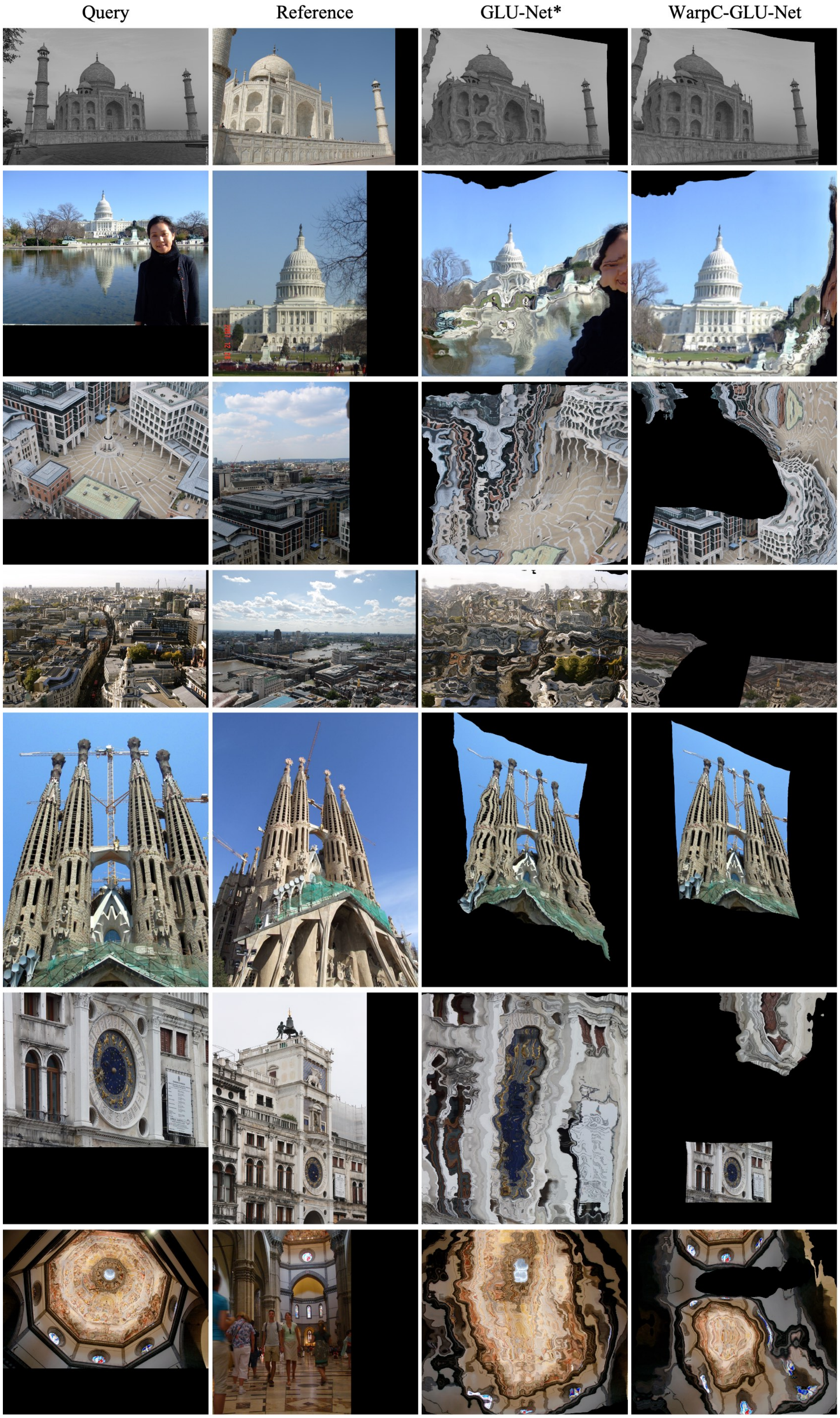


Figure 11. Visual comparison on image pairs of the MegaDepth dataset [24], of GLU-Net* and WarpC-GLU-Net. We visualize the query images warped according to the flow fields estimated by both networks. The warped queries should align with the reference images in overlapping regions. Note that in the dense estimation settings, the network is obligated to predict a match for every pixels in the reference, even in obviously occluded regions. Only correspondences found in overlapping regions are relevant nevertheless.

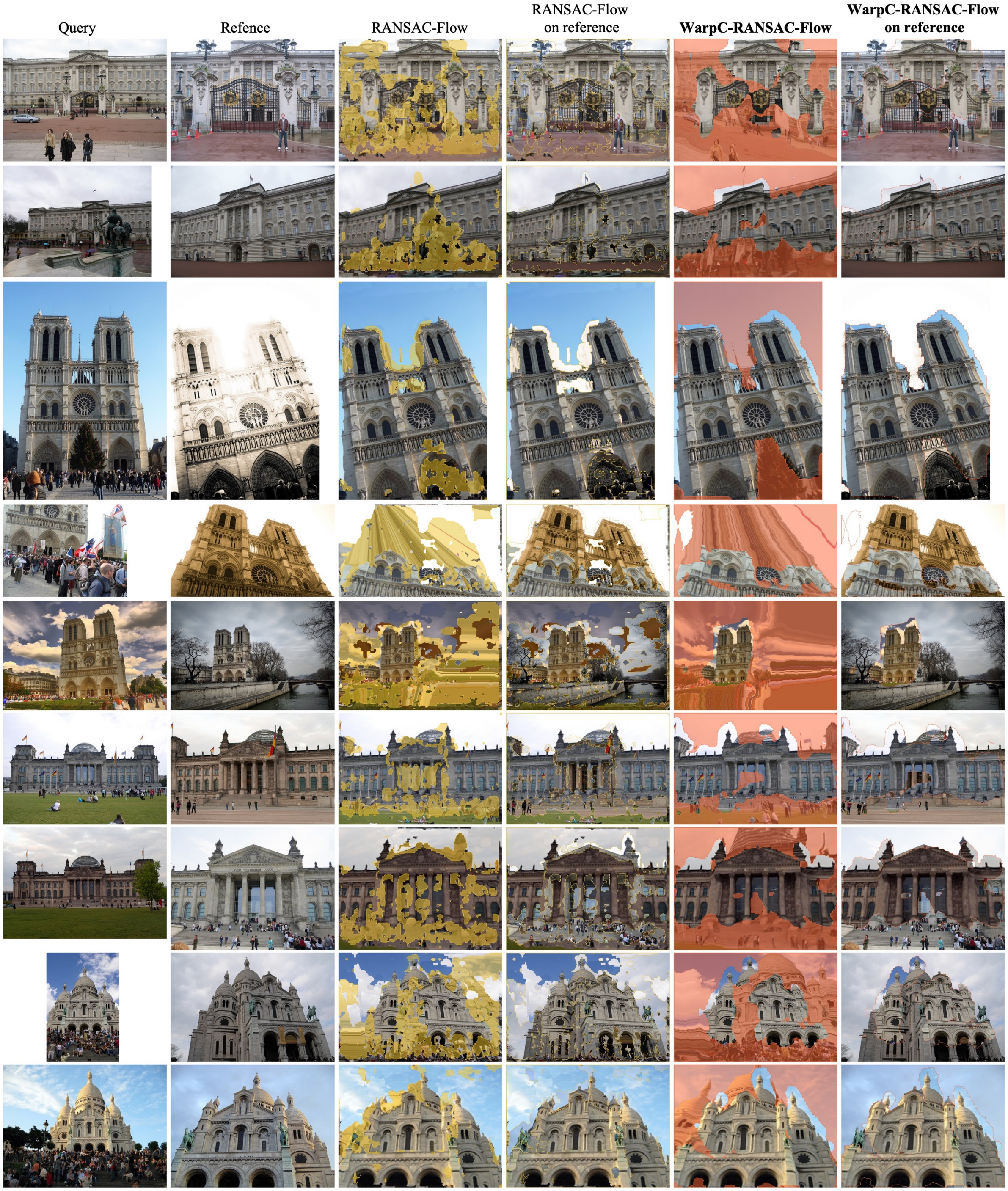


Figure 12. Visual comparison of RANSAC-Flow and our approach WarpC-RANSAC-Flow on image pairs of the YFCC100M dataset [39]. In the 3^{rd} and 5^{th} columns, we visualize the query images warped according to the flow fields estimated by the RANSAC-Flow and WarpC-RANSAC-Flow respectively. Both networks also predict a confidence map, according to which the regions represented in respectively yellow and red, are unreliable or inaccurate matching regions. In the 4^{th} and last columns, we overlay the reference image with the warped query, in the identified accurate matching regions.

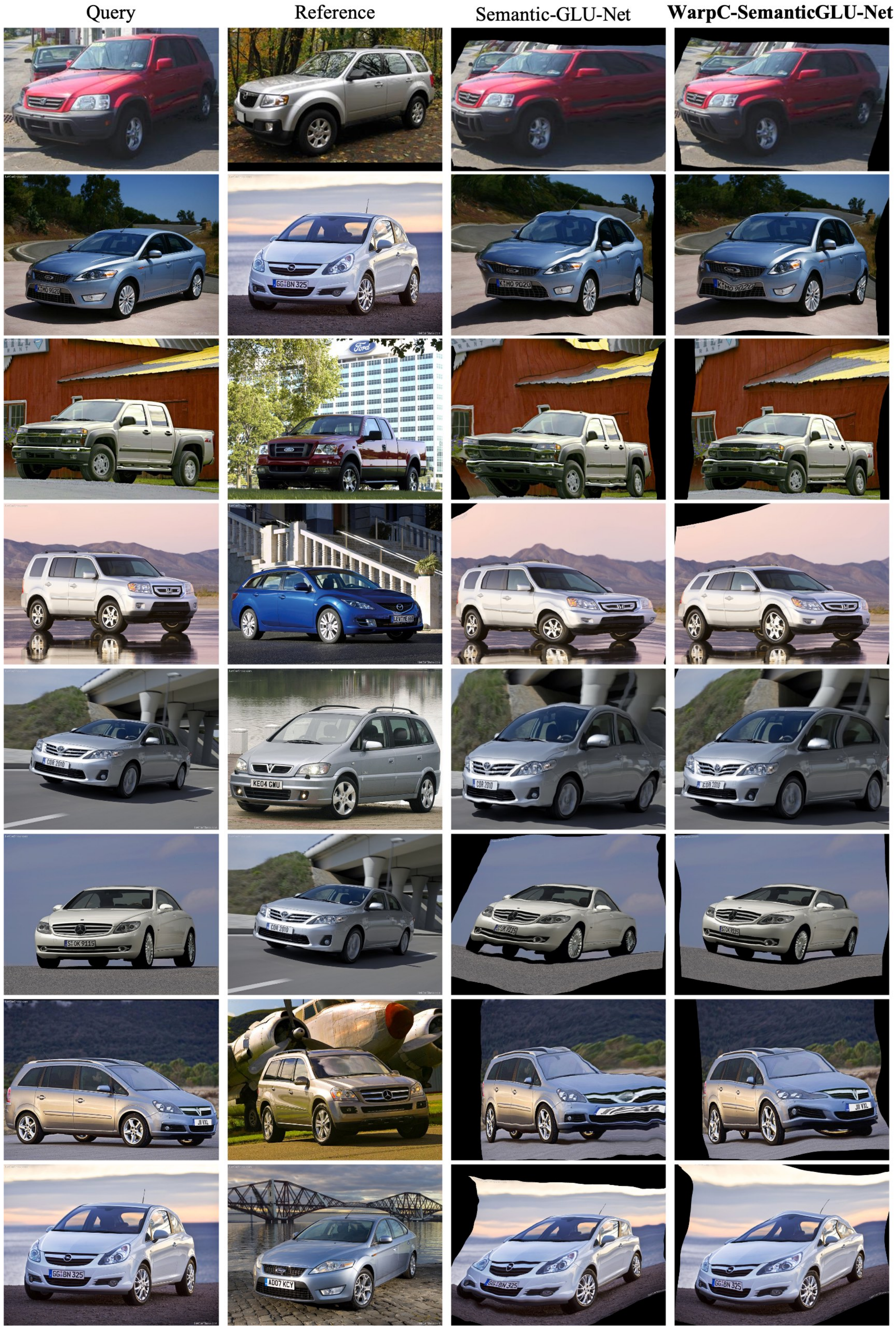


Figure 13. Visual comparison on image pairs of the TSS dataset [39] FD3Car, of original Semantic-GLU-Net [42], trained with the warp-supervision loss on a collection of street-view images, and our approach WarpC-Semantic-GLU-Net, where the network is further finetuned on semantic data using our proposed unsupervised loss. We visualize the query images warped according to the flow fields estimated by Semantic-GLU-Net and WarpC-Semantic-GLU-Net respectively. The warped queries should align with the reference images.

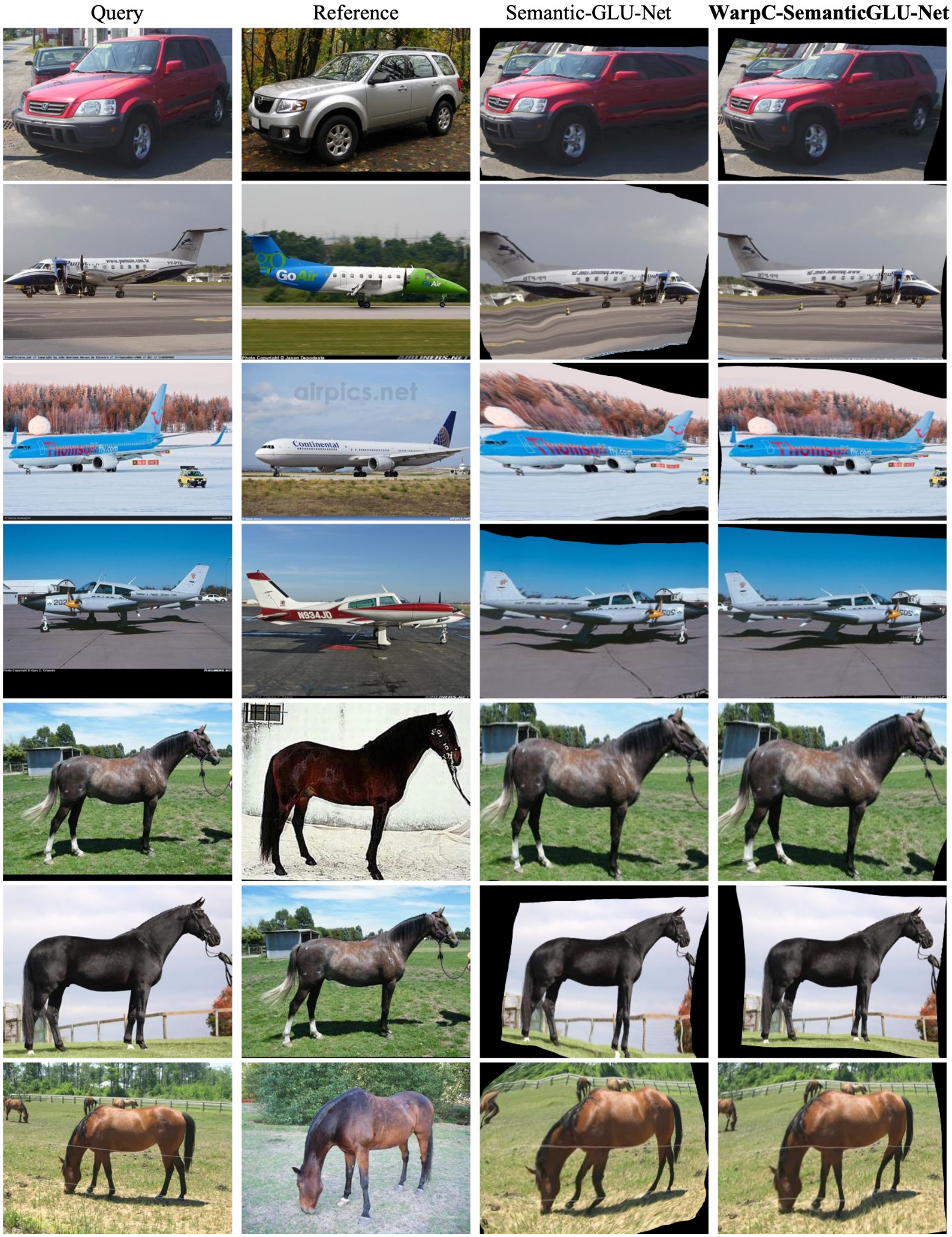


Figure 14. Visual comparison on image pairs of the TSS dataset [38] JODS, of original Semantic-GLU-Net [42], trained with the warp-supervision loss on a collection of street-view images, and our approach WarpC-Semantic-GLU-Net, where the network is further finetuned on semantic data using our proposed unsupervised loss. We visualize the query images warped according to the flow fields estimated by Semantic-GLU-Net and WarpC-Semantic-GLU-Net respectively. The warped queries should align with the reference images.

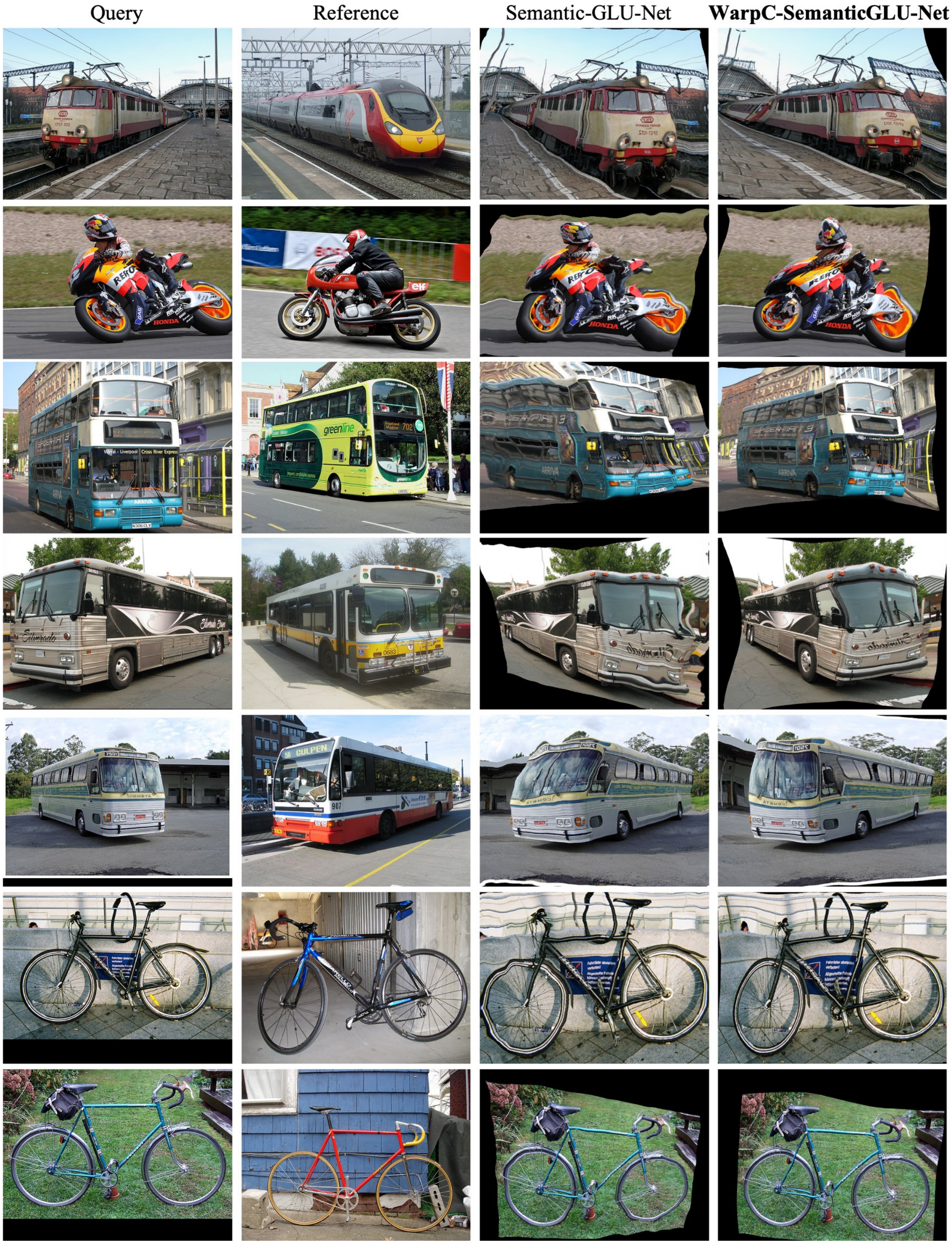


Figure 15. Visual comparison on image pairs of the TSS dataset [38] PASCAL, of original Semantic-GLU-Net [42], trained with the warp-supervision loss on a collection of street-view images, and our approach WarpC-Semantic-GLU-Net, where the network is further finetuned on semantic data using our proposed unsupervised loss. We visualize the query images warped according to the flow fields estimated by Semantic-GLU-Net and WarpC-Semantic-GLU-Net respectively. The warped queries should align with the reference images.

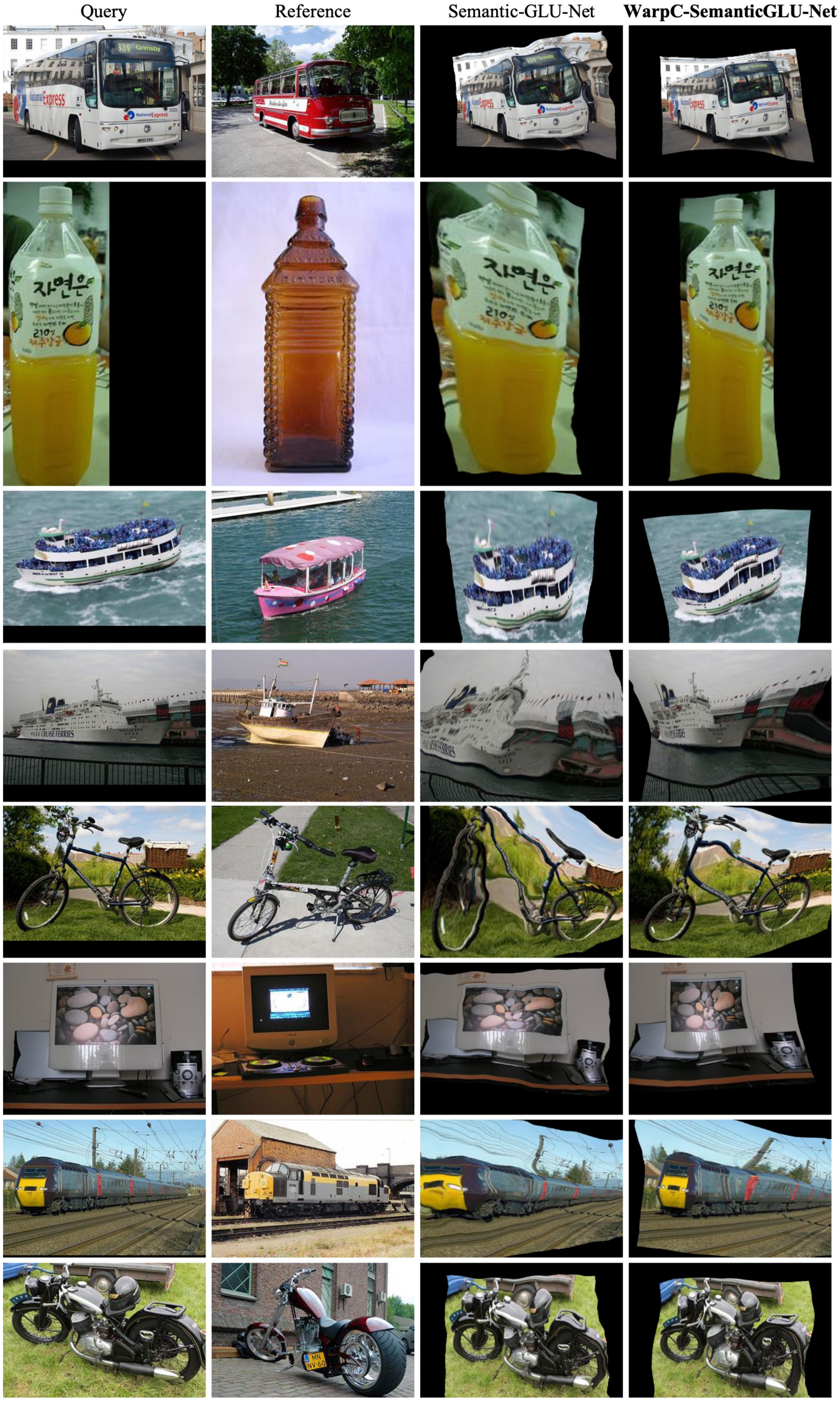


Figure 16. Visual comparison on image pairs of the PF-Pascal dataset [9], of original Semantic-GLU-Net [42], trained with the warp-supervision loss on a collection of street-view images, and our approach WarpC-Semantic-GLU-Net, where the network is further finetuned on semantic data using our proposed unsupervised loss. We visualize the query images warped according to the flow fields estimated by Semantic-GLU-Net and WarpC-Semantic-GLU-Net respectively. The warped queries should align with the reference images.

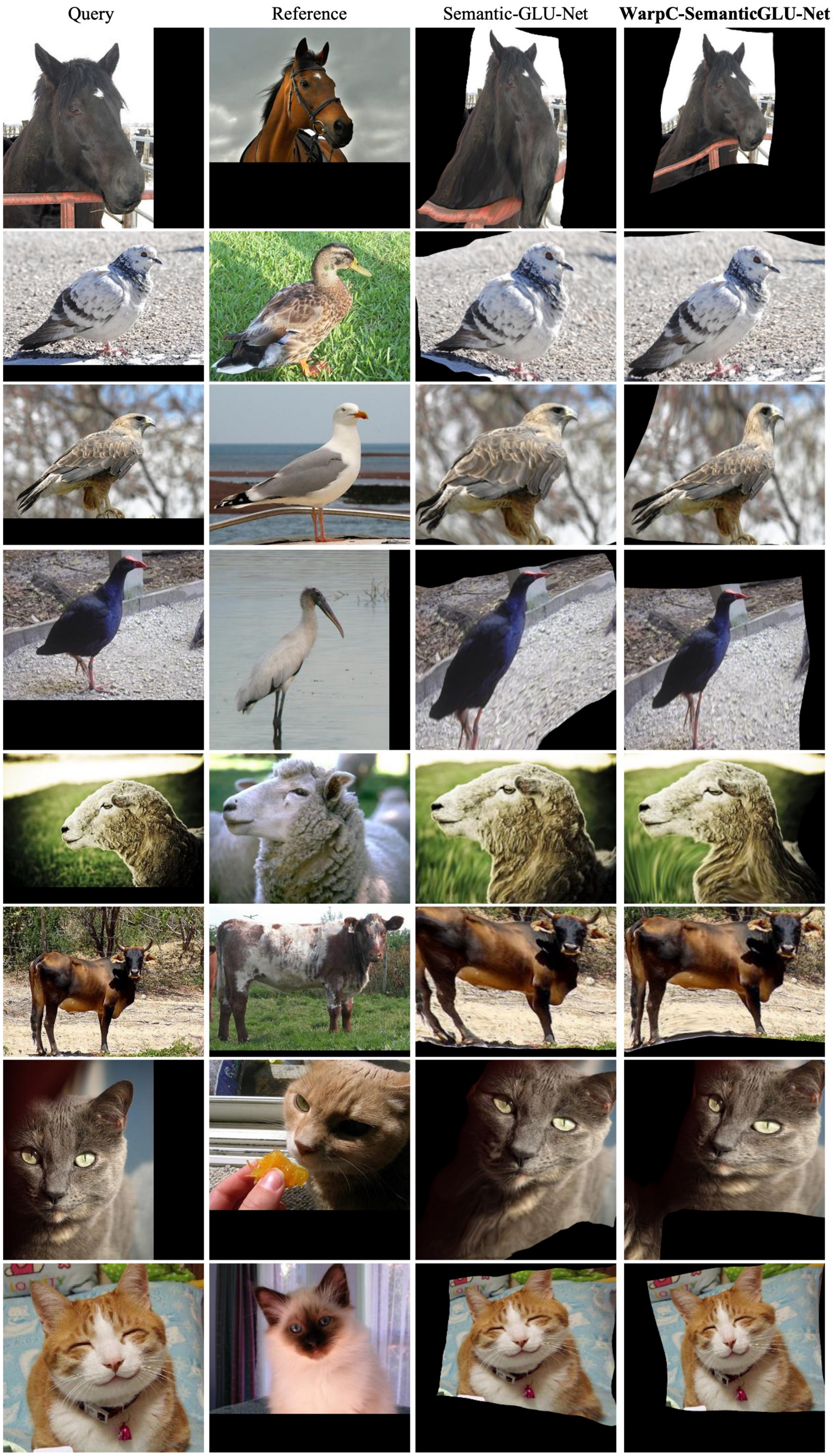


Figure 17. Visual comparison on image pairs of the PF-Pascal dataset [9], of original Semantic-GLU-Net [42] and our approach WarpC-Semantic-GLU-Net.

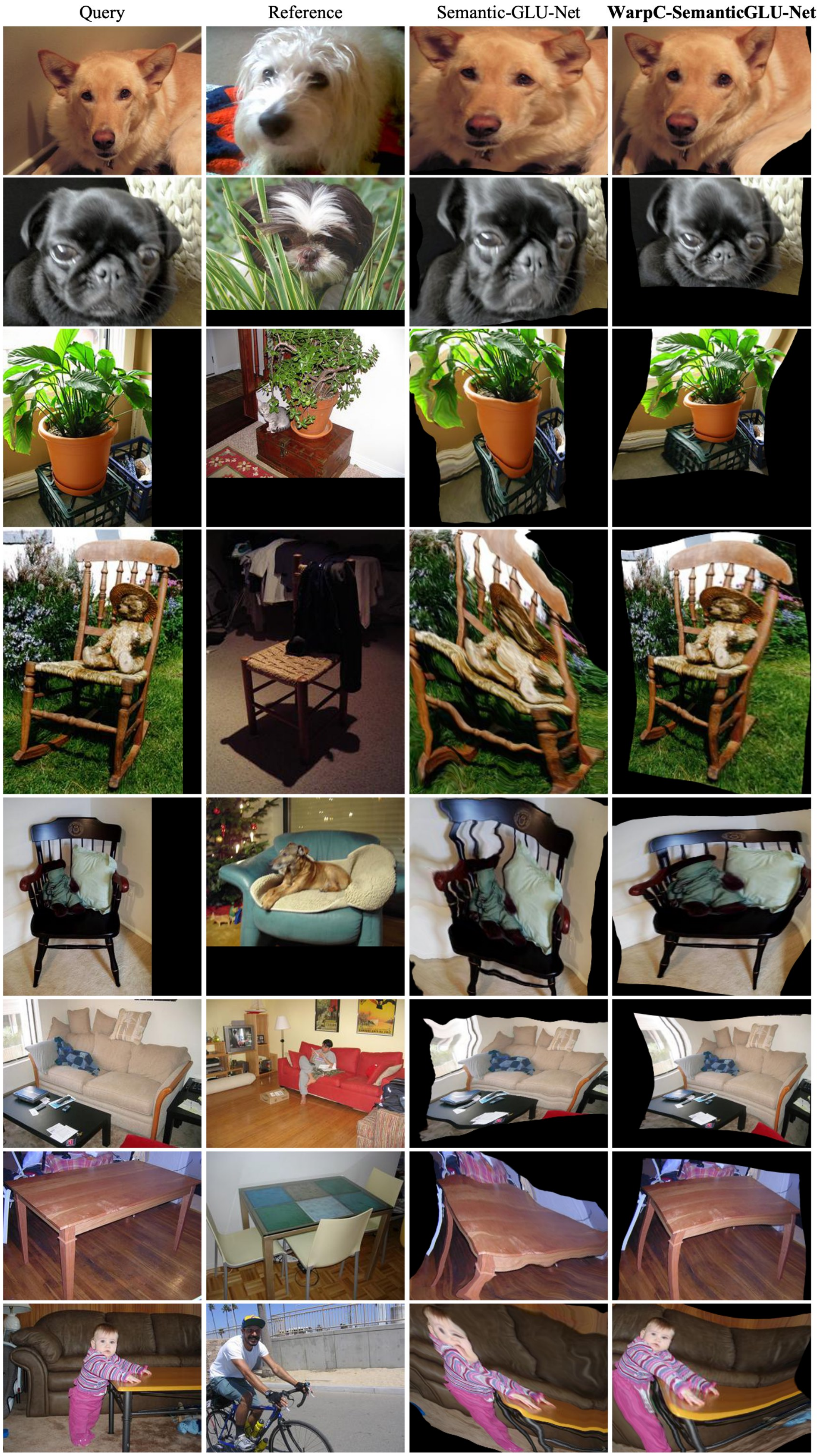


Figure 18. Visual comparison on image pairs of the PF-Pascal dataset [9], of original Semantic-GLU-Net [42] and our approach WarpC-Semantic-GLU-Net.

References

[1] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 3852–3861, 2017. 1, 11, 14, 15

[2] Lubomir D. Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, pages 1365–1372. IEEE Computer Society, 2009. 12

[3] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020. 5

[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014. 5

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[6] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 224–236, 2018. 15

[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 16

[8] Bumsub Ham, Minsu Cho, Cordelia Schmid, and Jean Ponce. Proposal flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1, 12, 14, 15, 16

[9] Bumsub Ham, Minsu Cho, Cordelia Schmid, and Jean Ponce. Proposal flow: Semantic correspondences from object proposals. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(7):1711–1725, 2018. 8, 9, 12, 14, 16, 25, 26, 27

[10] Kai Han, Rafael S. Rezende, Bumsub Ham, Kwan-Yee K. Wong, Minsu Cho, Cordelia Schmid, and Jean Ponce. Scnet: Learning semantic correspondence. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1849–1858, 2017. 8

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9726–9735, 2020. 6, 7

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 7

[13] Jared Heinly, Johannes Lutz Schönberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the World* in Six Days *(As Captured by the Yahoo 100 Million Image Dataset). In *Computer Vision and Pattern Recognition (CVPR)*, 2015. 14

[14] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society, 2017. 5

[15] Shuaiyi Huang, Qiuyue Wang, Songyang Zhang, Shipeng Yan, and Xuming He. Dynamic context correspondence network for semantic alignment. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 2010–2019. IEEE, 2019. 15, 16

[16] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Dslr-quality photos on mobile devices with deep convolutional networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 3297–3305, 2017. 5

[17] Sangryul Jeon, Seungryong Kim, Dongbo Min, and Kwanghoon Sohn. PARN: pyramidal affine regression networks for dense semantic correspondence. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VI*, pages 355–371, 2018. 8, 16

[18] Seungryong Kim, Stephen Lin, Sangryul Jeon, Dongbo Min, and Kwanghoon Sohn. Recurrent transformer networks for semantic correspondence. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 6129–6139, 2018. 16

[19] Seungryong Kim, Dongbo Min, Somi Jeong, Sunok Kim, Sangryul Jeon, and Kwanghoon Sohn. Semantic attribute matching networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 12339–12348, 2019. 16

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6, 8

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114, 2012. 6

[22] Måns Larsson, Erik Stenborg, Lars Hammarstrand, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. A cross-season correspondence dataset for robust semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 9532–9542, 2019. 18

[23] Junghyup Lee, Dohyung Kim, Jean Ponce, and Bumsub Ham. Sfnet: Learning object-aware semantic correspon-

dence. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2278–2287, 2019. 16

[24] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2041–2050, 2018. 7, 11, 19, 20

[25] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 15

[26] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017. 12

[27] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised learning of optical flow with a bidirectional census loss. In *AAAI*, New Orleans, Louisiana, Feb. 2018. 8, 13, 17

[28] Iaroslav Melekhov, Aleksei Tiulpin, Torsten Sattler, Marc Pollefeys, Esa Rahtu, and Juho Kannala. DGC-Net: Dense geometric correspondence network. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019. 5, 11, 13, 15, 17

[29] Juhong Min, Jongmin Lee, Jean Ponce, and Minsu Cho. Spair-71k: A large-scale benchmark for semantic correspondence. *CoRR*, abs/1908.10543, 2019. 1, 14, 16

[30] Juhong Min, Jongmin Lee, Jean Ponce, and Minsu Cho. Learning to compose hypercolumns for visual correspondence. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XV*, pages 346–363, 2020. 15, 16

[31] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. Convolutional neural network architecture for geometric matching. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 39–48, 2017. 4, 16

[32] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic. End-to-end weakly-supervised semantic alignment. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6917–6925, 2018. 16

[33] Ignacio Rocco, Mircea Cimpoi, Relja Arandjelovic, Akihiko Torii, Tomás Pajdla, and Josef Sivic. Neighbourhood consensus networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 1658–1669, 2018. 8, 16

[34] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4104–4113, 2016. 5

[35] Paul Hongsuck Seo, Jongmin Lee, Deunsol Jung, Bohyung Han, and Minsu Cho. Attentive semantic alignment with offset-aware correlation kernels. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV*, pages 367–383, 2018. 16

[36] Xi Shen, François Darmon, Alexei A Efros, and Mathieu Aubry. Ransac-flow: generic two-stage image alignment. In *16th European Conference on Computer Vision*, 2020. 7, 9, 12, 13, 14, 15, 17

[37] Patrice Y. Simard, Dave Steinkraus, and John C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition - Volume 2*, ICDAR '03, page 958, USA, 2003. IEEE Computer Society. 4

[38] Tatsunori Taniai, Sudipta N. Sinha, and Yoichi Sato. Joint recovery of dense correspondence and cosegmentation in two images. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4246–4255, 2016. 12, 14, 16, 23, 24

[39] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: the new data in multimedia research. *Commun. ACM*, 59(2):64–73, 2016. 1, 14, 15, 21, 22

[40] Prune Truong. GLU-Net: Github project page. `https://github.com/PruneTruong/GLU-Net`, 2020. 5

[41] Prune Truong, Martin Danelljan, Luc Van Gool, and Radu Timofte. GOCor: Bringing globally optimized correspondence volumes into your neural network. In *Annual Conference on Neural Information Processing Systems, NeurIPS*, 2020. 15, 16

[42] Prune Truong, Martin Danelljan, and Radu Timofte. GLU-Net: Global-local universal network for dense flow and correspondences. In *2020 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2020*, 2020. 6, 8, 15, 16, 22, 23, 24, 25, 26, 27

[43] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004. 7, 12

[44] Olivia Wiles, Sébastien Ehrhardt, and Andrew Zisserman. D2D: learning to find good correspondences for image matching and manipulation. *CoRR*, abs/2007.08480, 2020. 15

[45] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *2014 IEEE Winter Conference on Applications of Computer Vision (WACV)*, volume 00, pages 75–82, March 2014. 16

[46] Jason J. Yu, Adam W. Harley, and Konstantinos G. Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *Computer Vision - ECCV 2016 Workshops - Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III*, pages 3–10, 2016. 12

[47] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Hongen Liao, and Long Quan. Learning two-view correspondences and geometry using order-aware network. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 5844–5853, 2019. 14, 15

[48] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *Int. J. Comput. Vis.*, 127(3):302–321, 2019. 5