Traffic Sign Substitution

Exploring effective ways to substitute pre-existing traffic signs in images with other signs

Seminar/Project Computer Vision SS 2023

Final Presentation, June 13, 2023

Sayan Mandal

Motivation

- Detection and recognition of traffic signs is a key and challenging task for autonomous driving systems
- Dataset gathering, for training and testing such systems, is both costly and difficult

Solution?

- Find an effective augmentation technique to create new images from old ones
- In this work, we implement traffic sign substitution method as a way to create natural looking images by swapping already present traffic signs with new ones.

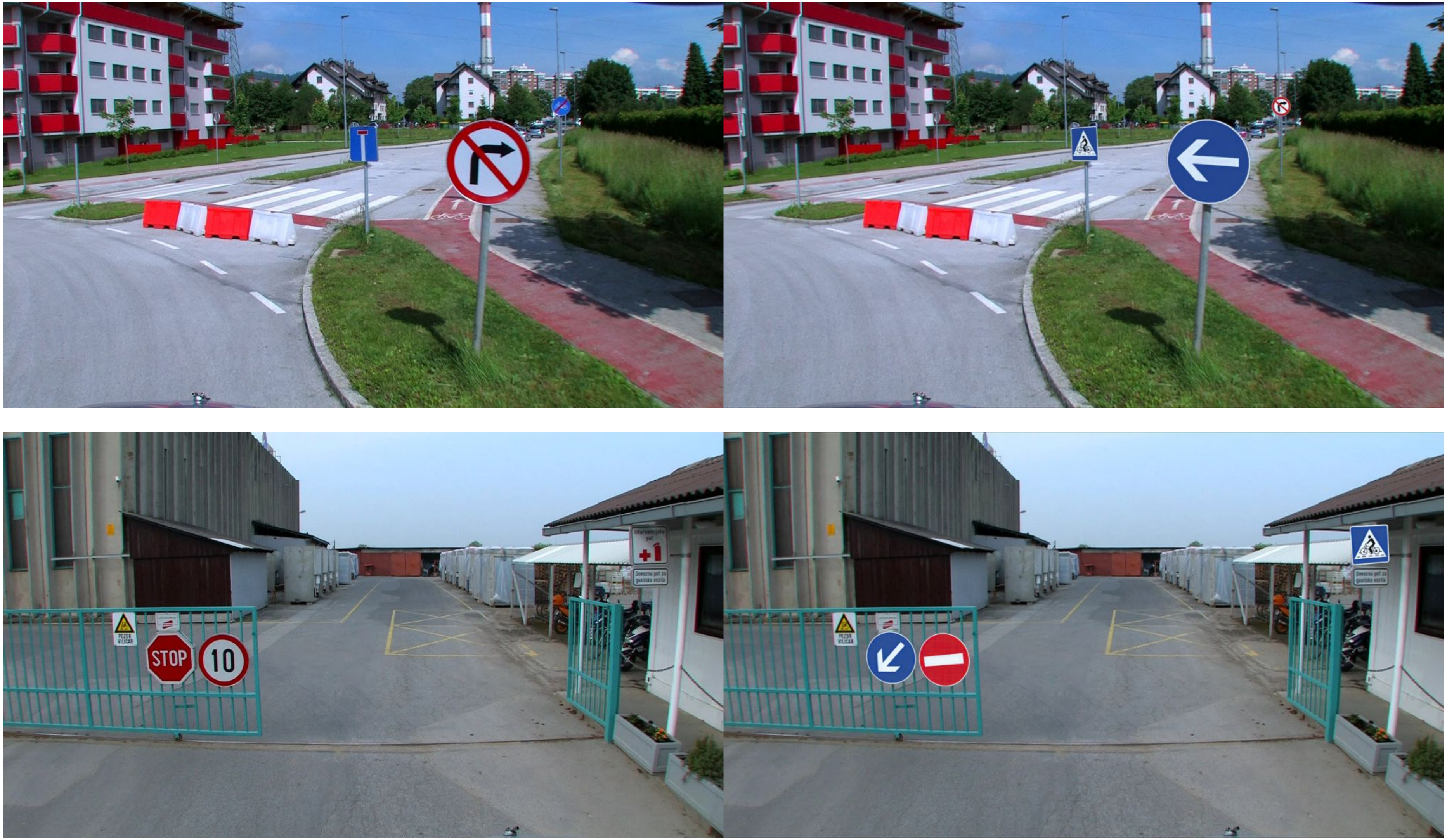


Fig 1: Original image in left and traffic sign substitution result in right. Images from DFG Traffic Sign Dataset [1]

[1] Tabernik, D., & Skočaj, D. (2019). Deep learning for large-scale traffic-sign detection and recognition. *IEEE transactions on intelligent transportation systems*, *21*(4), 1427-1440.

Problems faced in Image Composition

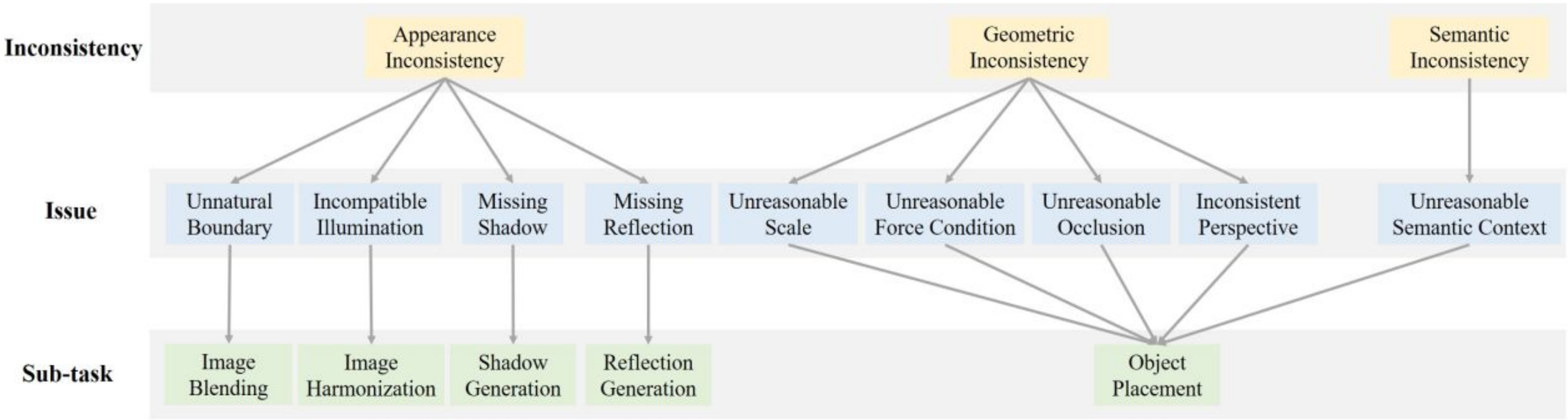


Fig 2: Image composition issues as highlighted in [2]

In this task we've looked into mitigating all the issues except for Missing Shadow and Missing Reflection, while performing substitutions

[2] Niu, L., Cong, W., Liu, L., Hong, Y., Zhang, B., Liang, J., & Zhang, L. (2021). Making images real again: A comprehensive survey on deep image composition. arXiv preprint arXiv:2106.14490.

Dataset

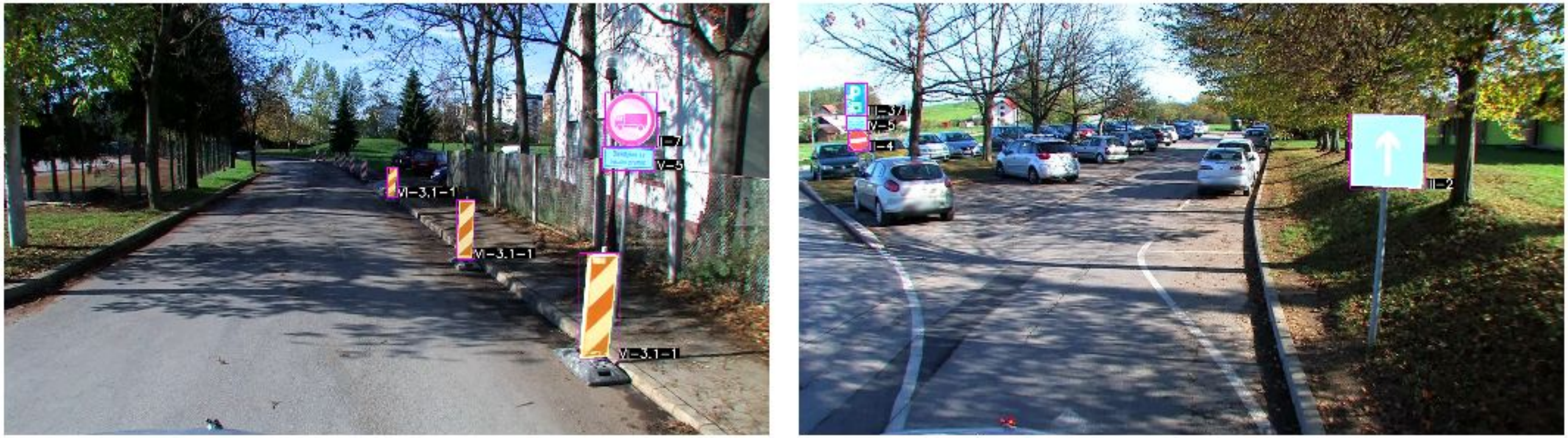


Fig 3: Two samples from the DFG Traffic Sign Dataset [1] with object label, bounding box and instance segmentation mask annotations

- 6957 1920x1080 RGB images with 13239 bbox & instance segmentation annotations corresponding to 200 categories.
- Captured across different Slovenian municipalities by driving a camera mounted car.

[1] Tabernik, D., & Skočaj, D. (2019). Deep learning for large-scale traffic-sign detection and recognition. *IEEE transactions on intelligent transportation systems*, *21*(4), 1427-1440.

Dataset

- For substitution we exclude the heterogenous (*arbitrary*) categories and only work with 116/200 categories
- We subdivided categories into *triangles*, *circles*, *squares*, *vertical_rect*, *horizontal_rect*, *arbitrary* super-categories and substitute a traffic sign with a random one from its super-category (excluding *arbitrary* super-category)
- Templates we downloaded from Wikimedia Commons or from dataset itself

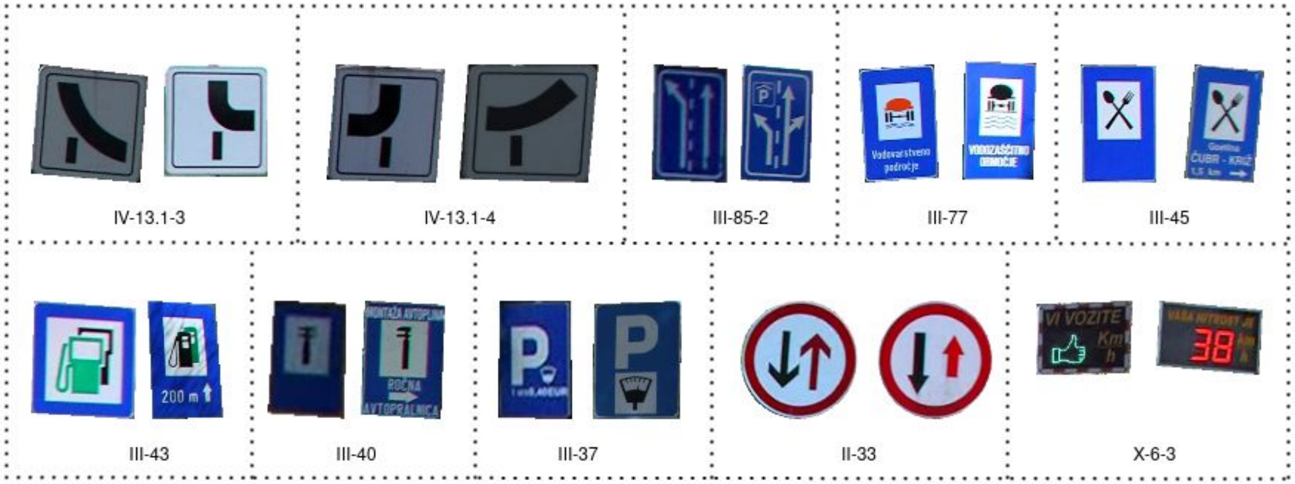
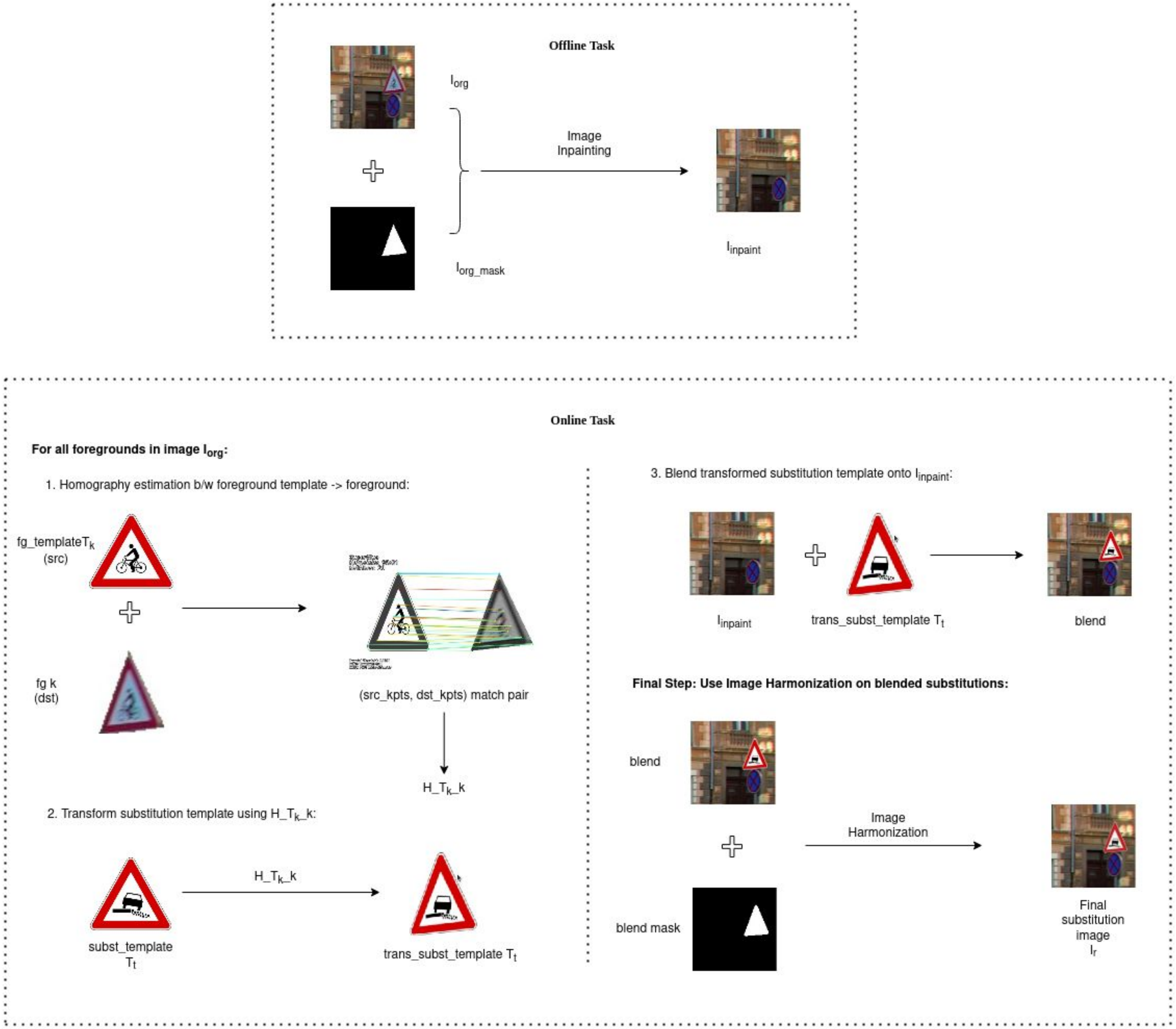


Fig 4: Examples of categories having heterogeneous content (*arbitrary* super-category). All samples are from DFG dataset

Method

Given a source image I_{org}, its instance segmentation map of k^{th} traffic sign M_k, k^{th} template T_k and a target template sign T_t:

1. Object Removal of k^{th} sign in I_{org} using Image Inpainting to get $I_{inpaint}$
2. Object Placement of T_t in the location of k^{th} sign in I_{org} with geometric consistency, via projective transformation using homography matrix H_T_k_k
3. Image blending of transformed T_t on $I_{inpaint}$ without visual artifacts and jagged edges
4. Image Harmonization of T_t in $I_{inpaint}$ w.r.t $I_{inpaint}$ background for appearance consistency to get result image I_r



Offline Task

I_{org}

Image Inpainting

$I_{inpaint}$

I_{org_mask}

Online Task

For all foregrounds in image I_{org}:

1. Homography estimation b/w foreground template -> foreground:

fg_templateT_k (src)

fg k (dst)

(src_kpts, dst_kpts) match pair

H_T_k_k

2. Transform substitution template using H_T_k_k:

H_T_k_k

subst_template T_t

trans_subst_template T_t

3. Blend transformed substitution template onto $I_{inpaint}$:

$I_{inpaint}$

trans_subst_template T_t

blend

Final Step: Use Image Harmonization on blended substitutions:

blend

blend mask

Image Harmonization

Final substitution image I_r

Fig 5: Our complete traffic sign substitution pipeline

Image Inpainting: LaMa & FMM

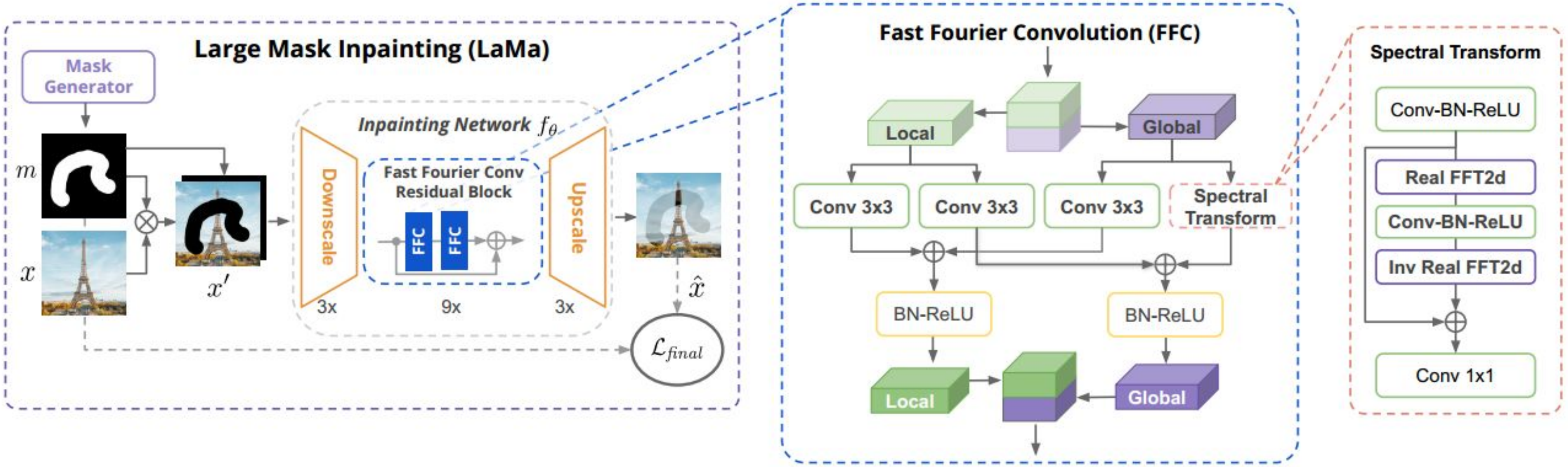


Fig 6: LaMa network architecture from the paper [3]

We tried two methods for this task: OpenCV's implementation of FMM [4] and Large Mask inpainting (LaMa) [3] (Big LaMa model)
LaMa uses Fast Fourier Convolution to get global context in the early layers itself

[3] Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., ... & Lempitsky, V. (2022). Resolution-robust large mask inpainting with fourier convolutions. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 2149-2159).
[4] Telea, A. (2004). An image inpainting technique based on the fast marching method. Journal of graphics tools, 9(1), 23-34.

Image Inpainting Results

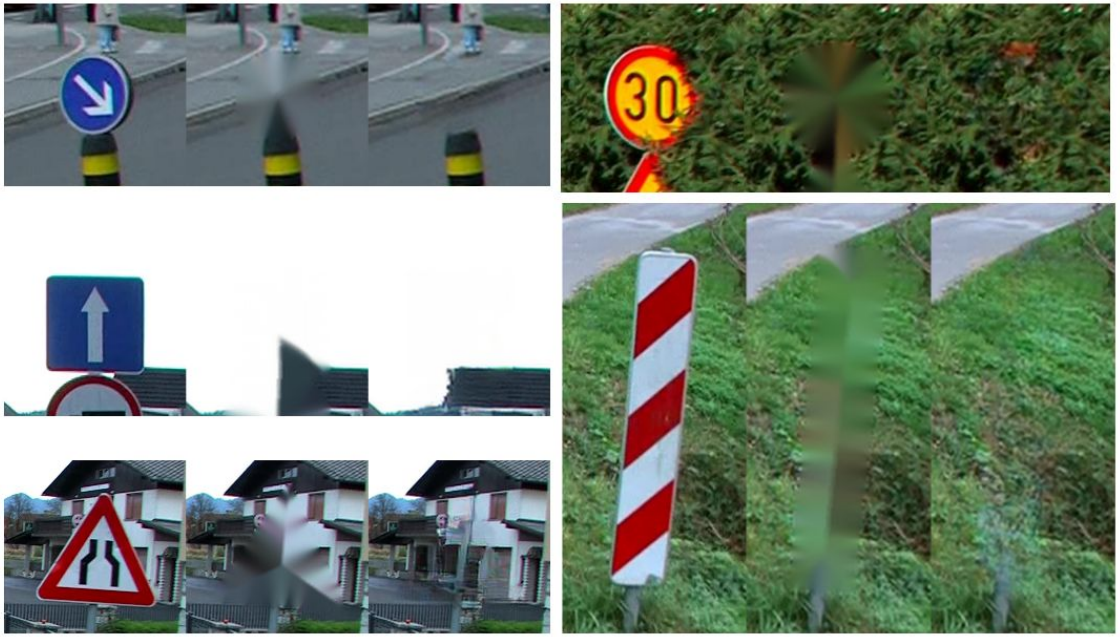


Fig 7: From left: Original image, FMM result, Big LaMa result

Method	**LPIPS** ↓[5]
FMM	0.00770
Big LaMa	**0.00745**

Table 1: LPIPS [5] score of FMM vs Big LaMa

[5] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 586-595).

Homography Estimation

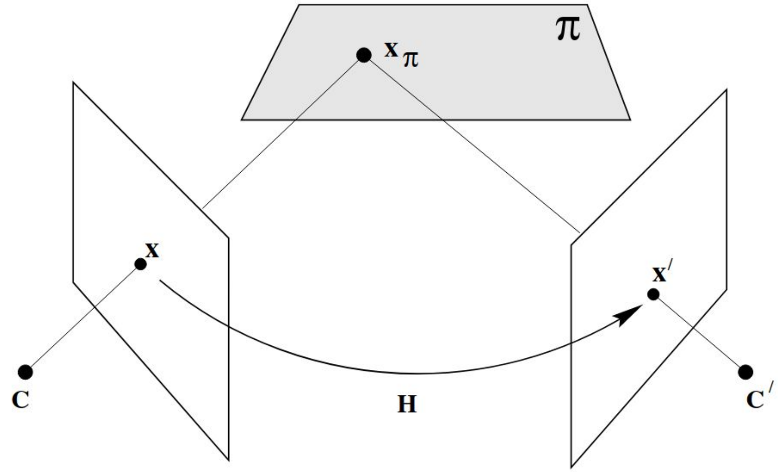


Fig 8: Plane induced homography (Fig. 13.1 in [6])

In order to transform the undistorted substitution template (**x**), to original sign's (**x′**) shape and size, we consider both the images as two views of the same plane **π**
Hence, the points on the template (**x**) are related to the points on the original sign (**x′**) by a planar homography **H** induced by the world plane **π**

[6] Hartley, R., & Zisserman, A. (2003). Multiple view geometry in computer vision. Cambridge university press.

Keypoint Detection: SuperPoint

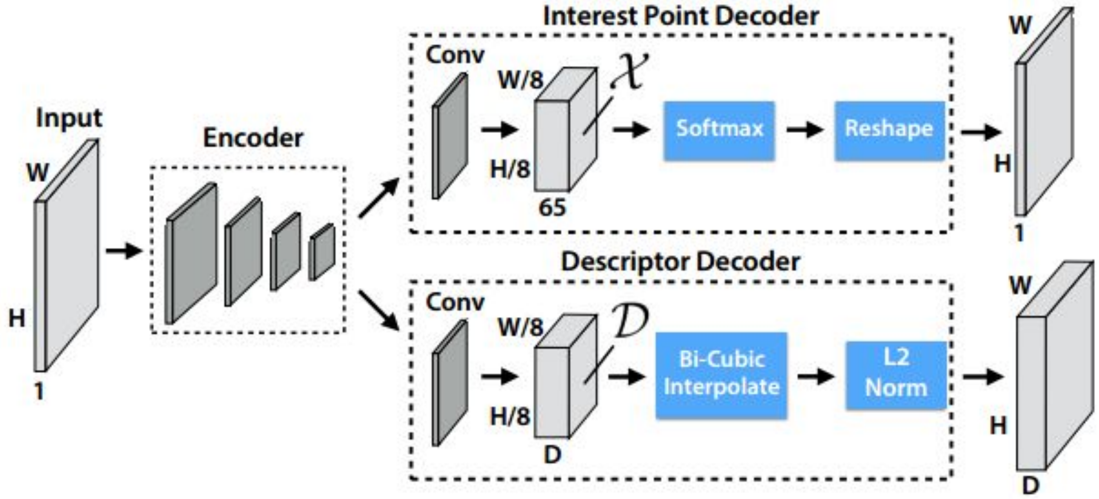


Fig 9: SuperPoint network architecture from the paper [7]

We experiment with ORB, AKAZE, BRISK keypoint detectors available in OpenCV and SuperPoint [7] network for keypoint detection

SuperPoint uses a VGG-style shared encoder for dimensionality reduction followed by two heads: Interest Point Decoder for interest point detection and Descriptor Decoder for getting L2-normalized fixed length descriptor.

[7] DeTone, D., Malisiewicz, T., & Rabinovich, A. (2018). Superpoint: Self-supervised interest point detection and description. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 224-236).

Keypoint Matching: SuperGlue

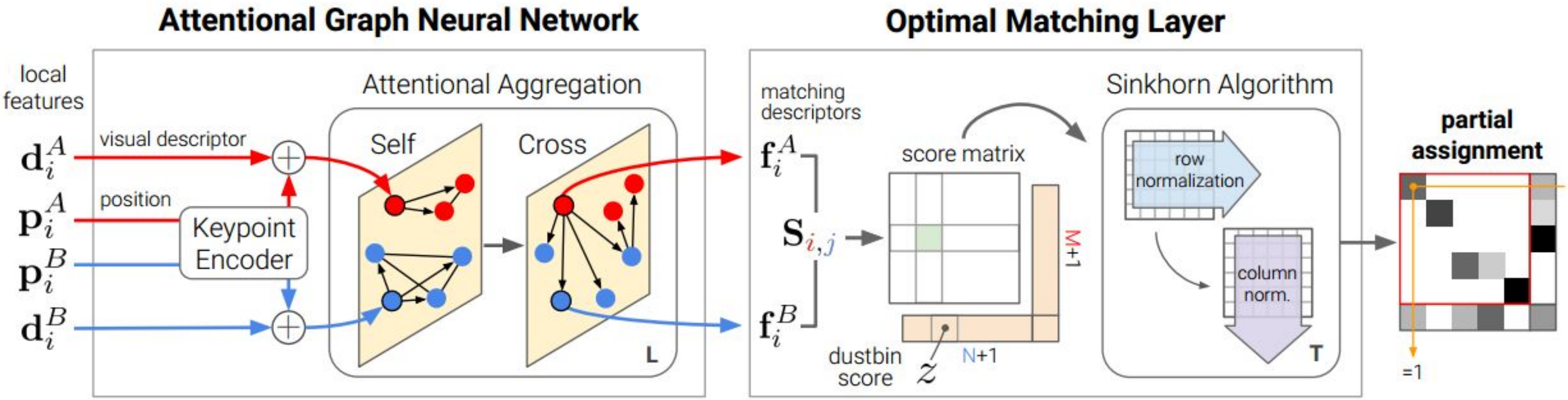


Fig 10: SuperGlue network architecture from the paper [8]

For keypoint matching, we used brute-force matcher for ORB, AKAZE and BRISK and SuperGlue [8] for SuperPoint

The Attentional GNN first uses a Keypoint Encoder to encode keypoint into a high-dimensional vector that later helps the GNN to reason appearance and position jointly. Optimal Matching Layer computes partial assignment from the feature descriptors extracted by Attentional GNN

For SuperPoint+SuperGlue we used Least-Squares and RANSAC for others.

[8] Sarlin, P. E., DeTone, D., Malisiewicz, T., & Rabinovich, A. (2020). Superglue: Learning feature matching with graph neural networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4938-4947).

Homography Estimation: Performance Study

Resolution	**Method**	**Misses↓**	**Average Matches↑**	**MACE↓**[9]
800x800	ORB	**766**	**66.903**	**63.1713**
	AKAZE	9453	17.5167	114.1825
	BRISK	2773	28.0403	89.2282
	SuperPoint	887	20.3456	99.1876
1920x1920	ORB	386	**154.1779**	24.7012
	AKAZE	2674	45.9755	35.2064
	BRISK	1174	53.6787	35.8634
	SuperPoint	**47**	50.9648	**10.8202**

Table 2: Efficacy of different keypoint methods for homography estimation on image pair with **BLACK** background. For matching, SuperGlue was used with SuperPoint and BFMatcher was used for all other methods. MACE is Mean Average Corner Error. Best results highlighted in **bold**

[9] DeTone, D., Malisiewicz, T., & Rabinovich, A. (2016). Deep image homography estimation. arXiv preprint arXiv:1606.03798.

Homography Estimation: Performance Study

Resolution	Method	Misses↓	Average Matches↑	MACE↓
800x800	ORB	**97**	**68.5059**	79.1818
	AKAZE	4080	21.9332	87.3210
	BRISK	895	27.7633	225.5695
	SuperPoint	1008	21.0290	**59.5252**
1920x1920	ORB	32	**162.9177**	12.332
	AKAZE	133	55.1918	23.8899
	BRISK	174	56.1642	44.1530
	SuperPoint	**0**	51.8875	**5.1721**

Table 3: Efficacy of different keypoint methods for homography estimation on image pair with **WHITE** background. For matching, SuperGlue was used with SuperPoint and BFMatcher was used for all other methods. MACE is Mean Average Corner Error. Best results highlighted in **bold**

SuperPoint+SuperGlue Results

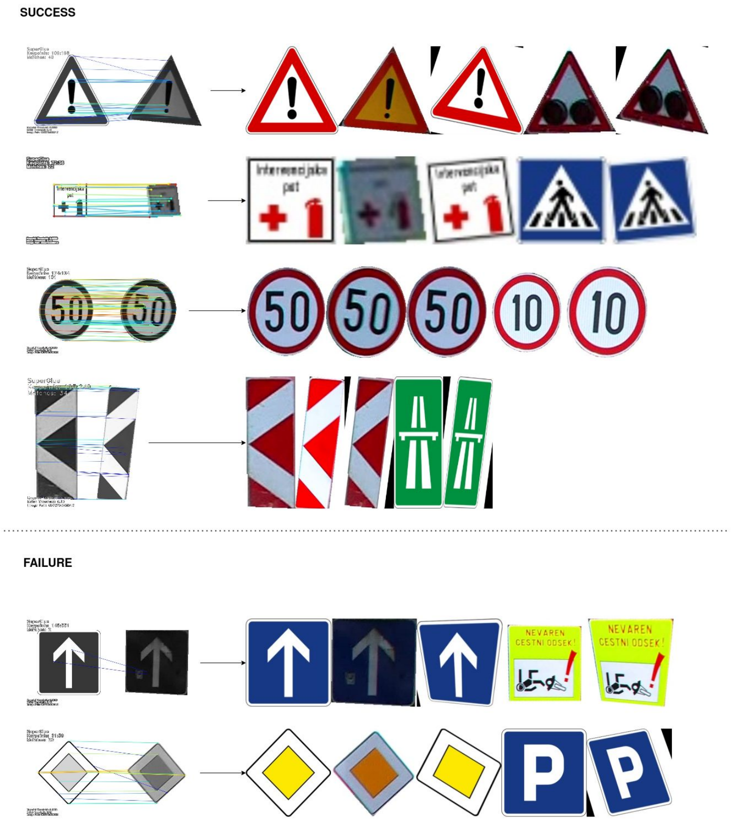


Fig 11: Homography estimation results of SuperPoint+SuperGlue. From top to bottom: Samples of success cases vs failure cases. From left to right: First image shows the matched keypoints between template T_k and original sign k. Second image shows template, original,transformed template via $H\_T_k\_k$, substitution template, transformed substitution template via $H\_T_k\_k$

Image Blending

After we get the transformed substitution template, we first apply a 3 × 3 Gaussian blur to the alpha channel of the transformed substitution template before using it to alpha blend the foreground to the background. We use the original foreground's bounding box coordinates for accurate object placement. The Gaussian blur smoothens the edges of the substitution foreground resulting in a more natural substitution.

$$\text{output\_img} = \alpha * \text{foreground} + (1 - \alpha) * \text{background}$$

Image Harmonization: RainNet

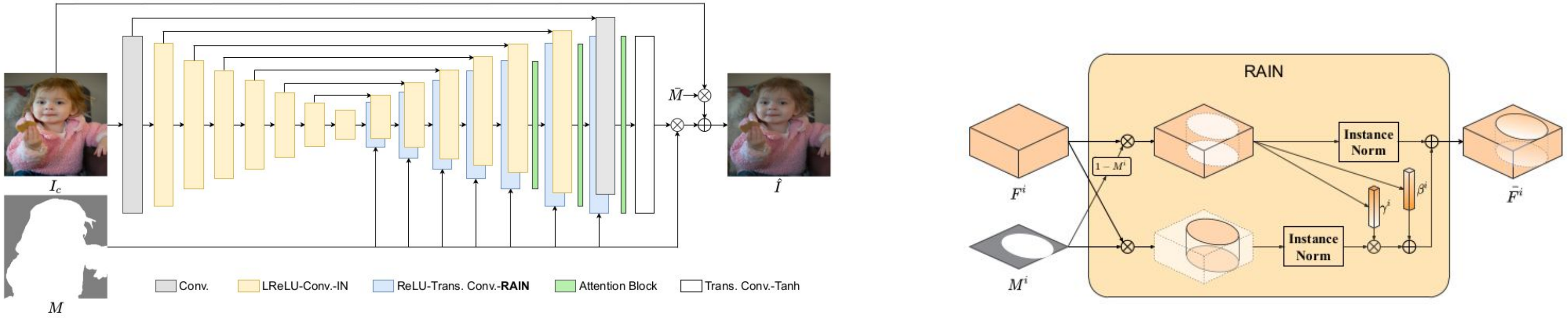


Fig 12: RainNet generator architecture and Region-aware Adaptive Instance Normalization (RAIN) module as proposed in paper [10]

RainNet [10] treats image harmonization as a style transfer task and applies style information extracted from background to foregrounds.
It consists of a learnable generator G which is a U-Net like model that takes as input the composite image I_c and the foreground mask M and predicts the harmonized image $I' = G(I_c, M)$ by optimizing $// I' - I //_1$ so that the harmonized image remains close to the groundtruth image I. The generator G has Region-aware Adaptive Instance Normalization (RAIN) modules in the decoder

[10] Ling, J., Xue, H., Song, L., Xie, R., & Gu, X. (2021). Region-aware adaptive instance normalization for image harmonization. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 9361-9370).

Instance Segmentation: Mask R-CNN

Split	mAP@0.5	mAP@0.75	mAP@0.50:0.95
Test org bbox	0.836	0.819	0.732
Test subst bbox	0.709	0.684	0.556
Test org segm	0.834	0.810	0.743
Test subst segm	0.709	0.701	0.636

Table 4: Mean Average Precision (mAP) of trained Mask R-CNN across different IoU ranges, on original and substituted test images of DFG dataset

In order to get a quantitative overview of the substitution performance, we train a Mask R-CNN [11] on original training set images and test the model on the test set original images and after substitution. We only consider 116/200 categories for training and testing, that we've considered for substitution. We finetune a ResNet50 backend Mask R-CNN pretrained on MS-COCO

[11] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

Final Results: Good

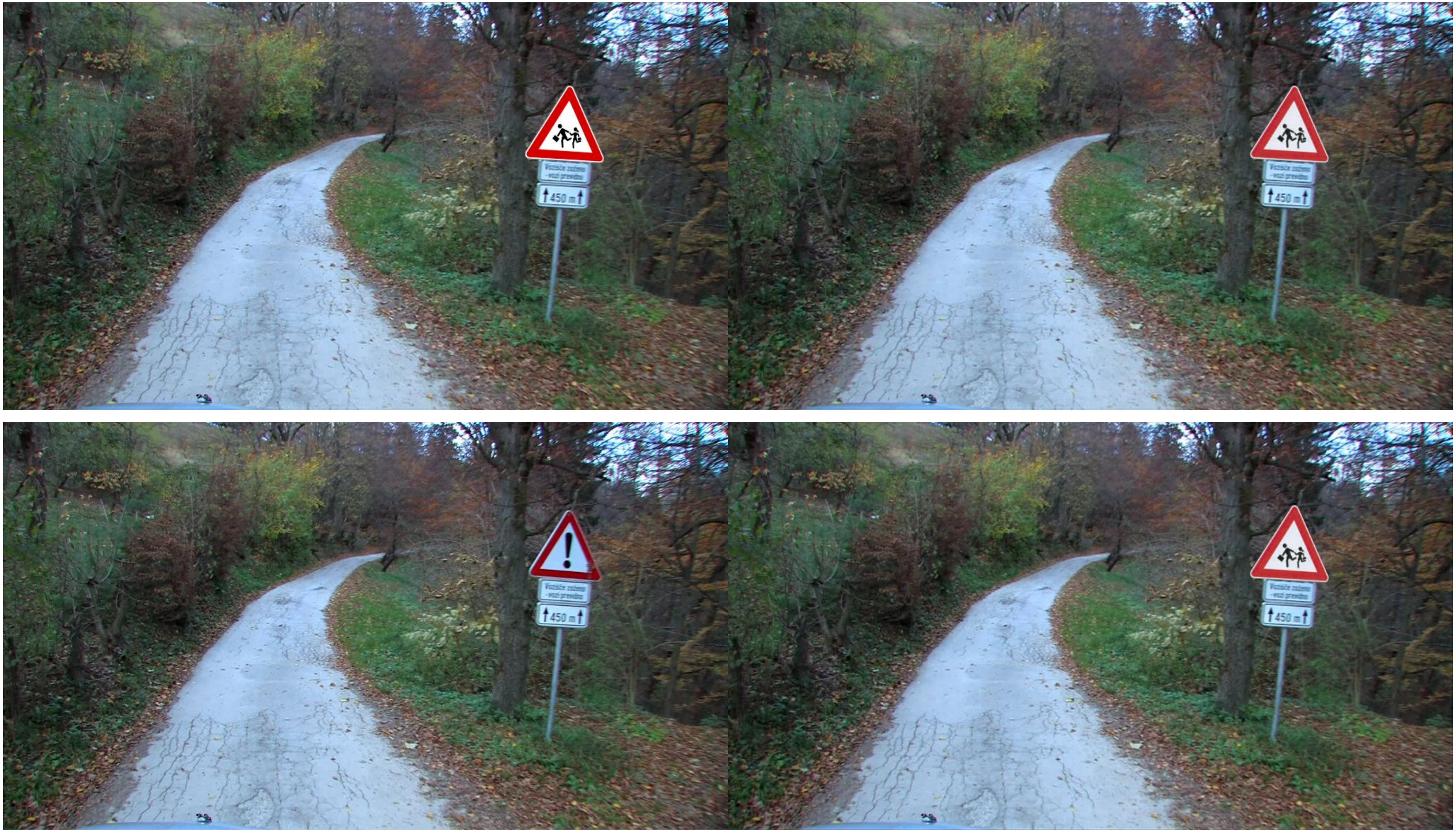


Fig 13: Top: Before and after RainNet harmonization. Bottom: Original image vs substitution result

Final Results: Good

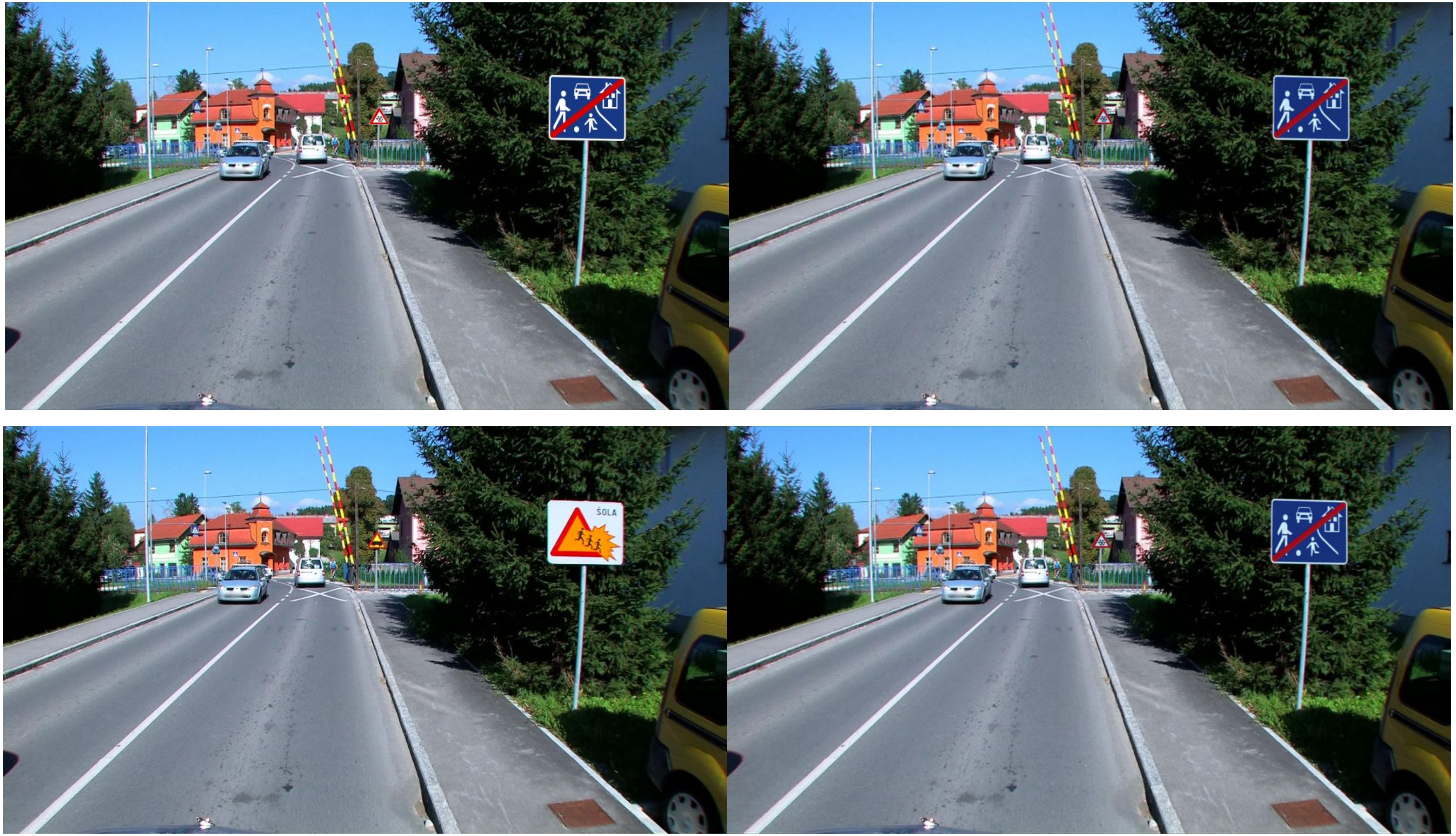


Fig 14: Top: Before and after RainNet harmonization. Bottom: Original image vs substitution result

Final Results: Poor Harmonization

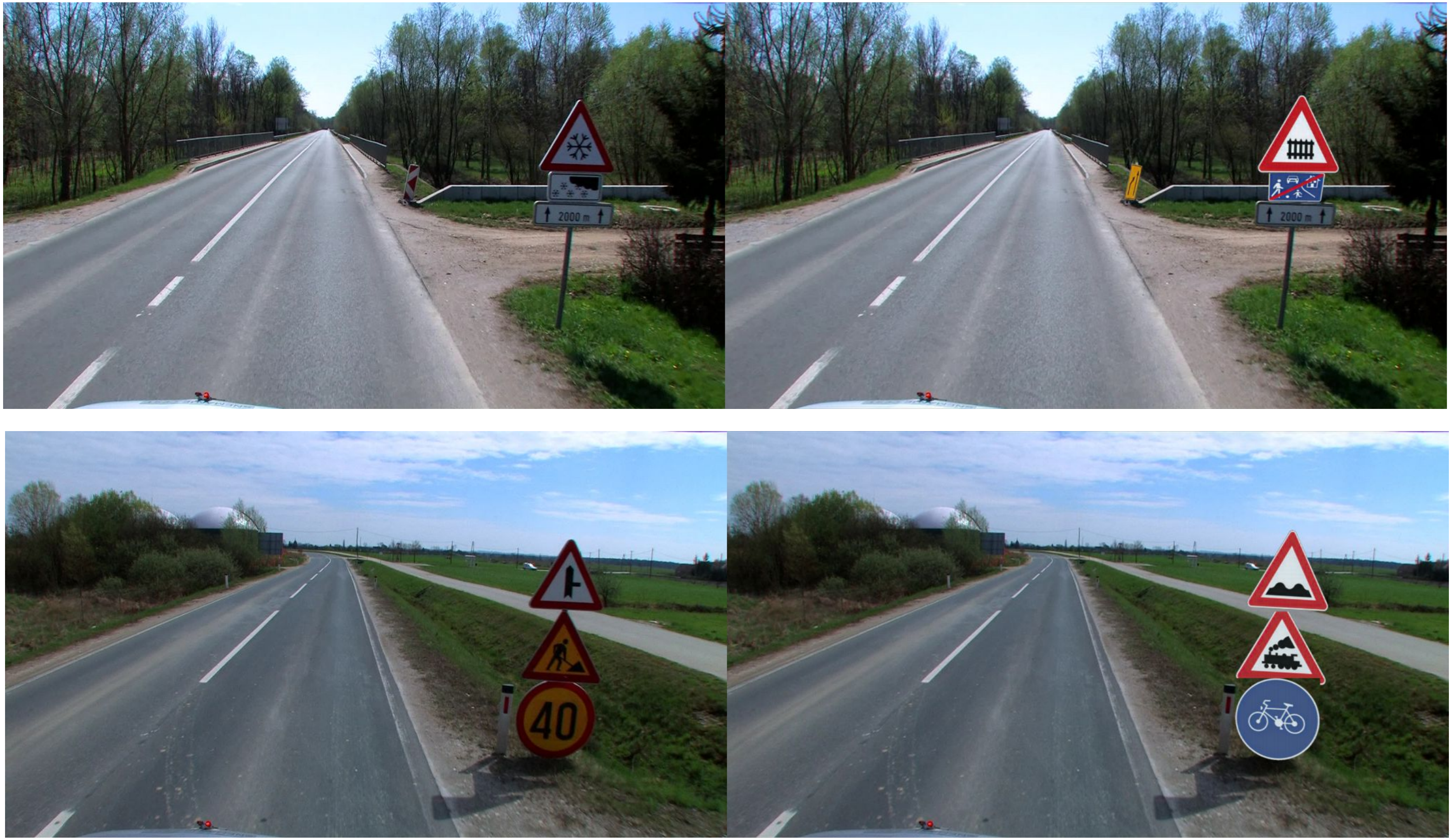


Fig 15: Two samples showing inadequate harmonization. Left: original image, right: final substitution result after harmonization

Final Results: Poor homography estimation

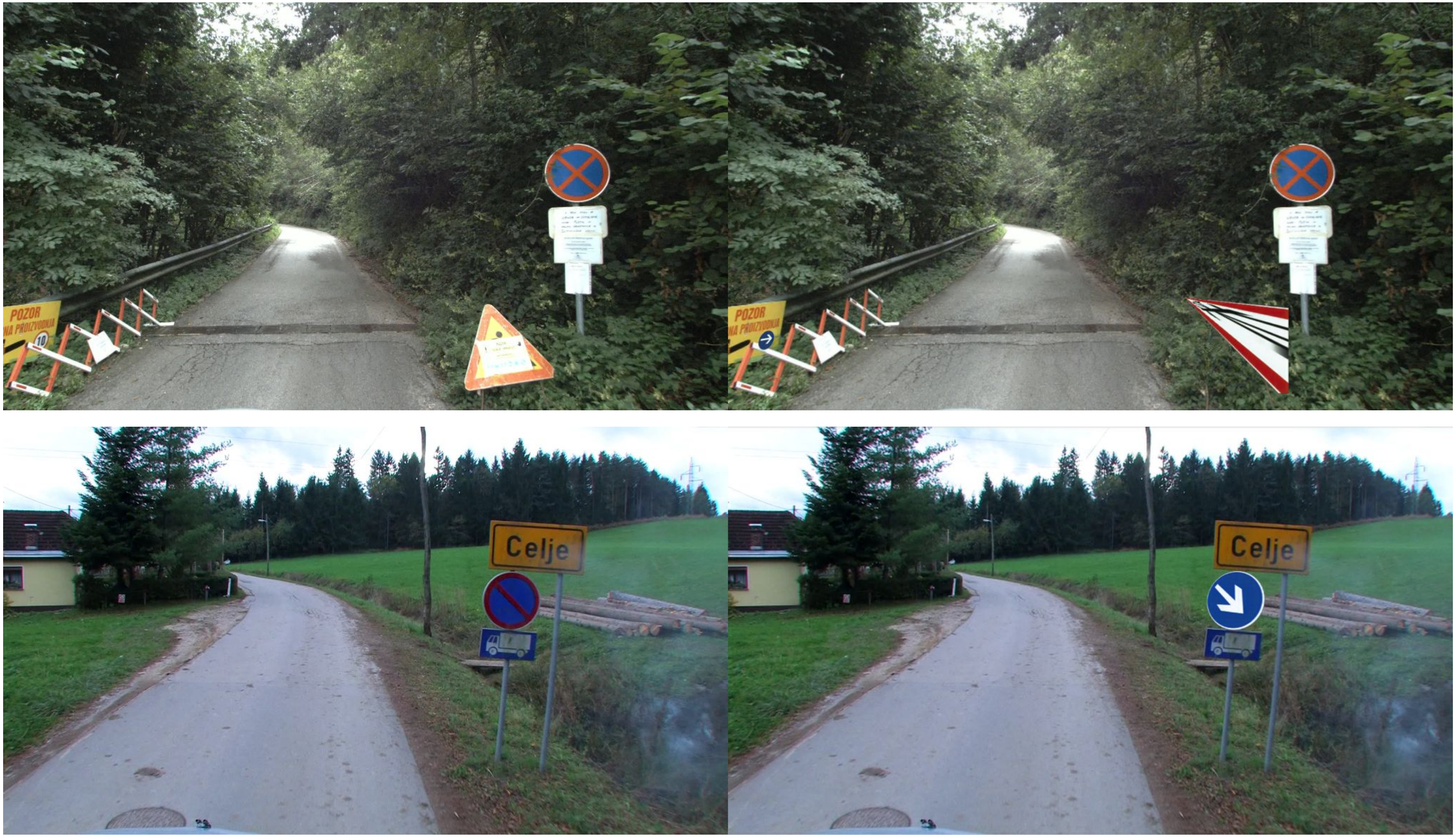


Fig 16: Two samples showing bad homography estimation. Left: original image, right: final substitution result after harmonization

Thank You!