FCSS: Fully Convolutional Self-Similarity for Dense Semantic Correspondence

Seungryong Kim[1], Dongbo Min[2], Bumsub Ham[1], Sangryul Jeon[1], Stephen Lin[3], Kwanghoon Sohn[1]
[1]Yonsei University [2]Chungnam National University [3]Microsoft Research
{srkim89,mimo,cheonjsr,khsohn}@yonsei.ac.kr dbmin@cnu.ac.kr stevelin@microsoft.com

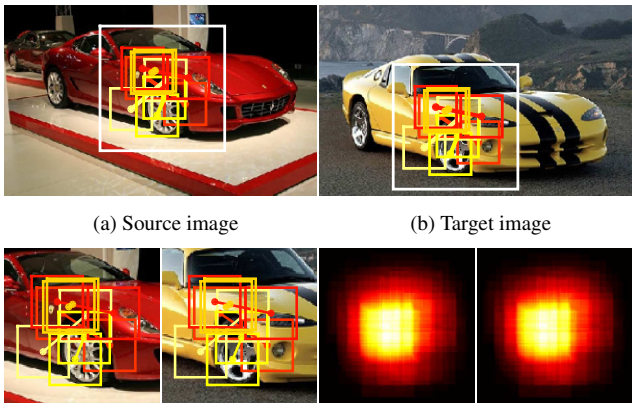
Abstract

We present a descriptor, called fully convolutional self-similarity (FCSS), for dense semantic correspondence. To robustly match points among different instances within the same object class, we formulate FCSS using local self-similarity (LSS) within a fully convolutional network. In contrast to existing CNN-based descriptors, FCSS is inherently insensitive to intra-class appearance variations because of its LSS-based structure, while maintaining the precise localization ability of deep neural networks. The sampling patterns of local structure and the self-similarity measure are jointly learned within the proposed network in an end-to-end and multi-scale manner. As training data for semantic correspondence is rather limited, we propose to leverage object candidate priors provided in existing image datasets and also correspondence consistency between object pairs to enable weakly-supervised learning. Experiments demonstrate that FCSS outperforms conventional handcrafted descriptors and CNN-based descriptors on various benchmarks.

1. Introduction

Establishing dense correspondences across *semantically* similar images is essential for numerous tasks such as scene recognition, image registration, semantic segmentation, and image editing [17, 31, 24, 48, 53]. Unlike traditional dense correspondence approaches for estimating depth [39] or optical flow [3, 44], in which *visually* similar images of the same scene are used as inputs, semantic correspondence estimation poses additional challenges due to intra-class variations among object instances, as exemplified in Fig. 1.

Often, basic visual properties such as colors and gradients are not shared among different object instances in the same class. These variations, in addition to other complications from occlusion and background clutter, lead to significant differences in appearance that can distract matching by handcrafted feature descriptors [34, 46]. Although powerful optimization techniques can help by enforcing smoothness constraints over a correspondence map [31, 24, 53, 45, 18], they are limited in effectiveness without a proper matching descriptor for semantic correspondence estimation.



(a) Source image (b) Target image (c) Window (d) Window (e) FCSS in (c) (f) FCSS in (d)

Figure 1. Visualization of local self-similarity. Even though there are significant differences in appearance among different instances within the same object class in (a) and (b), their local self-similarities computed by our FCSS descriptor are preserved as shown in (e) and (f), providing robustness to intra-class variations.

Over the past few years, convolutional neural network (CNN) based features have become increasingly popular for correspondence estimation thanks to their localization precision of matched points and their invariance to minor geometric deformations and illumination changes [19, 50, 41, 49]. However, for computing semantic correspondences within this framework, greater invariance is needed to deal with the more substantial appearance differences. This could potentially be achieved with a deeper convolutional network [42], but would come at the cost of significantly reduced localization precision in matching details (see [32, 21] for examples). Furthermore, as training data for semantic correspondence is rather limited, a network cannot be trained properly in a supervised manner.

To address these issues, we introduce a CNN-based descriptor that is inherently insensitive to intra-class appearance variations while maintaining precise localization ability. The key insight, illustrated in Fig. 1, is that among different object instances in the same class, their local structural layouts remain roughly the same. Even with dissimi-

lar colors, gradients, and small differences in feature positions, the local self-similarity (LSS) between sampled patch pairs is basically preserved. This property has been utilized for non-rigid object detection [40], sketch retrieval [5], and cross-modal correspondence [25]. However, existing LSS-based techniques are mainly handcrafted and need further robustness to capture reliable matching evidence from semantically similar images.

Our proposed descriptor, called fully convolutional self-similarity (FCSS), formulates LSS within a fully convolutional network in manner where the patch sampling patterns and self-similarity measure are both learned. We propose a convolutional self-similarity (CSS) layer that encodes the LSS structure and possesses differentiability, allowing for end-to-end training together with the sampling patterns. The convolutional self-similarities are measured at multiple scales, using skip layers [32] to forward intermediate convolutional activations. Furthermore, since limited training data is available for semantic correspondence, we propose a weakly-supervised feature learning scheme that leverages correspondence consistency within object candidate priors provided in existing datasets. Experimental results show that the FCSS descriptor outperforms conventional handcrafted descriptors and CNN-based descriptors on various benchmarks, including that of Taniai *et al.* [45], Proposal Flow [18], the PASCAL dataset [6], and Caltech-101 [13].

2. Related Work

Feature Descriptors Conventional gradient-based and intensity comparison-based descriptors, such as SIFT [34], HOG [8], DAISY [46], and BRIEF [4], have shown limited performance in a dense correspondence estimation across semantically similar but different object instances. Over the past few years, besides these handcrafted features, several attempts have been made using deep CNNs to learn discriminative descriptors for local patches from large-scale datasets. Some of these techniques have extracted immediate activations as the descriptor [16, 14, 9, 33], which have shown to be effective for patch-level matching. Other methods have directly learned similarity measures for comparing patches using a convolutional similarity network [19, 50, 41, 49]. Even though these CNN-based descriptors encode a discriminative structure with a deep architecture, they have inherent limitations in handling large intra-class variations [41, 10]. Furthermore, they are mostly tailored to estimate sparse correspondences, and cannot in practice provide dense descriptors due to their high computational complexity. Of particular importance, current research on semantic correspondence lacks an appropriate benchmark with dense ground-truth correspondences, making supervised learning of CNNs less feasible for this task.

LSS descriptor, proposed in [40], have achieved impressive results in object detection, image retrieval by sketching [40], deformable shape class retrieval [5], and cross-modal correspondence estimation [25, 26]. Inspired by LSS, among the more recent cross-modal descriptors is the dense adaptive self-correlation (DASC) descriptor [25], which provides satisfactory performance but is unable to handle non-rigid deformations due to its fixed patch pooling scheme. The deep self-correlation (DSC) descriptor [26] reformulates LSS in a deep non-CNN architecture. As all of these techniques use handcrafted descriptors, they lack the robustness to deformations that is possible with CNNs.

Dense Semantic Correspondence Many techniques for dense semantic correspondence employ handcrafted features such as SIFT [34] or HOG [8]. To improve the matching quality, they focus on optimization. Among these methods are some based on SIFT Flow [31, 24], which uses hierarchical dual-layer belief propagation (BP). Other instances include the methods with an exemplar-LDA approach [2], through joint image set alignment [53], or together with cosegmentation [45].

More recently, CNN-based descriptors have been used for establishing dense semantic correspondences. Pre-trained ConvNet features [27] were employed with the SIFT Flow algorithm [33] and with semantic flow using object proposals [18]. Choy *et al.* [7] proposed a deep convolutional descriptor based on fully convolutional feature learning and convolutional spatial transformer [23]. As these methods formulate the networks by combining successive convolutions only, they face a tradeoff between appearance invariance and localization precision that presents inherent limitations on semantic correspondence.

Weakly-Supervised Feature Learning For the purpose of object recognition, Dosovitskiy *et al.* [11] trained the network to discriminate between a set of surrogate classes formed by applying various transformations. For object matching, Lin *et al.* [28] proposed an unsupervised learning to learn a compact binary descriptor by leveraging an iterative training scheme. More closely related to our work is the method of Zhou *et al.* [52], which exploits cycle-consistency with a 3D CAD model [35] as a supervisory signal to train a deep network for semantic correspondence. However, the need to have a suitable 3D CAD model for each object class in a training stage limits its applicability.

3. The FCSS Descriptor

3.1. Problem Formulation and Overview

Let us define an image I such that $I_i : \mathcal{I} \rightarrow \mathbb{R}^3$ for pixel $i = [i_{\mathbf{x}}, i_{\mathbf{y}}]^T$. For each image point I_i, a dense descriptor $D_i : \mathcal{I} \rightarrow \mathbb{R}^L$ of dimension L is defined on a local support window. For LSS, this descriptor represents locally self-similar structure around a given pixel by recording the similarity between certain patch pairs within a local window. Formally, LSS can be described as a vector of feature

values $D_i = \bigcup_l D_i(l)$ for $l \in \{1, ..., L\}$, where the feature values are computed as

$$D_i(l) = \max_{j \in \mathcal{N}_i} \exp\left(-\mathcal{S}\left(P_{j-s_l}, P_{j-t_l}\right)/\lambda\right), \quad (1)$$

where $\mathcal{S}(P_{i-s_l}, P_{i-t_l})$ is a self-similarity distance between two patches P_{i-s_l} and P_{i-t_l} sampled on s_l and t_l, the l^{th} selected sampling pattern, around center pixel i. To alleviate the effects of outliers, the self-similarity responses are encoded by non-linear mapping with an exponential function of a bandwidth λ [1]. For spatial invariance to the position of the sampling pattern, the maximum self-similarity within a spatial window $\mathcal{N}_i$ is computed.

By leveraging CNNs, our objective is to design a dense descriptor that formulates LSS in a fully convolutional and end-to-end manner for robust estimation of dense semantic correspondences. Our network is built as a multi-scale series of convolutional self-similarity (CSS) layers where each includes a two-stream shifting transformer for applying a sampling pattern. To learn the network, including its self-similarity measures and sampling patterns, in a weakly-supervised manner, our network utilizes correspondence consistency between pairs of input images within object locations provided in existing datasets.

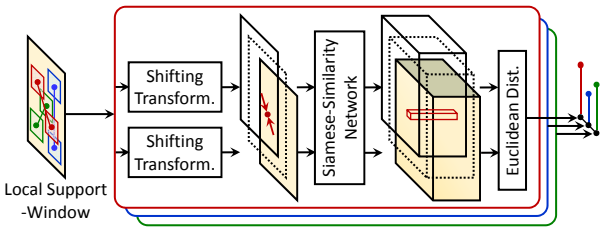
3.2. CSS: Convolutional Self-Similarity Layer

We first describe the convolutional self-similarity (CSS) layer, which provides robustness to intra-class variations while preserving localization precision of matched points around fine-grained object boundaries.

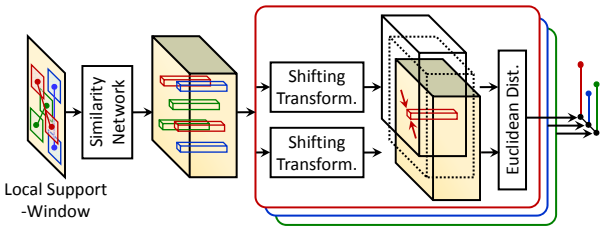
Convolutional Similarity Network Previous LSS-based techniques [40, 25, 26] evaluate (1) by sampling patch pairs and then computing their similarity using handcrafted metrics, which often fails to yield detailed matching evidence for estimating semantic correspondences. Instead, we compute the similarity of sampled patch pairs through CNNs. With l omitted for simplicity, the self-similarity between a patch pair P_{i-s} and P_{i-t} is formulated through a Siamese network, followed by decision or metric network [50, 19] or a simple Euclidean distance [41, 49] as shown in Fig. 2(a). Specifically, convolutional activations through feed-forward processes $\mathcal{F}(P_{i-s}; \mathbf{W}_c)$ and $\mathcal{F}(P_{i-t}; \mathbf{W}_c)$ with CNN parameters $\mathbf{W}_c$ are used to measure self-similarity based on the Euclidean distance, such that

$$\mathcal{S}(P_{i-s}, P_{i-t}) = \|\mathcal{F}(P_{i-s}; \mathbf{W}_c) - \mathcal{F}(P_{i-t}; \mathbf{W}_c)\|^2. \quad (2)$$

Note that our approach employs the Siamese network to measure *self-similarity within a single image*, in contrast to recent CNN-based descriptors [41] that directly measure the *similarity between patches from two different images*. Computing $\mathcal{S}(P_{i-s}, P_{i-t})$ for all sampling patterns (s, t) in this network is time-consuming, since the number



(a) Straightforward implementation of CSS layer



(b) Efficient implementation of CSS layer

Figure 2. Convolutional self-similarity (CSS) layers, implemented as (a) straightforward and (b) efficient versions, measure the self-similarity $\mathcal{S}(P_{i-\mathbf{W}_s}, P_{i-\mathbf{W}_t})$ between a patch pair $P_{i-\mathbf{W}_s}$ and $P_{i-\mathbf{W}_t}$ from sampling patterns $\mathbf{W}_s$ and $\mathbf{W}_t$. With the efficient scheme, convolutional self-similarity is equivalently solved while avoiding repeated computations for convolutions.

of iterations through the Siamese network is linearly proportional to the number of sampling patterns. To expedite this computation, we instead generate the convolutional activations of an entire image by passing it through the CNN, similar to [22], and then measure the self-similarity for the sampling patterns directly on the convolutional activations $\mathbf{A}_i = \mathcal{F}(I_i; \mathbf{W}_c)$, as shown in Fig. 2(b). Formally, this can be written as

$$\mathcal{S}(P_{i-s}, P_{i-t}) = \|\mathbf{A}_{i-s} - \mathbf{A}_{i-t}\|^2. \quad (3)$$

With this scheme, the self-similarity is measured by running the similarity network only once, regardless of the number of sampling patterns. Interestingly, a similar computational scheme was used to measure the similarity between two different images in [51], whereas our scheme instead measures self-similarity within a single image.

Two-Stream Shifting Transformer The sampling patterns (s, t) of patch pairs are a critical element of local self-similarity. In our CSS layer, a sampling pattern for a pixel i can be generated by shifting the original activation $\mathbf{A}_i$ by s and t to form two different activations from which self-similarity is measured. While this spatial manipulation of data within the network could be learned and applied using a spatial transformer layer [23], we instead formulate a simplification of this, called a shifting transformer layer, in which the shift transformations s and t are defined as network parameters that can be learned because of the differentiability of the shifting transformer layer. In this way, the optimized sampling patterns can be learned in the CNN.

Concretely, the sampling patterns are defined as network parameters $\mathbf{W}_s = [\mathbf{W}_{s_\mathbf{x}}, \mathbf{W}_{s_\mathbf{y}}]^T$ and $\mathbf{W}_t = [\mathbf{W}_{t_\mathbf{x}}, \mathbf{W}_{t_\mathbf{y}}]^T$ for all (s,t). Since the shifted sampling is repeated in an (integer) image domain, the convolutional self-similarity activation $\mathbf{A}_i$ is shifted simply without interpolation in the image domain according to the sampling patterns. We first define the sampled activations though a two-stream shifting transformer as

$$\mathbf{A}_{i-\mathbf{W}_s} = \mathcal{F}(\mathbf{A}_i; \mathbf{W}_s), \quad \mathbf{A}_{i-\mathbf{W}_t} = \mathcal{F}(\mathbf{A}_i; \mathbf{W}_t). \quad (4)$$

From this, convolutional self-similarity is then defined as

$$\mathcal{S}(P_{i-\mathbf{W}_s}, P_{i-\mathbf{W}_t}) = \|\mathcal{F}(\mathbf{A}_i; \mathbf{W}_s) - \mathcal{F}(\mathbf{A}_i; \mathbf{W}_t)\|^2. \quad (5)$$

Note that $\mathcal{S}(P_{i-\mathbf{W}_s}, P_{i-\mathbf{W}_t})$ represents a convolutional self-similarity vector defined for all (s,t).

Differentiability of Convolutional Self-Similarity For end-to-end learning of the proposed descriptor, the derivatives for the CSS layer must be computable, so that gradients of the final loss can be back-propagated to the convolutional similarity and shifting transformer layers.

To obtain the derivatives for the convolutional similarity layer and the shifting transformer layers, we first compute the Taylor expansion of the shifting transformer activations, under the assumption that $\mathbf{A}_i$ is smoothly varying with respect to shifting parameters $\mathbf{W}_s$:

$$\mathbf{A}_{i-\mathbf{W}_s^n} = \mathbf{A}_{i-\mathbf{W}_s^{n-1}} + (\mathbf{W}_s^n - \mathbf{W}_s^{n-1}) \circ \nabla \mathbf{A}_{i-\mathbf{W}_s^{n-1}}, \quad (6)$$

where $\mathbf{W}_s^{n-1}$ represents the sampling patterns at the $(n-1)^{th}$ iteration during training, and $\circ$ denotes the Hadamard product. $\nabla \mathbf{A}_{i-\mathbf{W}_s^{n-1}}$ is a spatial derivative on each activation slice with respect to $\nabla_\mathbf{x}$ and $\nabla_\mathbf{y}$. By differentiating (6) with respect to $\mathbf{W}_{s_\mathbf{x}}^n$, we get the shifting parameter derivatives as

$$\frac{\partial \mathbf{A}_{i-\mathbf{W}_s^n}}{\partial \mathbf{W}_{s_\mathbf{x}}^n} = \nabla_\mathbf{x} \mathbf{A}_{i-\mathbf{W}_s^{n-1}}. \quad (7)$$

By the chain rule, with n omitted, the derivative of the final loss $\mathcal{L}$ with respect to $\mathbf{W}_{s_\mathbf{x}}$ can be expressed as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{s_\mathbf{x}}} = \frac{\partial \mathcal{L}}{\partial \mathbf{A}_{i-\mathbf{W}_s}} \frac{\partial \mathbf{A}_{i-\mathbf{W}_s}}{\partial \mathbf{W}_{s_\mathbf{x}}}. \quad (8)$$

Similarly, $\partial\mathcal{L}/\partial\mathbf{W}_{s_\mathbf{y}}$, $\partial\mathcal{L}/\partial\mathbf{W}_{t_\mathbf{x}}$, and $\partial\mathcal{L}/\partial\mathbf{W}_{t_\mathbf{y}}$ can be calculated.

Moreover, the derivative of the final loss $\mathcal{L}$ with respect to $\mathbf{A}_i$ can be formulated as

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{A}_i} &= \frac{\partial \mathcal{L}}{\partial \mathbf{A}_{i-\mathbf{W}_s}} \frac{\partial \mathbf{A}_{i-\mathbf{W}_s}}{\partial \mathbf{A}_i} + \frac{\partial \mathcal{L}}{\partial \mathbf{A}_{i-\mathbf{W}_t}} \frac{\partial \mathbf{A}_{i-\mathbf{W}_t}}{\partial \mathbf{A}_i} \\ &= \frac{\partial \mathcal{L}}{\partial \mathbf{A}_{i-\mathbf{W}_s}} + \frac{\partial \mathcal{L}}{\partial \mathbf{A}_{i-\mathbf{W}_t}}, \end{aligned} \quad (9)$$

since $\partial \mathbf{A}_{i-\mathbf{W}_s}/\partial \mathbf{A}_i$ is 1 on the pixel $i - \mathbf{W}_s$. In this way, the derivatives for the CSS layer can be computed.

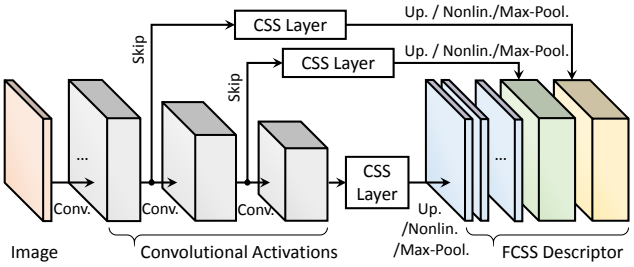


Figure 3. Network configuration of the FCSS descriptor, consisting of convolutional self-similarity layers at multiple scales.

3.3. Network Configuration for Dense Descriptor

Multi-Scale Convolutional Self-Similarity Layer In building the descriptor through a CNN architecture, there is a trade-off between robustness to semantic variations and fine-grained localization precision [32, 21]. The deeper convolutional layers gain greater robustness to semantic variations, but also lose localization precision of matching details around object boundaries. On the contrary, the shallower convolutional layers better preserve matching details, but are more sensitive to intra-class appearance variations.

Inspired by the skip layer scheme in [32], we formulate the CSS layers in a hierarchical manner to encode multi-scale self-similarities as shown in Fig. 3. Even though the CSS layer itself provides robustness to semantic variations and fine-grained localization precision, this scheme enables the descriptor to boost both robustness and localization precision. The CSS layers are located after multiple intermediate activations, and their outputs are concatenated to construct the proposed descriptor. In this way, the descriptor naturally encodes self-similarity at multiple scales of receptive fields, and further learns optimized sampling patterns on each scale. Many existing descriptors [21, 50] also employ a multi-scale description to improve matching quality.

For intermediate activations $\mathbf{A}_i^k = \mathcal{F}(I_i; \mathbf{W}_c^k)$, where $k \in \{1, ..., K\}$ is the level of convolutional activations and $\mathbf{W}_c^k$ is convolutional similarity network parameters at the k^{th} level, the self-similarity at the the k^{th} level is measured according to sampling patterns $\mathbf{W}_s^k$ and $\mathbf{W}_t^k$ as

$$\mathcal{S}(P_{i-\mathbf{W}_s^k}, P_{i-\mathbf{W}_t^k}) = \|\mathcal{F}(\mathbf{A}_i^k; \mathbf{W}_s^k) - \mathcal{F}(\mathbf{A}_i^k; \mathbf{W}_t^k)\|^2. \quad (10)$$

Since the intermediate activations are of smaller spatial resolutions than the original image resolution, we apply a bilinear upsampling layer [32] after each CSS layer.

Non-linear Gating and Max-Pooling Layer The CSS responses are passed through a non-linear gating layer to mitigate the effects of outliers [1]. Furthermore, since the pre-learned sampling patterns used in the CSS layers are fixed over an entire image, they may be sensitive to non-rigid deformation as described in [26]. To address this, we perform the max-pooling operation within a spatial window

$\mathcal{N}_i$ centered at a pixel i after the non-linear gating:

$$D_i^k = \max_{j \in \mathcal{N}_i} \exp(-\mathcal{S}(P_{j-\mathbf{W}_s^k}, P_{j-\mathbf{W}_t^k})/\mathbf{W}_\lambda^k), \quad (11)$$

where $\mathbf{W}_\lambda^k$ is a learnable parameter for scale k. The max-pooling layer provides an effect similar to using pixel-varying sampling patterns, providing robustness to non-rigid deformations. The descriptor for each pixel then undergoes L_2 normalization. Finally, the proposed descriptor $D_i = \bigcup_k D_i^k$ is built by concatenating feature responses across all scales. Fig. 3 displays an overview of the FCSS descriptor construction.

3.4. Weakly-Supervised Dense Feature Learning

A major challenge of semantic correspondence estimation with CNNs is the lack of ground-truth correspondence maps for training data. Constructing training data without manual annotation is difficult due to the need for semantic understanding. Moreover, manual annotation is very labor intensive and somewhat subjective. To overcome this, we propose weakly-supervised learning scheme based on correspondence consistency between image pairs. Unlike existing CNN-based descriptor learning methods which use a set of *patch pairs* [41, 50, 19], we use a set of *image pairs* for training. Such an image-wise learning scheme expedites feature learning by reducing the computational redundancy that occurs when computing convolutional activations for two adjacent pixels in the image. Our approach is conceptually similar to [7], but we learn the descriptor in a weakly-supervised manner that leverages correspondence consistency between each image pair so that the positive and negative samples are actively determined during training.

Correspondence Consistency Intuitively, the correspondence relation from a source image to a target image should be consistent with that from the target image to the source image. After forward-propagation with the training image pairs to obtain $\mathcal{F}(I;\mathbf{W})$ and $\mathcal{F}(I';\mathbf{W})$, the best match i^* for each pixel i is computed by comparing feature descriptors from the two images through nearest neighbor (NN) search [15]:

$$i^* = \text{argmin}_{i'} \|\mathcal{F}(I_i;\mathbf{W}) - \mathcal{F}(I'_{i'};\mathbf{W})\|^2. \quad (12)$$

$\mathbf{W} = \{\mathbf{W}_c^k, \mathbf{W}_s^k, \mathbf{W}_t^k, \mathbf{W}_\lambda^k \mid k = 1, ..., K\}$ represents all network parameters. After running NN search twice for the source and target images respectively, we check the correspondence consistency and identify the pixel pairs with valid matches as positive samples. Invalid matches are also used to generate negative samples. We randomly select the positive and negative samples during training. Since the negative samples ensue from erroneous local minima in the energy cost, they provide the effects of hard negative mining during training [41]. The feature learning begins by initializing the shifting transform with randomly selected sampling patterns. We found that even initial descriptors generated from the random patterns provide enough positive and negative samples to be used for weakly-supervised feature learning. A similar observation was also reported in [25].

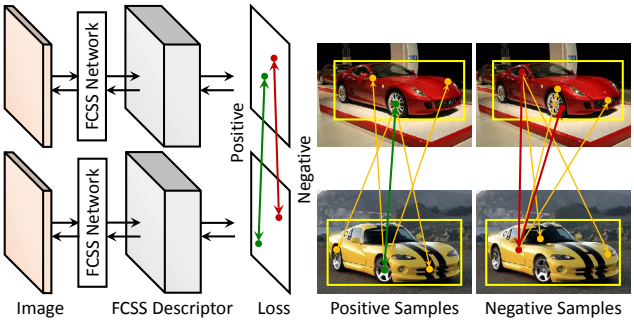


Figure 4. Weakly-supervised learning of the FCSS descriptor using correspondence consistency between object locations.

To boost this feature learning, we limit the correspondence candidate regions according to object location priors such as an object bounding box containing the target object to be matched, which are provided in most benchmarks [13, 12, 6]. Similar to [53, 52, 18], it is assumed that true matches exist only within the object region as shown in Fig. 4. Utilizing this prior mitigates the side effects that may occur due to background clutter when directly running the k-NN, and also expedites the feature learning process.

Correspondence Contrastive Loss For training the network with image pairs I and I', the correspondence contrastive loss [7] is defined as

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2N} \sum\nolimits_{i \in \Omega} l_i \|\mathcal{F}(I_i;\mathbf{W}) - \mathcal{F}(I'_{i'};\mathbf{W})\|^2 + (1 - l_i)\max(0, C - \|\mathcal{F}(I_i;\mathbf{W}) - \mathcal{F}(I'_{i'};\mathbf{W})\|^2), \quad (13)$$

where i and i' are either a matching or non-matching pixel pair, and l_i denotes a class label that is 1 for a positive pair and 0 otherwise. Ω represents the set of training samples, and N is the number of training samples. C is the maximal cost. The loss for a negative pair approaches zero as their distance increases. By back-propagating the partial derivative of $\mathcal{L}(\mathbf{W})$, the overall network can be learned.

4. Experimental Results and Discussion

4.1. Experimental Settings

For our experiments, we implemented the FCSS descriptor using the VLFeat MatConvNet toolbox [36]. For convolutional similarity networks in the CSS layers, we used the ImageNet pretrained VGG-Net [42] from the bottom conv1 to the conv3-4 layer, with their network parameters as initial values. Three CSS layers are located after conv2-2, conv3-2, and conv3-4, thus $K = 3$. Considering the trade-off between efficiency and robustness, the number of sampling

Methods	FD3D.	JODS	PASC.	Avg.
SIFT [31]	0.632	0.509	0.360	0.500
DAISY [46]	0.636	0.373	0.338	0.449
LSS [40]	0.644	0.349	0.359	0.451
DASC [25]	0.668	0.454	0.261	0.461
DeepD. [41]	0.684	0.315	0.278	0.426
DeepC. [50]	0.753	0.405	0.335	0.498
MatchN. [19]	0.561	0.380	0.270	0.404
LIFT [49]	0.730	0.318	0.306	0.451
VGG [42]	0.756	0.490	0.360	0.535
VGG w/S-CSS†	0.762	0.521	0.371	0.551
VGG w/S-CSS	0.775	0.552	0.391	0.573
VGG w/M-CSS	0.806	0.573	0.451	0.610
FCSS	**0.830**	**0.656**	**0.494**	**0.660**

Table 1. Matching accuracy for various feature descriptors with fixed SF optimization on the Taniai benchmark [45]. VGG w/S-CSS† denotes results with randomly selected sampling patterns.

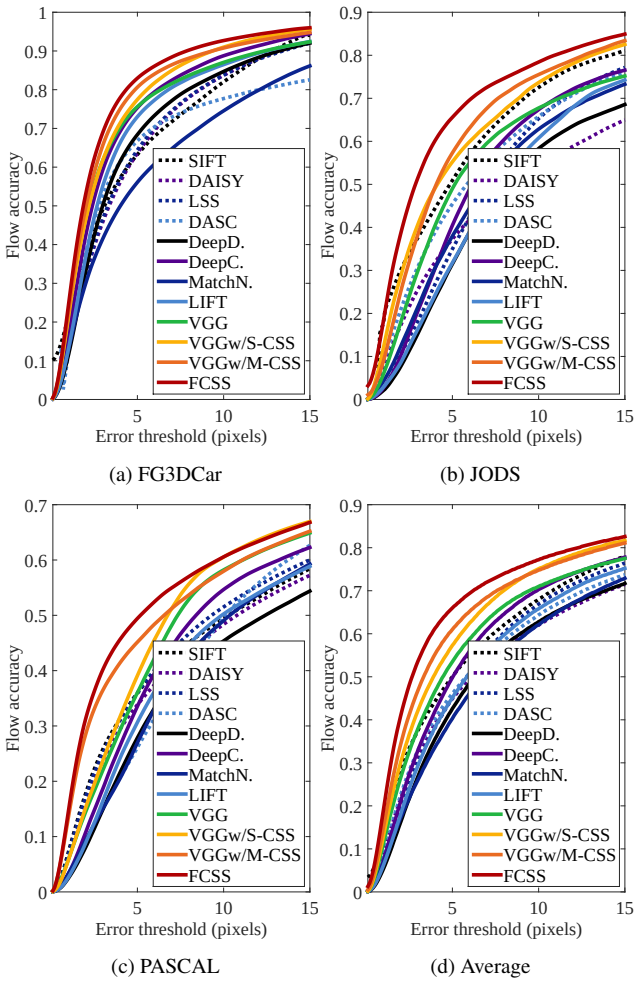
Methods	FG3D.	JODS	PASC.	Avg.
DFF [48]	0.495	0.304	0.224	0.341
DSP [24]	0.487	0.465	0.382	0.445
SIFT Flow [31]	0.632	0.509	0.360	0.500
Zhou *et al.* [52]	0.721	0.514	0.436	0.556
Taniai *et al.* [45]	0.830	0.595	0.483	0.636
Proposal Flow [18]	0.786	0.653	0.531	0.657
FCSS w/DSP [24]	0.527	0.580	0.439	0.515
FCSS w/SF [31]	0.830	**0.656**	0.494	0.660
FCSS w/PF [18]	**0.839**	0.635	**0.582**	**0.685**

Table 2. Matching accuracy compared to state-of-the-art correspondence techniques on the Taniai benchmark [45].

patterns is set to 64, thus the total dimension of the descriptor is $L = 192$. Before each CSS layer, convolutional activations are normalized with a L_2 norm [43]. To learn the network, we employed the Caltech-101 dataset [13] excluding testing image pairs used in experiments. The number of training samples N is 1024. C is set to 0.2. The learned parameters are used for all the experiments. Our code with pretrained parameters will be made publicly available.

In the following, we comprehensively evaluated our descriptor through comparisons to state-of-the-art handcrafted descriptors, including SIFT [34], DAISY [46], HOG [8], LSS [40], and DASC [25], as well as recent CNNs-based feature descriptors, including MatchNet (MatchN.) [19], Deep Descriptor (DeepD.) [41], Deep Compare (DeepC.) [50], UCN [7], and LIFT [49][1]. The performance was measured on Taniai benchmark [45], Proposal Flow dataset [18], PASCAL-VOC dataset [6], and Caltech-101 benchmark [13]. To additionally validate the components of the FCSS descriptor, we evaluated the initial VGG-Net (conv3-4) [42] (VGG), the VGG-Net with learned single-scale CSS layer (VGG w/S-CSS) and learned multi-scale CSS layers (VGG w/M-CSS)[2]. As an optimizer for estimating dense correspondence maps, we used the hierarchical dual-layer BP of the SIFT Flow (SF) optimization [31], whose code is publicly available. Furthermore, the performance of the FCSS descriptor when combined with other powerful optimizers was examined using the Proposal Flow (PF) [18] and the deformable spatial pyramid (DSP) [24].

[1]Since MatchN. [19], DeepC. [50], DeepD. [41], and LIFT [49] were developed for sparse correspondence, sparse descriptors were first built by forward-propagating images through networks and then upsampled.



(a) FG3DCar (b) JODS

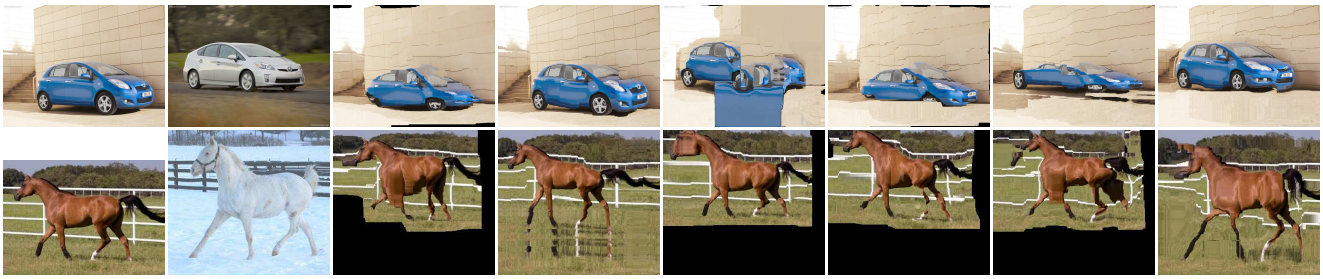
(c) PASCAL (d) Average

Figure 5. Average flow accuracy with respect to endpoint error threshold on the Taniai benchmark [45].

4.2. Results

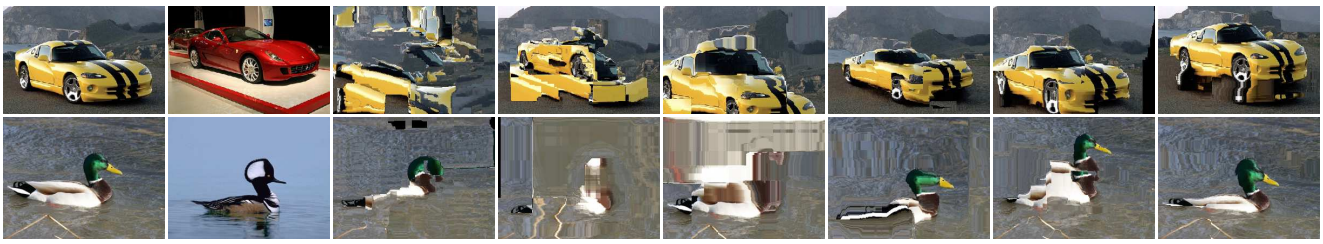
Taniai Benchmark [45] We first evaluated our FCSS descriptor on the Taniai benchmark [45], which consists of 400 image pairs divided into three groups: FG3DCar [29], JODS [37], and PASCAL [20]. As in [45], flow accuracy was measured by computing the proportion of foreground pixels with an absolute flow endpoint error that is smaller

[2]In the 'VGG w/S-CSS' and 'VGG w/M-CSS', the sampling patterns were only learned with VGG-Net layers fixed.



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

Figure 6. Qualitative results on the Taniai benchmark [45]: (a) source image, (b) target image, (c) SIFT [34], (d) DASC [25], (e) DeepD. [41], (f) MatchN. [19], (g) VGG [42], and (h) FCSS. The source images were warped to the target images using correspondences.



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

Figure 7. Qualitative results on the Proposal Flow benchmark [18]: (a) source image, (b) target image, (c) DAISY [46], (d) DeepD. [41], (e) DeepC. [50], (f) LIFT [49], (g) VGG [42], and (h) FCSS. The source images were warped to the target images using correspondences.

Methods	PCK		
	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.15$
SIFT [31]	0.247	0.380	0.504
DAISY [46]	0.324	0.456	0.555
LSS [40]	0.347	0.504	0.626
DASC [25]	0.255	0.411	0.564
DeepD. [41]	0.187	0.308	0.430
DeepC. [50]	0.212	0.364	0.518
MatchN. [19]	0.205	0.338	0.476
LIFT [49]	0.197	0.322	0.449
LIFT† [49]	0.224	0.346	0.489
VGG [42]	0.224	0.388	0.555
VGG w/S-CSS	0.239	0.422	0.595
VGG w/M-CSS	0.344	0.514	0.676
FCSS	**0.354**	**0.532**	**0.681**

Table 3. Matching accuracy for various feature descriptors with SF optimization on the Proposal Flow benchmark [18]. LIFT† denotes results of LIFT [49] with densely sampled windows.

than a certain threshold T, after resizing images so that its larger dimension is 100 pixels. Table 1 summarizes the matching accuracy for various feature descriptors with the SF optimization fixed ($T = 5$ pixels). Interestingly, while both the CNN-based descriptors [41, 50, 19, 49] and the handcrafted descriptors [34, 40, 46, 25] tend to show similar performance, our method outperforms both of these approaches. Fig. 5 shows the flow accuracy with varying error thresholds. Fig. 6 shows qualitative results. More results are available in the supplementary materials.

Table 2 compares the matching accuracy ($T = 5$ pixels)

Methods	PCK		
	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.15$
DSP [24]	0.239	0.364	0.493
SIFT Flow [31]	0.247	0.380	0.504
Zhou *et al.* [52]	0.197	0.524	0.664
Proposal Flow [18]	0.284	0.568	0.682
FCSS w/DSP [24]	0.302	0.475	0.602
FCSS w/SF [31]	**0.354**	0.532	0.681
FCSS w/PF [18]	0.295	**0.584**	**0.715**

Table 4. Matching accuracy compared to state-of-the-art correspondence techniques on the Proposal Flow benchmark [18].

with other correspondence techniques. Taniai *et al.* [45] and Proposal Flow [18] provide plausible flow fields, but their methods have limitations due to their usage of handcrafted features. Thanks to its invariance to intra-class variations and precise localization ability, our FCSS achieves the best results both quantitatively and qualitatively.

Proposal Flow Benchmark [18] We also evaluated our FCSS descriptor on the Proposal Flow benchmark [18], which includes 10 object sub-classes with 10 keypoint annotations for each image. For the evaluation metric, we used the probability of correct keypoint (PCK) between flow-warped keypoints and the ground truth [33, 18]. The warped keypoints are deemed to be correctly predicted if they lie within $\alpha \cdot \max(h, w)$ pixels of the ground-truth keypoints for $\alpha \in [0, 1]$, where h and w are the height and width of the object bounding box, respectively.

The PCK values were measured for various feature de-

(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 8. Visualizations of dense flow field with color-coded part segments on the PASCAL-VOC part dataset [6]: (a) source image, (b) target image, (c) source mask, (d) LSS [38], (e) DeepD. [41], (f) DeepC. [50], (g) LIFT [49], (h) FCSS, and (i) target mask.

(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 9. Visualizations of dense flow fields with mask transfer on the Caltech-101 dataset [13]: (a) source image, (b) target image, (c) source mask, (d) SIFT [34], (e) DASC [25], (f) MatchN. [19], (g) LIFT [49], (h) FCSS, and (i) target mask.

Methods	IoU	PCK	
		$\alpha = 0.05$	$\alpha = 0.1$
FlowWeb [24]	0.43	0.26	-
Zhou *et al.* [52]	-	-	0.24
Proposal Flow [18]	0.41	0.17	0.36
UCN [7]	-	0.26	0.44
FCSS w/SF [31]	0.44	0.28	**0.47**
FCSS w/PF [18]	**0.46**	**0.29**	0.46

Table 5. Matching accuracy on the PASCAL-VOC part dataset [6].

scriptors with SF optimization fixed in Table 3, and for different correspondence techniques in Table 4. Fig. 7 shows qualitative results for dense flow estimation. Our FCSS descriptor with SF optimization shows competitive performance compared to recent state-of-the-art correspondence methods. When combined with PF optimization instead, our method significantly outperforms the existing state-of-the-art descriptors and correspondence techniques.

PASCAL-VOC Part Dataset [6] Our evaluations also include the dataset provided by [53], where the images are sampled from the PASCAL part dataset [6]. With human-annotated part segments, we measured part matching accuracy using the weighted intersection over union (IoU) score between transferred segments and ground truths, with weights determined by the pixel area of each part. To evaluate alignment accuracy, we measured the PCK metric using keypoint annotations for the 12 rigid PASCAL classes [47]. Table 5 summarizes the matching accuracy compared to state-of-the-art correspondence methods. Fig. 8 visualizes estimated dense flow with color-coded part segments. From the results, our FCSS descriptor is found to yield the highest matching accuracy.

Methods	LT-ACC	IoU	LOC-ERR
DSP [24]	0.77	0.47	0.35
SIFT Flow [31]	0.75	0.48	0.32
Proposal Flow [18]	0.78	0.50	0.25
VGG [42] w/SF [31]	0.78	0.51	0.25
FCSS w/SF [31]	0.80	0.50	**0.21**
FCSS w/PF [31]	**0.83**	**0.52**	0.22

Table 6. Matching accuracy on the Caltech-101 dataset [13].

Caltech-101 Dataset [13] Lastly, we evaluated our FCSS descriptor on the Caltech-101 dataset [13]. Following the experimental protocol in [24], we randomly selected 15 pairs of images for each object class, and evaluated matching accuracy with three metrics: label transfer accuracy (LT-ACC) [30], the IoU metric, and the localization error (LOC-ERR) of corresponding pixel positions. Table 6 summarizes the matching accuracy compared to state-of-the-art correspondence methods. Fig. 9 visualizes estimated dense flow fields with mask transfer. For the results, our FCSS descriptor clearly outperforms the comparison techniques.

5. Conclusion

We presented the FCSS descriptor, which formulates local self-similarity within a fully convolutional network. In contrast to previous LSS-based techniques, the sampling patterns and the self-similarity measure were jointly learned within the proposed network in an end-to-end and multi-scale manner. The network was additionally trained in a weakly-supervised manner, using correspondence consistency between object bounding boxes in the training image pairs. We believe FCSS can potentially benefit instance-level object detection and segmentation, thanks to its robustness to intra-class variations and precise localization ability.

Acknowledgement. This work was supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (R0124-16-0002, Emotional Intelligence Technology to Infer Human Emotion and Carry on Dialogue Accordingly). The work of B. Ham was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2017R1C1B2005584).

References

[1] M. J. Black, G. Sapiro, D. H. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Trans. IP*, 7(3):421–432, 1998.

[2] H. Bristow, J. Valmadre, and S. Lucey. Dense semantic correspondence where every pixel is a classifier. *In: ICCV*, 2015.

[3] D. Butler, J. Wulff, G. Stanley, and M. Black. A naturalistic open source movie for optical flow evaluation. *In: ECCV*, 2012.

[4] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. Brief : Computing a local binary descriptor very fast. *IEEE Trans. PAMI*, 34(7):1281–1298, 2011.

[5] K. Chatfield, J. Philbin, and A. Zisserman. Efficient retrieval of deformable shape classes using local self-similarities. *In:ICCV Workshop*, 2009.

[6] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasum, and A. Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. *In: CVPR*, 2014.

[7] C. B. Choy, Y. Gwak, and S. Savarese. Universal correspondence network. *In:NIPS*, 2016.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *In: CVPR*, 2005.

[9] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *In: ICML*, 2014.

[10] J. Dong and S. Soatto. Domain-size pooling in local descriptors: Dsp-sift. *In: CVPR*, 2015.

[11] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and B. T. Discriminative unsupervised feature learning with convolutional neural networks. *In: NIPS*, 2014.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisseman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[13] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Trans. PAMI*, 28(4):594–611, 2006.

[14] P. Fischer, A. Dosovitskiy, and T. Brox. Descriptor matching with convolutional neural networks: A comparison to sift. *arXiv:1405.5769*, 2014.

[15] V. Garcia, E. Devreuve, F. Nielsen, and M. Barlaud. K-nearest neighbor search: Fast gpu-based implementations and application to high-dimensional feature matching. *In: ICIP*, 2010.

[16] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional acitivation features. *In: ECCV*, 2014.

[17] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *In: SIGGRAPH*, 2011.

[18] B. Ham, M. Cho, C. Schmid, and J. Ponce. Proposal flow. *In: CVPR*, 2016.

[19] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. *In: CVPR*, 2015.

[20] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. *In: ICCV*, 2011.

[21] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. *In: CVPR*, 2015.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. PAMI*, 37(9):1904–1916, 2015.

[23] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. *In: NIPS*, 2015.

[24] J. Kim, C. Liu, F. Sha, and K. Grauman. Deformable spatial pyramid matching for fast dense correspondences. *In: CVPR*, 2013.

[25] S. Kim, D. Min, B. Ham, S. Ryu, M. N. Do, and K. Sohn. Dasc: Dense adaptive self-correlation descriptor for multi-modal and multi-spectral correspondence. *In: CVPR*, 2015.

[26] S. Kim, D. Min, S. Lin, and K. Sohn. Deep self-correlation descriptor for dense cross-modal correspondence. *In: ECCV*, 2016.

[27] K. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *In: NIPS*, 2012.

[28] K. Lin, J. Lu, C. S. Chen, and J. Zhou. Learning compact binary descriptors with unsupervised deep neural networks. *In: CVPR*, 2016.

[29] Y. L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. *In: ECCV*, 2014.

[30] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. *IEEE Trans. PAMI*, 33(12):2368–2382, 2011.

[31] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE Trans. PAMI*, 33(5):815–830, 2011.

[32] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *In: CVPR*, 2015.

[33] J. Long, N. Zhang, and T. Darrell. Do convnets learn correspondence? *In: NIPS*, 2014.

[34] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[35] Online. `http://www.shapenet.org/`.

[36] Online. `http://www.vlfeat.org/matconvnet/`.

[37] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. *In: CVPR*, 2013.

[38] S. Saleem and R. Sablatnig. A robust sift descriptor for multispectral images. *IEEE SPL*, 21(4):400–403, 2014.

[39] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1):7–42, 2002.

[40] E. Schechtman and M. Irani. Matching local self-similarities across images and videos. *In: CVPR*, 2007.

[41] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. *In: ICCV*, 2015.

[42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *In: ICLR*, 2015.

[43] H. O. Song, Y. Xiang, S. Jegelk, and S. Savarese. Deep metric learing via lifted structured feature embedding. *In: CVPR*, 2016.

[44] D. Sun, S. Roth, and M. J. Black. Secret of optical flow estimation and their principles. *In: CVPR*, 2010.

[45] T. Taniai, S. N. Sinha, and Y. Sato. Joint recovery of dense correspondence and cosegmentation in two images. *In: CVPR*, 2016.

[46] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans. PAMI*, 32(5):815–830, 2010.

[47] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. *In: WACV*, 2014.

[48] H. Yang, W. Y. Lin, and J. Lu. Daisy filter flow: A generalized discrete approach to dense correspondences. *In: CVPR*, 2014.

[49] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. Lift: Learned invariant feature transform. *In: ECCV*, 2016.

[50] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. *In: CVPR*, 2015.

[51] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *JMLR*, 17(1):2287–2318, 2016.

[52] T. Zhou, P. Krahenbuhl, M. Aubry, Q. Huang, and A. A. Efros. Learning dense correspondence via 3d-guided cycle consistency. *In: CVPR*, 2016.

[53] T. Zhou, Y. J. Lee, S. X. Yu, and A. A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. *In: CVPR*, 2015.