LSM: Learning Subspace Minimization for Low-level Vision – Supplementary Material –

A. Derivatives of various data terms $D(\boldsymbol{x})$

In Sec 3.4, we introduced two categories of tasks. Now, we show the first-order and the (approximated) second-order derivatives of the data terms, which compose the vector $\boldsymbol{d}$ and the (block) diagonal matrix $\boldsymbol{D}$ at each iteration.

Binary Image Labeling Recall that the first category is binary image labeling (interactive segmentation and video segmentation) as:

$$D(\boldsymbol{x}) = \sum_{\boldsymbol{p}} \alpha_{\boldsymbol{p}} \|\tau(\boldsymbol{x}_{\boldsymbol{p}}) - 1\|_2^2 + \beta_{\boldsymbol{p}} \|\tau(\boldsymbol{x}_{\boldsymbol{p}}) + 1\|_2^2, \quad \text{(A.1)}$$

where $\boldsymbol{p} = [x, y]^\top$ is a pixel coordinate, τ is an activation function to relax the binary label $\tau(\boldsymbol{x}_{\boldsymbol{p}})$ between $(+1, -1)$, and $\alpha_{\boldsymbol{p}}$ and $\beta_{\boldsymbol{p}}$ are the probabilities that $\tau(\boldsymbol{x}_{\boldsymbol{p}}) = +1$ or -1. Therefore, the first-order and the second-order derivatives at an intermediate solution $\boldsymbol{x}$ are:

$$\begin{aligned} \frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}} &= [(\alpha_{\boldsymbol{p}} + \beta_{\boldsymbol{p}})\tau(\boldsymbol{x}_{\boldsymbol{p}}) + (\beta_{\boldsymbol{p}} - \alpha_{\boldsymbol{p}})][\frac{\partial \tau(\boldsymbol{x}_{\boldsymbol{p}})}{\partial \boldsymbol{x}_{\boldsymbol{p}}}], \\ \frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2} &= (\alpha_{\boldsymbol{p}} + \beta_{\boldsymbol{p}})[\frac{\partial \tau(\boldsymbol{x}_{\boldsymbol{p}})}{\partial \boldsymbol{x}_{\boldsymbol{p}}}]^2, \end{aligned} \quad \text{(A.2)}$$

where we ignore the scale factor 2 for simplicity, and $\frac{\partial \tau(\boldsymbol{x}_{\boldsymbol{p}})}{\partial \boldsymbol{x}_{\boldsymbol{p}}}$ can be $1 - \tau^2(\boldsymbol{x}_{\boldsymbol{p}})$ for $tanh$ activation function.

Dense Correspondence Estimation The second category is the dense correspondence estimation (stereo matching and optical flow) where the data term is:

$$D(\boldsymbol{x}) = \sum_{\boldsymbol{p}} \|F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}}) - F_T(\boldsymbol{p})\|_2^2. \quad \text{(A.3)}$$

For stereo matching, the derivatives are derived as:

$$\begin{aligned} \frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}} &= \nabla_x F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}})^\top [F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}}) - F_T(\boldsymbol{p})], \\ \frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2} &= \|\nabla_x F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}})\|_2^2, \end{aligned} \quad \text{(A.4)}$$

where ∇_x is the gradient operator along the horizontal direction. $\nabla_x F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}})$ and $[F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}}) - F_T(\boldsymbol{p})]$ are vectors, so $\frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}}$ and $\frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2}$ are scalars, which is also a one-dimensional problem and can be unified with the binary image label tasks with the same network and the parameters.

For optical flow, $\boldsymbol{x}_{\boldsymbol{p}} = [u, v]^\top$ is a 2D vector and the derivatives are:

$$\begin{aligned} \frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}} &= \nabla F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}})^\top [F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}}) - F_T(\boldsymbol{p})], \\ \frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2} &= \nabla F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}})^\top \nabla F_S(\boldsymbol{p} + \boldsymbol{x}_{\boldsymbol{p}}), \end{aligned} \quad \text{(A.5)}$$

where ∇ is the gradient operator along both the horizontal and vertical direction. Therefore, $\frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}}$ is a 2×1 vector, and $\frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2}$ is a 2×2 matrix, which makes unification with other one-dimensional tasks difficult. To address this problem, we apply Cramer's rule [12] as follows:

- First, we compute the determinant of $\frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2}$ as $det_{\boldsymbol{p}}$.
- Next, we replace the first column of $\frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2}$ with $\frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}}$, and denote the determinant of the modified matrix as $det_{\boldsymbol{p}}^x$. Similarly, $det_{\boldsymbol{p}}^y$ is computed by replacing the second column of $\frac{\partial^2 D}{\partial \boldsymbol{x}_{\boldsymbol{p}}^2}$ with $\frac{\partial D}{\partial \boldsymbol{x}_{\boldsymbol{p}}}$.
- Finally, we collect $det_{\boldsymbol{p}}^x$ and $det_{\boldsymbol{p}}$ at all pixel locations as the minimization context, concatenate it with the image context to generate the subspace $\mathcal{V}_x$ for the horizontal component of the flow field. Similarly, the $det_{\boldsymbol{p}}^y$ and the $det_{\boldsymbol{p}}$ are collected as the minimization context for the vertical subspace $\mathcal{V}_y$. Thus the subspace generation for optical flow is unified with other one-dimensional tasks by generating the subspace for the horizontal and the vertical components of flow individually.

B. Anisotropic Diffusion

In addition to the above two categories of tasks, our LSM framework supports the anisotropic diffusion [18, 25] as:

$$D(\boldsymbol{x}) = \sum_{\boldsymbol{p}} \|\nabla F(\boldsymbol{p})\|_2^2 \|\boldsymbol{x}_{\boldsymbol{p}} - \boldsymbol{y}_{\boldsymbol{p}}\|_2^2, \quad \text{(A.6)}$$

where $\boldsymbol{y}$ includes be but not limited to a color image. Therefore, we have the corresponding derivatives as:

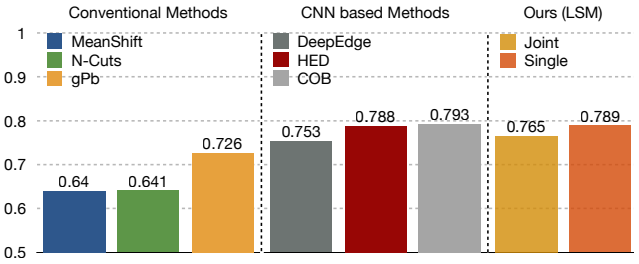
$$\begin{aligned}\frac{\partial D}{\partial \boldsymbol{x_p}} &= \|\nabla F(\boldsymbol{p})\|_2^2(\boldsymbol{x_p} - \boldsymbol{y_p}),\\ \frac{\partial^2 D}{\partial \boldsymbol{x_p^2}} &= \|\nabla F(\boldsymbol{p})\|_2^2.\end{aligned} \tag{A.7}$$

If $\boldsymbol{y}$ is multi-channel, Eq. (A.7) can be applied to each channel of $\boldsymbol{y}$ individually to maintain the unified formulation.

For a color image $\boldsymbol{y}$, there are two applications of the diffused image $\boldsymbol{x}$:

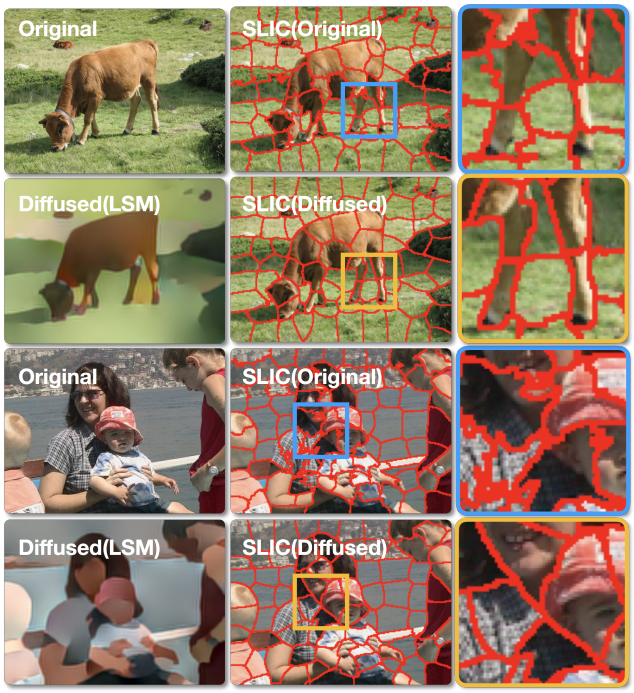
- The first application is **edge detection**, where we apply a 3×3 Sobel operator [21] on the diffused image $\boldsymbol{x}$ to detect edges, and use the same classification and regression loss from DeepEdge [5] for training. We quantitatively compare the results to several typical conventional methods, including MeanShift [6], Normalized Cuts [20], gPb [3] and Structured Random Forest [7], as well as more recent CNN-based methods, including DeepEdge [5], HED [27] and COB [17]. The comparisons are shown in Fig. A.1(a), where our LSM has achieved an F-score (ODS) of 0.789 and is comparable to other CNN based methods. As shown in Fig. A.1(b), our edges trade off recall for higher precision while HED tends to hallucinate edges on textured regions.

- We can also use a LSM diffused image $\boldsymbol{x}$ for **super-pixel** segmentation [20, 16, 1, 24]. In Fig. A.1(c), we qualitatively compare the SLIC [1] on the original and the diffused RGB images, where the super-pixels on the diffused image are more consistent with object boundaries than the ones on the original RGB images. Another advantage is that the diffused images is still only three- channel, which is more efficient for affinity computation than feature embeddings [23, 14].



(a) Average F-score (higher is better) of edge detection on diffused images.

(b) Our edges are qualitatively more precise than HED.



(c) SLIC super-pixels on the original v.s. the diffused RGB images.

Figure A.1: Results of anisotropic image diffusion. (a): The edges detected on the LSM diffused image are quantitatively comparable to other CNN based methods, (b): The detected edges are qualitatively more precise than HED, and (c): the SLIC [1] super pixels are more consistent with the object boundaries (*e.g.* cow's legs and human faces) on diffused images.

C. Network Structures

C.1. Feature Pyramid

The feature pyramid learns to extract feature maps that *not only* evaluate the data term $D(\boldsymbol{x})$, *but also* serve as the image context features for subspace generation. We use DRN-22 [28] as the backbone network for efficiency and denote the last residual blocks of conv6, conv5, conv4 and conv3 of DRN-22 as $\mathcal{C} = \{C^1, C^2, C^3, C^4\}$, with channels $\{512, 256, 128, 64\}$ and strides $\{32, 16, 8, 4\}$ respectively. We halve the channels of a feature map C^k by a 1×1 convolution, upsample it by factor of 2 with bilinear interpolation, concatenate the upsampled feature map with C^{k+1} in the next level, and finally apply a 3×3 convolution on the concatenated feature maps to reduce its dimensionality . This procedure is iterated until the finest level, which leads to the final feature pyramid $\mathcal{F} = \{F^1, F^2, F^3, F^4\}$ with the same

channels and strides as $\mathcal{C}$.

C.2. Subspace Generator

As introduced in Sec. 3.2, the subspace generator is a stack of residual blocks that generate the subspace $\mathcal{V}$ from the multi-scale context features. The channels for the multi-scale context features introduced in Sec. 3.2 are $\{512, 256, 128, 64\}$ for each pyramid level respectively, and the number of channels are maintain inside the residual blocks. The dimension K of the subspace $\mathcal{V}$, i.e. the output channels of the last convolutional layers of the corresponding subspace generator, are $\{2, 4, 8, 16\}$ at each pyramid level.

C.3. Model Efficiency

Our LSM model is efficient in terms of model size, training time, and inference time, which are contributed by integrating data terms explicitly.

Model Size We implement our LSM framework with the aforementioned settings, which contains about 15M parameters and costs 57.26 MB in memory. As shown in Fig. A.2, our LSM model maintains a relatively small model size when compared with other CNN based methods. But our LSM model handles multiple tasks within the same parameters while others are designed specifically for single tasks.

Training Efficiency We train our model with 143.2K iterations for all the experiments, which tasks roughly 20 hours and is relatively faster compared to existing CNN based methods. For example, training FlowNet2 [13] tasks more than 14 days and PWC-Net [22] takes 4.8 days. We initialize the backbone DRN-22 from the ImageNet pre-trained model, which also helps the training converges faster [11].

Inference Efficiency Our LSM framework is also efficient during inference. Since we unify different tasks into a single network, the inference times for various tasks are roughly the same, which consume about 25ms for 512×384 images. The computation is dominated by the feature pyramid construction, the subspace generation and the minimization.

D. Zero-shot Interactive Segmentation

Similar to the other zero-shot generalization tests in Sec. 4.3, we also leave the interactive segmentation out for testing and train on the other tasks. When interact only once, the average IoU is 0.802 for our LSM model learned on the other tasks and tested on the interactive segmentation. Which is still superior than the conventional method [10, 9] as shown in Fig. A.3.

E. Relation to Regularization Term

The Eq. (5) in Sec. 3.3 not only maintains the subspace constraint but also establishes the relation to the conventional energy minimization in Eq. (1), which contains the regularization term and is solved iteratively as:

$$\min_{\Delta \boldsymbol{x}} \frac{1}{2} \Delta \boldsymbol{x}^\top (\boldsymbol{D} + \boldsymbol{L}) \Delta \boldsymbol{x} + (\boldsymbol{d} + \boldsymbol{L}\boldsymbol{x})^\top \Delta \boldsymbol{x}, \qquad \text{(A.8)}$$

where $\boldsymbol{L}$ contains the second-order derivatives of the regularization term $R(\boldsymbol{x})$ and is called the generalized laplacian filtering matrix, and $\boldsymbol{L}\boldsymbol{x}$ is the first order derivatives of $R(\boldsymbol{x})$.

Proposition 1. *If $\boldsymbol{L} = \boldsymbol{D}(\boldsymbol{P} - \boldsymbol{I})$, and $\Delta \boldsymbol{x}$ is a solution to Eq. (A.8), its lower dimensional representation (coefficients) $\boldsymbol{c} = (\boldsymbol{V}^\top \boldsymbol{V})^{-1} \boldsymbol{V}^\top \Delta \boldsymbol{x}$ is the solution to Eq. (5).*

Proof. Let $\boldsymbol{L} = \boldsymbol{D}(\boldsymbol{P} - \boldsymbol{I})$,

$$\begin{aligned}
(\boldsymbol{D} + \boldsymbol{L})\Delta \boldsymbol{x} = & \\
&= [\boldsymbol{D} + \boldsymbol{D}(\boldsymbol{P} - \boldsymbol{I})]\Delta \boldsymbol{x} \\
&= (\boldsymbol{D} + \boldsymbol{D}\boldsymbol{P} - \boldsymbol{D})\Delta \boldsymbol{x} \\
&= (\boldsymbol{D}\boldsymbol{P})\Delta \boldsymbol{x} \\
&= \boldsymbol{D}\boldsymbol{V}(\boldsymbol{V}^\top \boldsymbol{V})^{-1} \boldsymbol{V}^\top \Delta \boldsymbol{x} \\
&= \boldsymbol{D}\boldsymbol{V}[(\boldsymbol{V}^\top \boldsymbol{V})^{-1} \boldsymbol{V}^\top \Delta \boldsymbol{x}] \\
&= \boldsymbol{D}\boldsymbol{V}\boldsymbol{c}.
\end{aligned}$$

Since $\Delta \boldsymbol{x}$ is a solution to Eq. (A.8) which satisfies the second-order optimality condition $(\boldsymbol{D} + \boldsymbol{L})\Delta \boldsymbol{x} = -(\boldsymbol{d} + \boldsymbol{L}\boldsymbol{x})$,

$$\begin{aligned}
\boldsymbol{V}^\top \boldsymbol{D}\boldsymbol{V}\boldsymbol{c} = & \\
&= -\boldsymbol{V}^\top (\boldsymbol{d} + \boldsymbol{L}\boldsymbol{x}) \\
&= -\boldsymbol{V}^\top [\boldsymbol{d} + \boldsymbol{D}(\boldsymbol{P} - \boldsymbol{I})\boldsymbol{x}] \\
&= -\boldsymbol{V}^\top (\boldsymbol{d} + \boldsymbol{D}\boldsymbol{r}),
\end{aligned}$$

which is the second-order optimality condition for Eq. (5), and $\boldsymbol{c} = -(\boldsymbol{V}^\top \boldsymbol{D}\boldsymbol{V})^{-1} \boldsymbol{V}^\top (\boldsymbol{d} + \boldsymbol{D}\boldsymbol{r})$. □

The Proposition.1 shows the relation between the proposed LSM framework and conventional variational minimization with regularization term.

In Eq. (A.8), $\boldsymbol{L}$ is manually defined according to the regularization term. If $R(\boldsymbol{x})$ is the Tikhonov regularization ($L2$ smoothness term), multiplication with $\boldsymbol{L}$ can be implemented efficiently as laplacian filtering, but its performance is poor. More sophisticated may give better performance, but is complicated and expensive, e.g. $\boldsymbol{L}$ is dense for non-local regularization [15, 26, 4, 19] and need to be approximated using high-dimensional filtering [8, 2], but the approximated multiplication still costs more than one minute [15].

In contrast, our LSM framework learns to generate the subspace $\mathcal{V}$ from data, so $\boldsymbol{L} = \boldsymbol{D}(\boldsymbol{P} - \boldsymbol{I})$ adapts to each sample and does not have a fixed form. Since we incorporate the minimization context in subspace generation, $\boldsymbol{L}$ progressively evolves and guides the minimization to a better solution, while conventionally it is defined only based

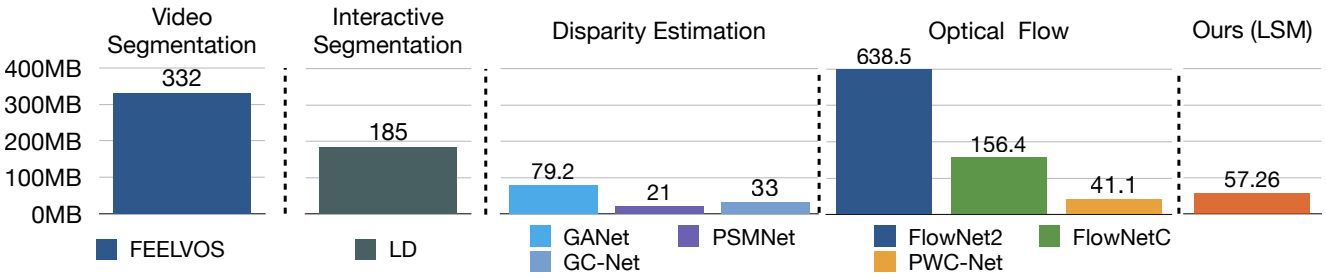


Figure A.2: Our LSM model handles multiple tasks in a relatively small model.

Figure A.3: Our zero-shot generalized LSM model performs better than GSC [10] for interactive segmentation.

on the image and fixed for all iterations. The LSM framework is also efficient, e.g. we can compute $\boldsymbol{V}^\top \boldsymbol{L}\boldsymbol{x}$ as $(\boldsymbol{V}^\top \boldsymbol{D}\boldsymbol{V})(\boldsymbol{V}^\top \boldsymbol{V})^{-1}\boldsymbol{V}^\top \boldsymbol{x} - \boldsymbol{V}^\top \boldsymbol{D}\boldsymbol{x}$, which avoids constructing the dense N-by-N matrix $\boldsymbol{L}$ and the intractable multiplication with it.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012. 2

[2] Andrew Adams, Jongmin Baek, and Myers Abraham Davis. Fast high-dimensional filtering using the permutohedral lattice. *Computer Graphics Forum*, 29(2):753–762, 2010. 3

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5): 898–916, 2011. 2

[4] Pablo Arias, Vicent Caselles, and Guillermo Sapiro. A variational framework for non-local image inpainting. In Daniel Cremers, Yuri Boykov, Andrew Blake, and Frank R. Schmidt, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, pages 345–358, 2009. 3

[5] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. Deepedge: A multi-scale bifurcated deep network for top-down contour detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2

[6] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(5):603–619, 2002. 2

[7] P. Dollár and C. L. Zitnick. Fast edge detection using structured forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(8):1558–1570, 2015. 2

[8] Eduardo S. L. Gastal and Manuel M. Oliveira. Domain transform for edge-aware image and video processing. *ACM Transactions of Graphics (TOG)*, 30(4):69:1–69:12, 2011. 3

[9] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(11):1768–1783, 2006. 3

[10] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman. Geodesic star convexity for interactive image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 3, 4

[11] Kaiming He, Ross B. Girshick, and Piotr Dollár. Rethinking imagenet pre-training. *CoRR*, abs/1811.08883, 2018. 3

[12] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, second edition, 2002. 1

[13] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[14] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Superpixel samping networks. In *European Conference on Computer Vision (ECCV)*, September 2018. 2

[15] Philipp Krähenbühl and Vladlen Koltun. Efficient nonlocal regularization for optical flow. In *European Conference on Computer Vision (ECCV)*, pages 356–369, 2012. 3

[16] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, Dec 2009. 2

[17] Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Pablo Arbeláez, and Luc Van Gool. Convolutional oriented boundaries. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[18] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. 1

[19] Matan Protter, Michael Elad, Hiroyuki Takeda, and Peyman Milanfar. Generalizing the non-local-means to super-resolution reconstruction. In *IEEE Transactions on Image Processing (TIP)*, page 36, 2009. 3

[20] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(8):888–905, 2000. 2

[21] I. Sobel and G. Feldman. A 3x3 isotropic gradient operator for image processing. *Pattern Classification and Scene Analysis*, page 271–272, 1968. 2

[22] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[23] Wei-Chih Tu, Ming-Yu Liu, Varun Jampani, Deqing Sun, Shao-Yi Chien, Ming-Hsuan Yang, and Jan Kautz. Learning superpixels with segmentation-aware affinity loss. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[24] Michael Van den Bergh, Xavier Boix, Gemma Roig, Benjamin Capitani, and Luc Van Gool. Seeds: Superpixels extracted via energy-driven sampling. volume 111, 10 2012. 2

[25] Joachim Weickert. Anisotropic diffusion in image processing. 1998. 1

[26] M. Werlberger, T. Pock, and H. Bischof. Motion estimation with non-local total variation regularization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2464–2471, 2010. 3

[27] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *IEEE International Conference on Computer Vision (ICCV)*, December 2015. 2

[28] F. Yu, V. Koltun, and T. Funkhouser. Dilated residual networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 636–644, 2017. 2