Efficient Large-scale Localization by Global Instance Recognition Supplementary Material

Fei Xue[†] Ignas Budvytis[†] Daniel Olmeda Reino[‡] Roberto Cipolla[†]
[†]University of Cambridge [‡]Toyota Motor Europe
{fx221, ib255, rc10001}@cam.ac.uk daniel.olmeda.reino@toyota-europe.com

A. Implementation

In this section, we first give a detailed description of the automatic global instance annotation strategy in Sec. A.1. Then, we introduce the augmentation of training data in Sec. A.2 and more details about the network and training process in Sec. A.3.

A.1. Automatic global instance annotation

For each image $I \in R^{H \times W}$ (H and W are the height and width of the image), we first utilize a building detection network based on Mask-RCNN [3] to detect all potential building instances $B = \{b^1, b^2, ..., b^n\}$. Candidates with low confidence (lower than 0.9) are discarded. Next, we build a map of the environment with an off-the-shelf SfM library such as colmap [12]. The 3D map provides pixel-wise correspondences for keypoints extracted from different images, which tell us which 2D building instances are identical in the 3D map. We then assign each identical building instance a global ID ranging from 1 (0 indicates background). Finally, for each image I, we obtain a dense segmentation map $S \in R^{H \times W}$ in which each pixel S_{ij} contains the global label $l = 0, 1, ..., N-1$ of the global building instance that it belongs to (N is the number of global instances).

All images in the Aachen dataset [11] are labeled automatically, though some minor mistakes exist at the boundaries due to the noise of building instance detection and structure-from-motion. However, as the quality of images in the RobotCar-Seasons (RoboCS) dataset [7] is relatively low because of motion blur and over-exposure, we can hardly detect building instances correctly with the pre-trained network. In this case, we have to annotate global instances manually. Fortunately, images in the RoboCS dataset are densely sampled from videos, providing much overlap for consecutive images. Therefore, we manually label 857 out 6954 images captured by the rear camera and train our recognition module with them. Ground-truth of unlabeled images are obtained from the prediction of our trained model. We have 452 and 692 global building instances in the Aachen and RoboCS datasets, respectively.

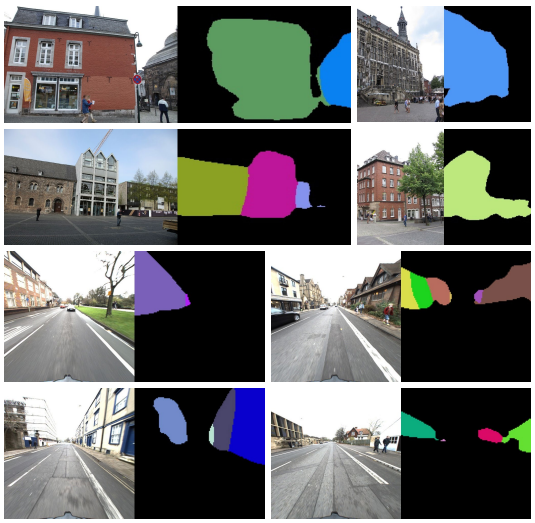

Figure 1. **Visualization ground-truth global instances.** Global instances in Aachen [11] dataset are annotated automatically while in RoboCS [7] they are predicted the model trained on a small subset of images in database which are manually labeled.

Fig. 1 shows some samples of generated global instances in the Aachen [11] and RoboCS [7] datasets. Since we obtain these *ground-truth* global instances automatically, we can see some minor mistakes at the boundaries.

A.2. Training data augmentation

In Aachen [11] and RoboCS [7] datasets, only daytime images are available for training, which however, are not enough to mitigate the domain gap between query (captured at different seasons, weather and illumination conditions) and reference images. Instead of training our model directly on the raw images, we take inspiration from [1,9] and augment the training data by generating more samples with style transfer techniques [6]. Some raw and stylized images are shown in Fig. 2. Although these stylized images are more artificial, they effectively augment the training data,

Figure 2. **Stylized images.** We adopt style transformation techniques [6] to generate more images for augmentation. Although the stylized images are more artificial, they effectively augment the training data, improving the generalization ability of our model.

improving the generalization ability of our model.

A.3. Network and training details

Network. We adopt the ResNet101 [4] pretrained on semi-weakly supervised ImageNet [13] as our encoder. Considering the efficiency, we set the input size for recognition to 256×256. As local features require higher resolution and lighter networks, the input size is 1024×1024 and only the first and second layers of Resnet101 are shared with local feature module. We additionally introduce two basic blocks [4] into the local feature network to enhance its ability of representation.

Training details. We implement the network in Pytorch [8]. We train the recognition module with Adam optimizer [5] with weight decay of 1e-5, batch size of 16, initial learning rate of 1e-4 for 120 epochs in total. Learning rate is adjusted to 1e-5 and 1e-6 after 80 and 100 epochs. As R2D2 [9], we train the local feature branch on the Aachen [11] dataset. The local feature module is trained with the same optimizer as the recognition with weight decay of 1e-5, batch size of 8, initial learning rate of 1e-4 for 40 epochs in total. The learning rate is adjusted to 1e-5 after 30 epochs. For the training of global feature branch, positive and negative samples are obtained according to their number of covisible 3D points in the map. Pairs with over 200 co-visible points are deemed as positive pairs, otherwise negative ones. We use the same optimizer as the recognition branch to train the global feature branch with batch size of 4 for 40 epochs. Each batch consists of 16 positive and 48 negative pairs.

B. Visualization of instance-wise feature detection

Fig. 4 illustrates of the distribution of valid 2D keypoints in the map reconstructed with R2D2 [9], Superpoint (SPP) [2], SPP+Superglue [2, 10], and our model. The track length (number of observations of each 3D point) is visualized with different colors (low to high). We also list the number of valid keypoints and the minimum, median, and maximum values of the track length at the top-left of each image. For all methods, we detect 1k keypoints to reconstruct the map. From Fig. 4, we can see that:

- Compared with SPP and SPP+Superflue, instance-wise detection enables our model to retain more valid keypoints in both summer ((1), (2)) and winter ((3), (4)) and most of them are on buildings rather than trees or other objects ((1), (2), (5)). Although the number of keypoints decreases as the size occluded regions increase ((5), (6), (7)), our model can still keep the close number of keypoints to SPP+Superglue, which is more than the number of SPP.

- SPP+Superglue obtains more valid keypoints than SPP, but fails to increase the number of track length because Superglue is able to establish correspondences for keypoints in difficult areas including trees (Fig. 4 (3)) and dynamic objects (Fig. 4 (5)). However, since all keypoints are detected by SPP, resulting in fewer on robust objects, both SPP and SPP+Superglue report the similar number of track length. While our instance-wise detection and matching can effectively increase the track length even under highly occluded conditions (Fig. 4 (7)).

- R2D2 [9] detects features from image pyramids, allowing cross-scale matching, so it produces sparser keypoints with much higher track length. As R2D2 is more like a uniform detector, it also gives more features on trees ((1), (2)) and is very sensitive to occluded images ((6), (7)).

C. Visualization of instance-wise matching

In this section, we first give a qualitative comparison of matches provided by R2D2 [9], SPP [2], SPP+Superglue [2, 10], and our method in Fig. 5. Then, we show more results of global instance prediction and instance-wise matching between the query and reference images in Aachen [11] and RoboCS datasets [7]. Query and predicted labels (top), reference and ground-truth labels (bottom), and their matches are shown in Fig. 6 and Fig. 7.

Qualitative comparison of matches. Fig. 5 shows the matches of R2D2, SPP, SPP+Superglue, and our approach

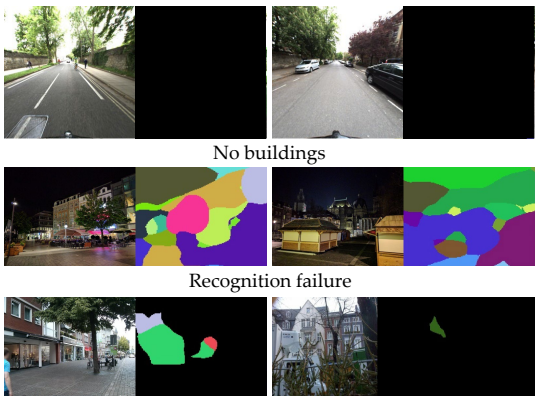

Figure 3. **Failed cases.** Our system fails sometimes due to no existence of buildings, recognition or local feature matching failure.

on images with a variety of season and illumination conditions and different extent of viewpoint changes. For simple images (Fig. 5 (1) (5)), both R2D2 and SPP provide many inliers. With advanced matcher, SPP+Superglue offers more inliers than R2D2 and SPP, while our model (without any advanced network for feature matching) yields close results to SPP+Superglue. As the change of viewpoint becomes larger (Fig. 5 (2) (3) (4)) or the illumination condition becomes worse (Fig. 5 (6) (7) (8)), all previous methods including SPP+Superglue suffer from dramatic decrease of inliers. However, our model still gives much more inliers because of our instance-wise detection and matching and the initially estimated pose for robust matching.

Instance prediction in Aachen dataset. Fig. 6 shows more cases with heavy occlusions, huge illumination changes, and large viewpoint variations. Due to the robustness of buildings to these challenges, our recognition module is still able to recognize global instances correctly. At the same time, our robust instance-wise detection and matching can take full advantage of even a limited number of correctly predicted pixels to produce enough inliers, boosting the robustness.

Instance prediction in RoboCS dataset. Fig. 7 shows that compared with general objects, *e.g.*, trees and traffic lanes, buildings are much more robust to appearance changes caused by changing seasons, weather, and illumination. Even with snow or at night time, when traffic lanes are not recognizable due to occlusion and low illumination, buildings are still very discriminative, making recognition-based localization succeed. Sometimes, due to very heavy occlusion or low illumination, our model can only recognize a small part of the building, yet several pixels are enough for local reference search and our robust instance-wise detection and matching ensure robust results even under these circumstances.

D. Failed cases

The global instances in our framework are defined on building facades, so the localization performance is influenced by the distribution of buildings. For scenes without buildings, we still have to execute global search frequently to find reference images. Besides, as many tasks, the recognition accuracy will decrease under extreme illumination changes, which also will impair the performance of both coarse and fine localization. We visualize some examples of failed cases due to no buildings, recognition failure, and local feature matching failure in Fig. 3.

References

[1] Asha Anoosheh, Torsten Sattler, Radu Timofte, Marc Pollefeys, and Luc Van Gool. Night-to-day image translation for retrieval-based localization. In *ICRA*, 2019. 1

[2] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, 2018. 2, 4, 5

[3] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2

[6] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *ECCV*, 2018. 1, 2

[7] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford RobotCar dataset. *IJRR*, 2017. 1, 2

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 2

[9] Jerome Revaud, Philippe Weinzaepfel, César Roberto de Souza, and Martin Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 1, 2, 4, 5

[10] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 2, 4, 5

[11] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, 2012. 1, 2

[12] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *CVPR*, 2016. 1

[13] I Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*, 2019. 2

Figure 4. **Qualitative comparison of feature detection.** We visualize valid 2D keypoints produced by R2D2 [9], SPP [2], SPP+Superglue [2, 10], and our model. Different colors indicate the number of track length (low to high). Samples from different seasons (summer: (1) (2), winter: (3) (4)) with different extent of occlusions ((5) (6) (7)) are visualized. The number of valid keypoints and minimum, median, and maximum values of observations are written at the top-left of each image.

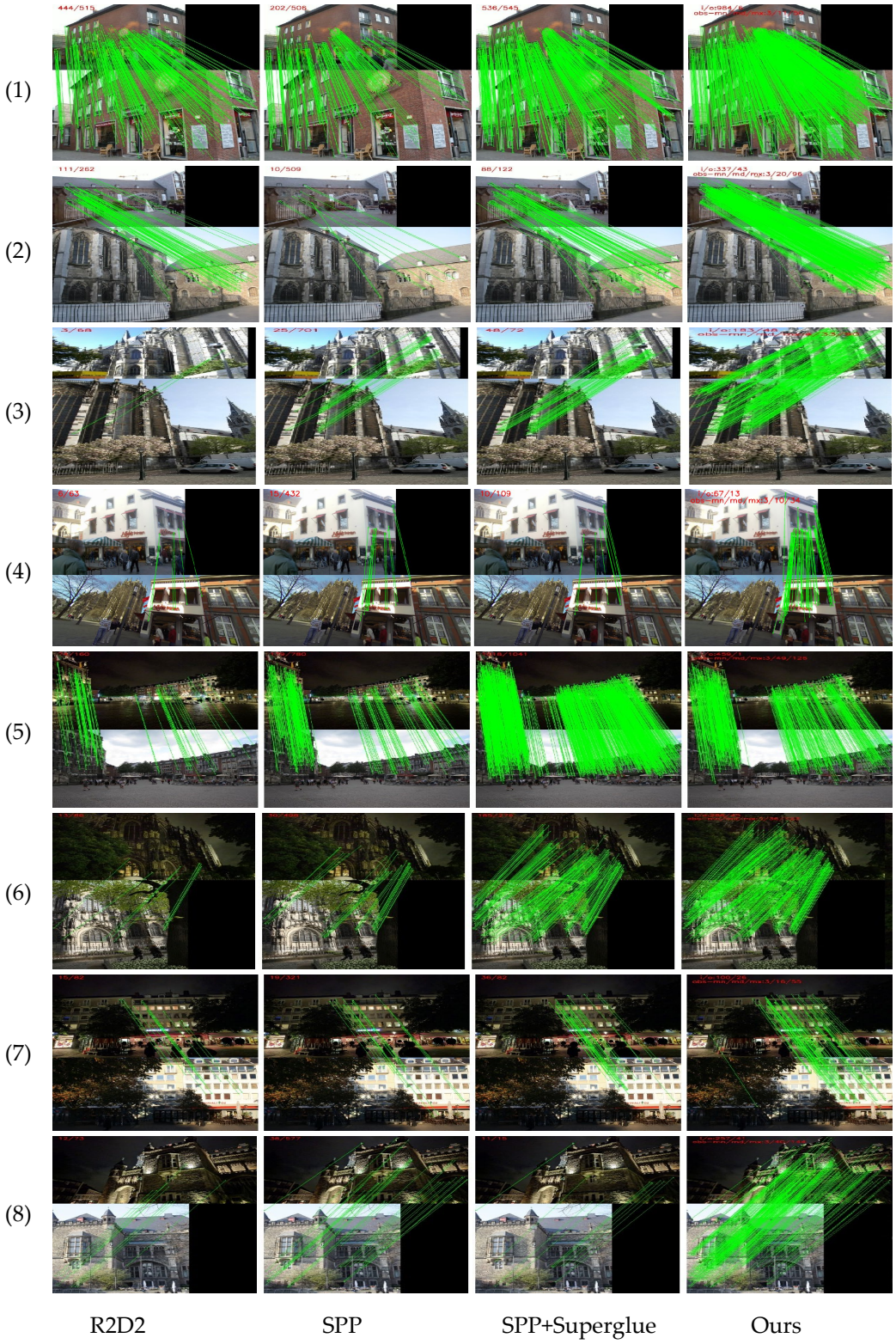

Figure 5. **Qualitative comparison of matches.** We visualize the matches provided by R2D2 [9], SPP [2], SPP+Superglue [2, 10], and our model. Query images are under a variety of seasons, illumination conditions, and viewpoint changes.

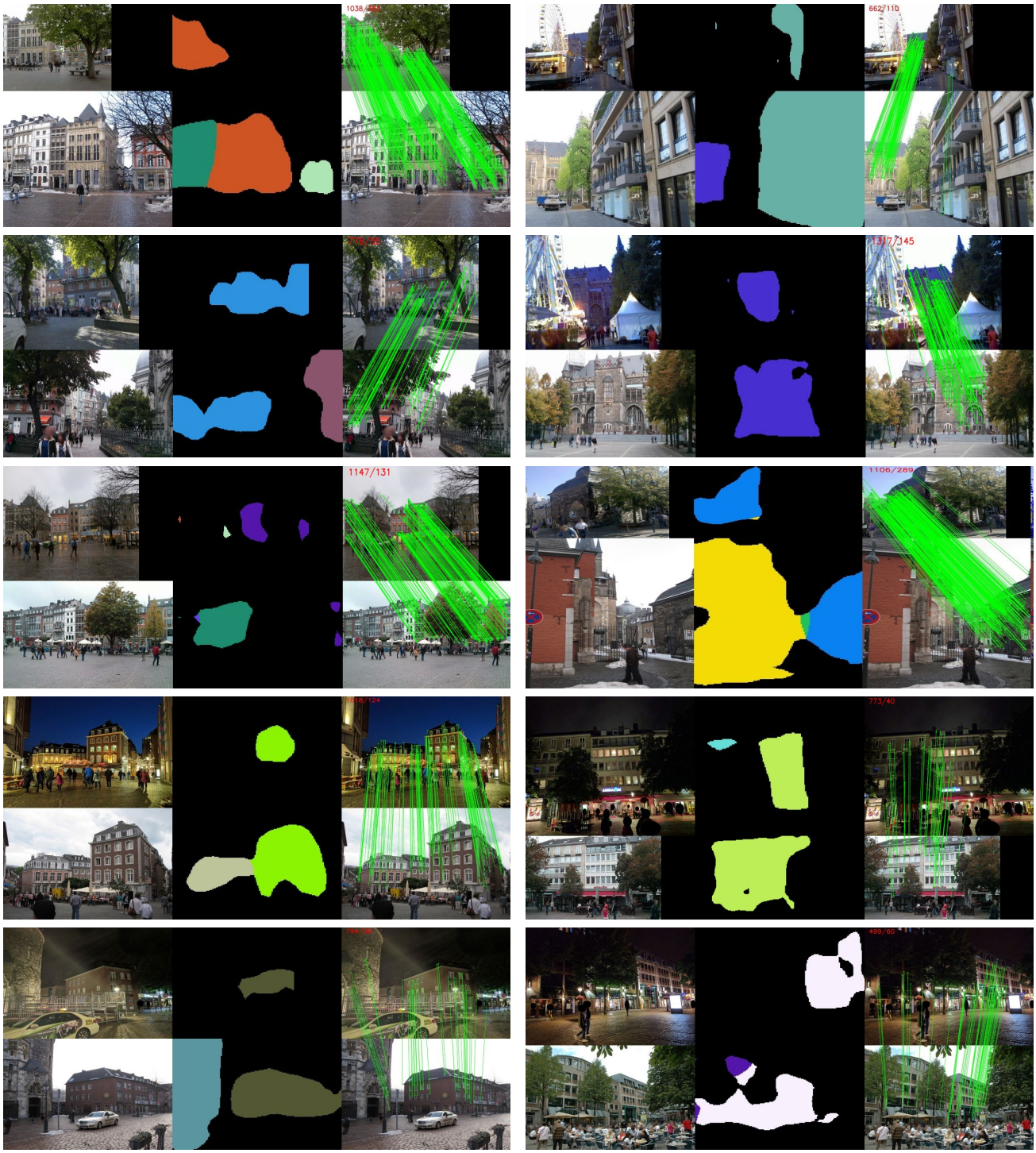

Figure 6. **Instance prediction on Aachen dataset.** We visualize instance-wise matching under heavy occlusions, huge illumination changes, and large viewpoint variations. These cases show that even with the aforementioned challenges, our recognition module is still able to recognize global instances correctly. Moreover, thanks to our robust instance-wise detection and matching, even a limited number of correctly predicted pixels can provide enough inliers for a successful pose estimation.

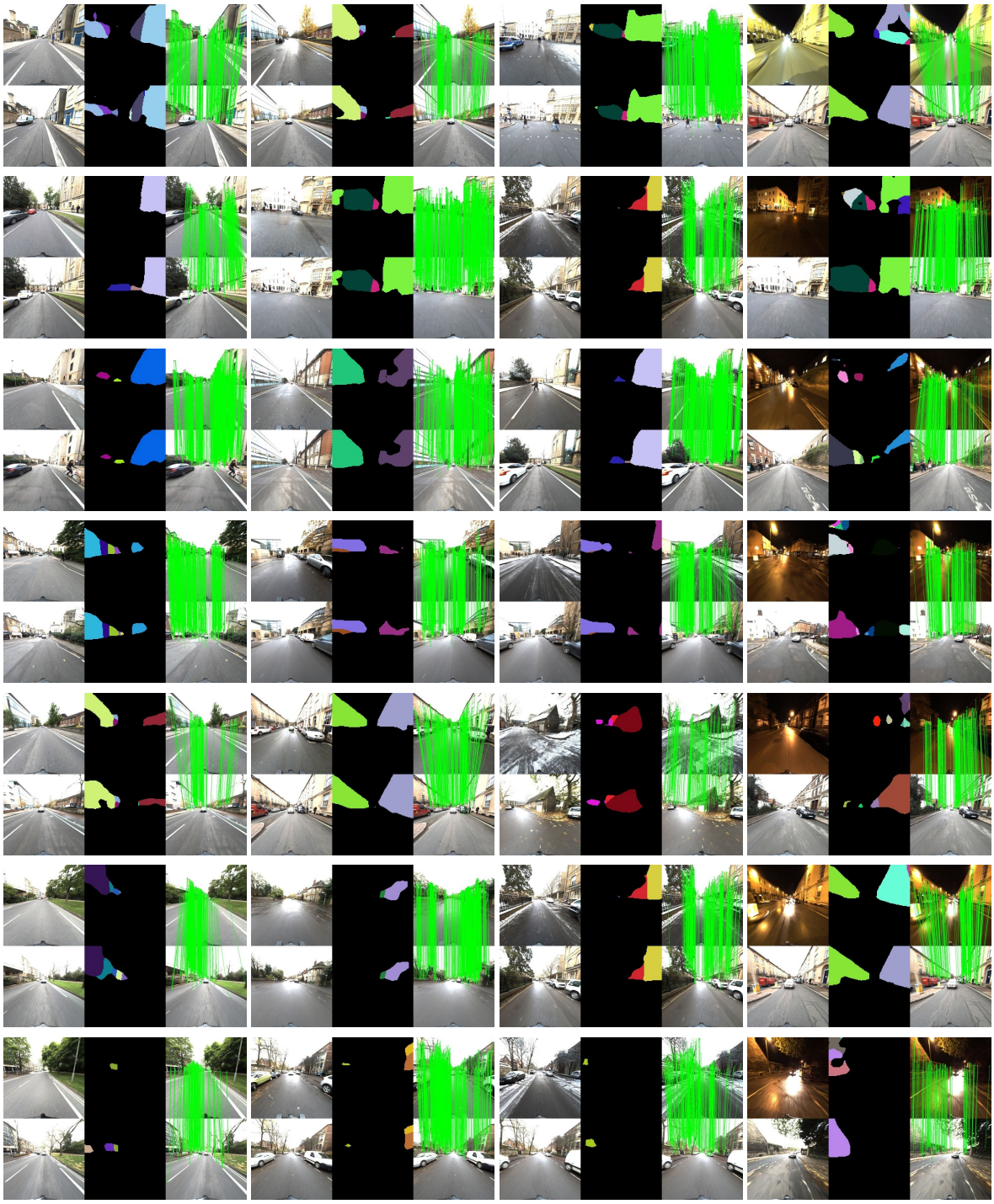

Figure 7. **Instance prediction on RobotCar.** We visualize the query image and predicted labels (top row), reference image and ground-truth labels (bottom row), and their matches. Query images are from different seasons (summer, winter), weather conditions (overcast, snow), and illuminations (day, night).