		NYU		Garage (Pseudo)	
		BN	GN	BN	GN
Ratio target : NYU	1 : 1	84.9	**87.2**	**87.7**	86.5
	3 : 1	77.8	**81.1**	90.6	**91.7**
	4 : 1	76.4	**79.8**	92.0	**92.4**
	10 : 1	70.3	**73.4**	93.6	**94.7**
	20 : 1	66.7	**67.5**	94.5	**95.3**
	200 : 1	**54.6**	53.9	95.3	**96.1**
Fraction replay NYU	10%	67.6	**68.3**	94.0	**95.4**
	5%	63.6	**65.0**	94.9	**95.9**
	0% (fine-tuning)	**37.3**	36.4	95.5	**96.3**

Table 4: Comparison of segmentation quality [% mIoU] on NYU→`Garage` between models trained with batch normalization (BN) and models trained with group normalization (GN), under different replay regimes.

A Appendix

Next to the content on the following pages, the supplementary material for this paper also consists of:

- summary video
- code supplement

A.1 Runtime

We conduct our experiments on 6-year-old hardware with a 8-core i7-6700K CPU and GeForce GTX 980 Ti GPU. While our implementations are not heavily optimised for runtime, we carefully select a fast rather than precise neural network architecture. Accordingly, the segmentation of all three camera images takes 127 ± 23ms. The following ICP localisation takes 529 ± 132 ms on our hardware (CPU only). Given the LiDAR frequency of 5 Hz (or 200 ms per scan), the total delay from the beginning of the scan to the localised pose is approximately 856 ms. This requires a factor 5 optimisation for real-time deployment. After localisation, our pseudolabel generation takes 1.327 ± 0.127 s, most of which is taken by the superpixel segmentation. However, this process is not time-critical since we only produce pseudolabels from a subset of all frames.

A.2 Details on the Segmentation Training

In all our experiments we use a batch size of 10 and train the network for up to 100 epochs, using early stopping with a patience of 20 epochs based on the validation loss. We set the learning rate to 10^{-4} for the pre-training on NYU and to 10^{-5} for the remaining experiments, and adaptively decrease it when the validation loss reaches a plateau. We optimize the cross-entropy loss on the binary foreground-background labels. Our network architecture, based on Fast-SCNN [61], has a total of 1,775,110 trainable parameters. We use group normalization [66] in all layers; we conducted a preliminary ablation study (cf. Table 4) comparing this design choice with the alternative batch normalization [67]. In accordance with [66], we found group normalization to be more indicated for our transfer-learning tasks, in which the statistics of the *source* training data, used by batch normalization to fit per-layer parameters [67], do not match in general those of the *target* domain. This is reflected in the models trained with group normalization performing consistently better or comparably to those trained with batch normalization, as soon as a non-negligible amount of replay is used.

A.3 Details on Cross-Domain Forgetting

We present a detailed analysis of forgetting in terms of segmentation in Table 5 as supplementary information to the main results presented in Table 2. With no exception, memory replay performs better on source environments than finetuning. We note that the effect of forgetting is even stronger on the NYU data than in the deployment environments.

For deployment into 4 subsequent domains, we present additional results to the two listed in Table 2 in Table 6. The results for this 'stage 3' deployment show that the system scales well also to 4 consecutive environments. Interestingly, there is rarely any forgetting measurable in the localisation results in the Garage, and also in the segmentation quality forgetting is minor. We offer the explanation

Stage	Source → target	Segmentation quality [% mIoU]													
		NYU		Garage				Construction				Office			
		GT		Pseudo		GT		Pseudo		GT		Pseudo		GT	
		RB	FT	RB	FT	RB	FT	RB	FT	RB	FT	RB	FT	RB	FT
0	Pretraining on NYU	–	86.4	–	(22.5)	–	(33.9)	–	(22.7)	–	(27.6)	–	(39.6)	–	(46.5)
1	NYU → Garage	**68.3**	36.4	95.4	96.3	62.8	61.8	–	–	–	–	–	–	–	–
1	NYU → Construction	**78.6**	36.6	–	–	–	–	77.0	79.5	48.2	48.9	–	–	–	–
1	NYU → Office	**81.0**	66.2	–	–	–	–	–	–	–	–	69.7	70.9	53.9	51.2
2	NYU → Garage→ Construction	**70.3**	30.7	**91.8**	77.1	60.8	55.1	77.4	78.5	48.6	49.4	–	–	–	–
2	NYU → Garage→ Office	**70.9**	42.7	**92.8**	71.7	62.6	61.0	–	–	–	–	69.9	72.2	47.2	47.4
2	NYU → Construction→ Office	**78.6**	48.9	–	–	–	–	**71.3**	55.9	50.3	45.4	70.3	72.2	47.6	49.4
2	NYU → Construction→ Garage	**70.5**	36.7	94.4	95.6	62.2	62.0	**61.4**	43.3	49.3	42.3	–	–	–	–
2	NYU → Office → Garage	**68.7**	36.4	95.3	96.4	62.1	61.0	–	–	–	–	**61.2**	46.9	47.8	40.2
2	NYU → Office → Construction	**77.7**	38.8	–	–	–	–	73.1	73.0	49.9	49.1	**63.4**	44.7	47.5	33.3
3	NYU → Garage → Construction → Office	**70.9**	42.4	**91.5**	60.4	62.4	56.9	**72.1**	52.3	49.9	46.1	67.4	72.6	46.6	45.9
3	NYU → Garage → Office → Construction	**71.4**	33.0	**91.7**	71.2	62.7	53.1	75.5	79.1	49.3	48.9	**64.6**	43.6	41.6	33.2
3	NYU → Construction → Office → Garage	**69.4**	35.0	96.3	97.2	61.1	60.6	**60.6**	44.2	47.2	42.5	**61.2**	45.6	43.7	36.3
3	NYU → Construction → Garage → Office	**72.0**	39.9	**91.8**	74.6	64.7	62.2	**64.1**	39.4	50.4	37.2	68.9	71.5	45.2	46.3
3	NYU → Office → Garage → Construction	**71.2**	32.8	**89.9**	77.9	63.2	57.2	82.0	80.2	50.3	48.2	**62.7**	41.7	43.9	33.8
3	NYU → Office → Construction → Garage	**69.2**	35.0	95.9	96.9	61.7	61.6	**60.3**	45.6	47.5	40.7	**62.3**	45.6	42.3	37.6

Table 5: Evaluation of forgetting and knowledge transfer when deploying into multiple environments. The perception system is subsequently trained on different environment and at every step evaluated on all seen environments. Bold shows how the replay buffer (RB) prevents degradation of performance on the datasets on which the model has previously been trained, as opposed to simple fine-tuning (FT).

		mean/median/std translation error [mm]		
environment sequence	method	Office	Construction	Garage
NYU → Garage → Construction → Office	replay	155 / 123 / 112	100 / 71 / 90	39 / 30 / 29
	finetuning	217 / 130 / 283	167 / 80 / 270	41 / 31 / 36
NYU → Garage → Office → Construction	replay	157 / 124 / 110	104 / 71 / 97	40 / 31 / 30
	finetuning	190 / 117 / 254	98 / 71 / 86	43 / 37 / 29
NYU → Construction → Office → Garage	replay	176 / 137 / 123	116 / 72 / 116	39 / 31 / 29
	finetuning	194 / 171 / 112	104 / 74 / 87	40 / 31 / 31
NYU → Construction → Garage → Office	replay	167 / 129 / 113	105 / 72 / 92	39 / 30 / 29
	finetuning	145 / 114 / 130	385 / 95 / 868*	41 / 32 / 32
NYU → Office → Garage → Construction	replay	157 / 132 / 102	105 / 70 / 100	41 / 31 / 32
	finetuning	158 / 145 / 85	112 / 82 / 92	43 / 35 / 30
NYU → Office → Construction → Garage	replay	170 / 142 / 114	114 / 72 / 114	42 / 32 / 32
	finetuning	185 / 155 / 107	131 / 74 / 129	42 / 34 / 31

Table 6: Localisation results for the stage-3 deployments through all environments. For the segmentation quality, see Table 5.

that the garage is similar enough to both other environments such that even when training on another environment, most of the knowledge about the garage can be kept.

A.4 Details on the Continual-Learning Ablation Study

For both distillation and EWC, we use the same learning parameters as the experiments with replay buffers. In the following, we denote with $\mathbf{X}$ and $\mathbf{M}$ respectively an image and the corresponding mask from the training dataset $\mathcal{D}$. When $\mathbf{X}$ is a pseudo-label image, a pixel in $\mathbf{M}$ is *masked* if the corresponding pixel in $\mathbf{X}$ has an associated pseudo-label (background/foreground) and *not masked* if the corresponding pixel has unknown label; if $\mathbf{X}$ is an image replayed from NYU, all pixels in $\mathbf{X}$ are masked. For a given stage-1 experiment (i.e., in which we deploy the model pretrained on NYU in a new environment, cf., e.g., Tab. 5), we denote the output prediction of the model pretrained on NYU as $\mathbf{y}_0(\mathbf{X})$ and the output prediction of the current stage-1 model as $\mathbf{y}(\mathbf{X})$; to indicate the predicted score associated to each class $c \in \{b, f\}$ (b = background, f = foreground) we write $\mathbf{y}_0(\mathbf{X})[c]$ and $\mathbf{y}(\mathbf{X})[c]$. Finally, we denote with $M(\mathbf{X}, \mathbf{M})$ a function that maps an input image $\mathbf{X}$ and its corresponding mask $\mathbf{M}$ to a vectorized version of $\mathbf{X}$ that contains only the pixels that are masked in $\mathbf{M}$.

The generic distillation loss reads as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \mathcal{L}_d, \tag{1}$$

where λ is a hyper-parameter and $\mathcal{L}_{ce}$ is the cross-entropy loss (cf. Sec. 4.4).

ICP parameters			mean/median/std translation error [mm]			
			Construction		Office	
β [rad]	#NN	DOF	no segmentation	self-improving	no segmentation	self-improving
1.5	*10*	*6*	488 / 183 / 999*	150 / 138 / 81	169 / 164 / 86	162 / 158 / 78
1.5	20	6	81 / 63 / 66	94 / 68 / 82		
1.2	10	6	1547 / 649 / 1746*	2413 / 719 / 2923*		
1.5	10	4	112 / 82 / 86	116 / 76 / 108		
1.2	20	6	182 / 191 / 100	164 / 142 / 152	190 / 172 / 98	173 / 150 / 95
0.8	30	6			190 / 177 / 96	163 / 142 / 97
1.0	30	4			182 / 177 / 92	154 / 141 / 81
0.8	20	4			202 / 190 / 95	166 / 146 / 97
0.8	*30*	*4*	102 / 82 / 72	105 / 74 / 91	167 / 168 / 88	158 / 135 / 98

Table 7: Ablation of the change of ICP parameters between default (top italic) and values initially used in office experiments (bottom italic). β is the maximum allowed angle between the normal directions of a point in the scan and the associated point in the map. #NN is the number of nearest neighbors used to estimate that normal direction in the scan. DOF is the number of degrees of freedom in which to perform localisation, where 4DOF disables pitch and roll. We analyse both slight and grave changes in parameters and find that (i) our self-improving approach is better than the baseline for most parameter combinations, and (ii) given the runtime increase from top to bottom, $\beta = 1.5rad$ with 6DOF and 10NN is a feasible parameter choice.

For output distillation, the regularization loss $\mathcal{L}_d$ is a cross-entropy loss between the prediction of the previous and the current model, masked by the input mask of each image, i.e.,

$$\mathcal{L}_d = - \sum_{(\mathbf{X},\mathbf{M})\in\mathcal{D}} \sum_{c\in\{b,f\}} \frac{M(\mathbf{y}_0(\mathbf{X}),\mathbf{M})[c] \cdot \log(M(\mathbf{y}(\mathbf{X}),\mathbf{M}))[c]}{|\mathcal{D}|}. \tag{2}$$

For feature distillation, similarly to [37] we consider the features outputted by the network at a selected layer and minimize the ℓ_2 norm between these as returned by the pre-trained model and by the current model. In particular, we consider the layer that precedes the final classification module in the Fast-SCNN architecture [61] and denote its output as $\mathbf{l}_0(\mathbf{X})$ and $\mathbf{l}(\mathbf{X})$, respectively for the pre-trained and for the current model. The regularization loss can therefore be expressed as:

$$\mathcal{L}_d = \frac{\|\mathbf{l}_0(\mathbf{X}) - \mathbf{l}(\mathbf{X})\|_2^2}{|\mathcal{D}|}. \tag{3}$$

For Elastic Weight Consolidation (EWC), we adopt the original loss introduced in [29], which is of the form:

$$\mathcal{L} = \mathcal{L}_{\text{main}} + \lambda \sum_i F_i(\theta_i - \theta_{i,0})^2, \tag{4}$$

where the sum is computed over the trainable parameters θ_i and $\theta_{i,0}$ respectively of the current and of the pre-trained model, and F_i is the element on the diagonal of the Fisher information matrix associated with the i-th parameters. $\mathcal{L}_{\text{main}}$ represents the main loss optimized in the given task, which in our case is the background-foreground cross-entropy loss $\mathcal{L}_{ce}$.

A.5 Localisation Parameters

In general, we run point-to-plane ICP with 3 nearest neighbors and initialise on the previously solved pose. We apply multiple filters to the input scan, even after the semantic filtering:

- We require the scan to have at minimum 500 points (i.e., rejecting scans where the segmentation classifies nearly everything as foreground).
- We subsample the scan to a maximum density of 10,000 pts/m^3.
- After nearest neighbor association, we reject the 20% points that are further away from the map.
- We reject associations where the estimated surface normals (estimated based on the 10 nearest neighbors) have a larger angle deviation than 1.5 rad.

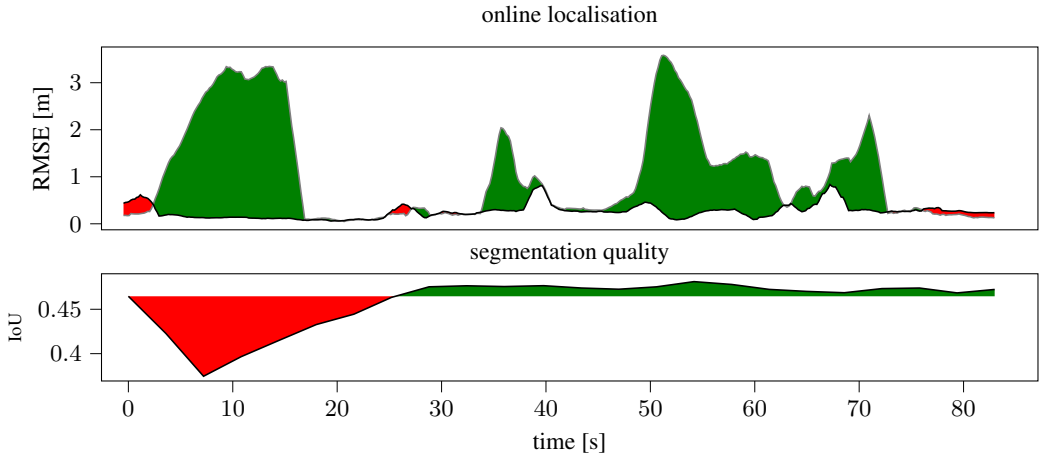


Figure 5: Online Learning in the office.

For initial experiments, in order to localise without segmentation and generate pseudolabels in the very cluttered office environment, we enforced additional filters:

- We only localised in 4 degrees of freedom (x, y, z, yaw).
- We estimated normal directions based on 30 nearest neighbors and only associated points to the map if the angle between the normals is below 0.8 rad.

These additions were used in Table 1^+ and for generating the office pseuodlabels. However, our ablation study from Table 7 shows that this is not necessary. Our final system is sufficiently robust to the choice of localisation parameters and can improve over the baseline for most choices of parameters.

A.6 Pseudolabel Parameters

We empirically set the distance threshold to $\delta = 0.1$m and discard superpixels with a depth variance that surpasses 0.5m. We smooth the images with a Gaussian kernel ($\sigma = 0.2$) and oversegment them into approximately 400 superpixels with SLIC parameter $\texttt{compactness} = 10$[5]. On the data captured from the garage, we use a different superpixel algorithm (SCALP [68]) that we later discard because of long runtimes. We do not notice qualitative differences between the created superpixels. In the office environment, we increase the standard deviation threshold to 1m due to large amounts of clutter.

To get an estimate of the quality of the pseudolabels themselves, we match frames where we have both manual ground-truth annotations and pseudolabels. Unfortunately, we could not recover pseudolabels for the images that were used to generate ground-truth in the office environment. When evaluating the pseudolabels, we also ignore all pixels that are not labelled (due to high variance or no reprojected LiDAR points in that superpixel). Therefore, the evaluation is strongly biased in favor of the pseudolabels. We measure 68.4% mIoU on the garage pseudolabels. For the same pixels (only those where pseudolabels are not ignored), our trained models get 64.3% mIoU. In the construction site environment, we measure 49.5% mIoU for the pseudolabels and 54.3% mIoU for our trained model.

A.7 Additional Online Learning Runs

Additional demonstrations of online learning are shown in Figures 5 and 6.

A.8 Example of segmentation predictions

Figures 7, 8, and 9 show examples of segmentation masks produced by the network on the source environment in the experiments with transfer from a first to a second environment. We report a

[5]This procedure is suggested by the skimage implementation that we use.

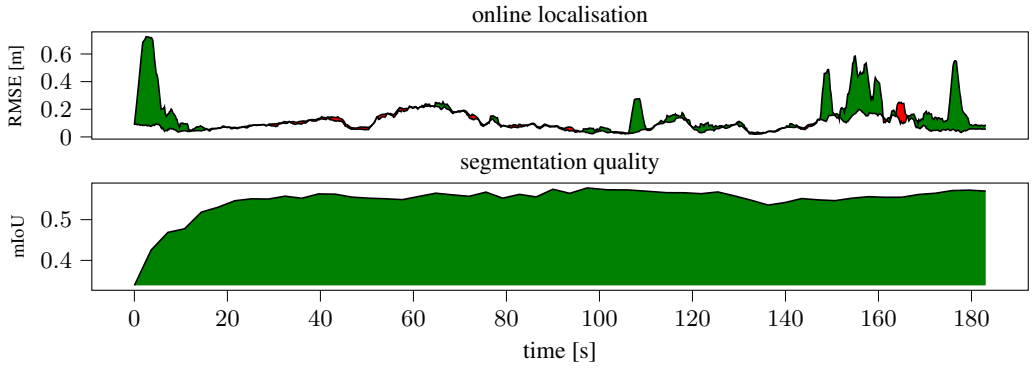


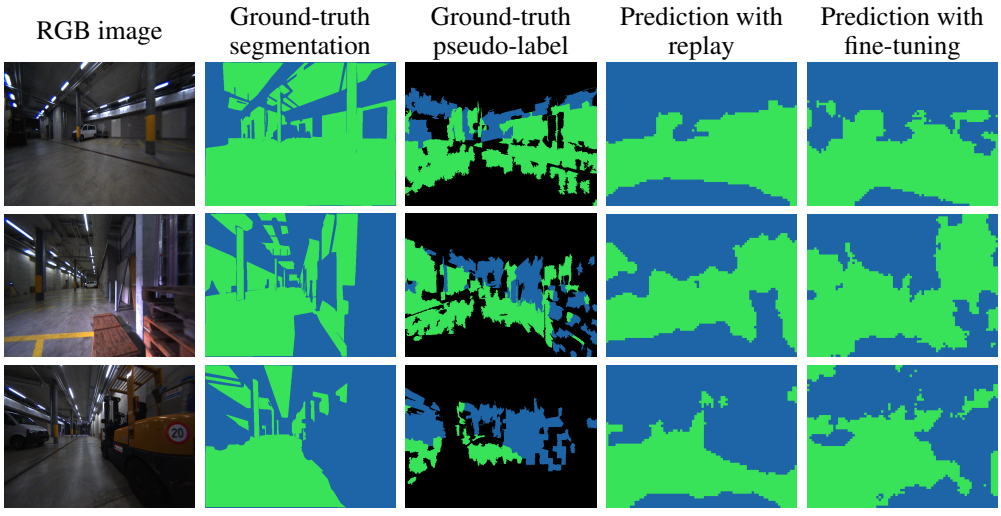
Figure 6: Online Learning in the garage.

Stage	Source → target	Segmentation quality [% mIoU]													
		NYU		Garage				Construction				Office			
		GT (no mask)		Pseudo		GT (no mask)		Pseudo		GT (no mask)		Pseudo		GT (no mask)	
		RB	FT	RB	FT	RB	FT	RB	FT	RB	FT	RB	FT	RB	FT
0	Pretraining on NYU	–	86.4	–	(22.5)	–	(40.3)	–	(22.7)	–	(29.4)	–	(39.6)	–	(46.3)
1	NYU → Garage	**68.3**	36.4	95.4	96.3	44.5	43.3	–	–	–	–	–	–	–	–
1	NYU → Construction	**78.6**	36.6	–	–	–	–	77.0	79.5	32.7	32.7	–	–	–	–
1	NYU → Office	**81.0**	66.2	–	–	–	–	–	–	–	–	69.7	70.9	53.2	51.7
2	Garage→ Construction	**70.3**	30.7	**91.8**	77.1	43.8	46.0	77.4	78.5	34.7	34.6	–	–	–	–
2	Garage→ Office	**70.9**	42.7	**92.8**	71.7	45.3	48.0	–	–	–	–	69.9	72.2	52.1	50.3
2	Construction→ Office	**78.6**	48.9	–	–	–	–	**71.3**	55.9	34.7	36.4	70.3	72.2	46.6	47.5
2	Construction→ Garage	**70.5**	36.7	94.4	95.6	43.7	44.2	**61.4**	43.3	33.1	31.0	–	–	–	–
2	Office → Garage	**68.7**	36.4	95.3	96.4	43.3	42.9	–	–	–	–	**61.2**	46.9	46.8	42.7
2	Office → Construction	**77.7**	38.8	–	–	–	–	73.1	73.0	34.1	33.7	**63.4**	44.7	46.6	36.7

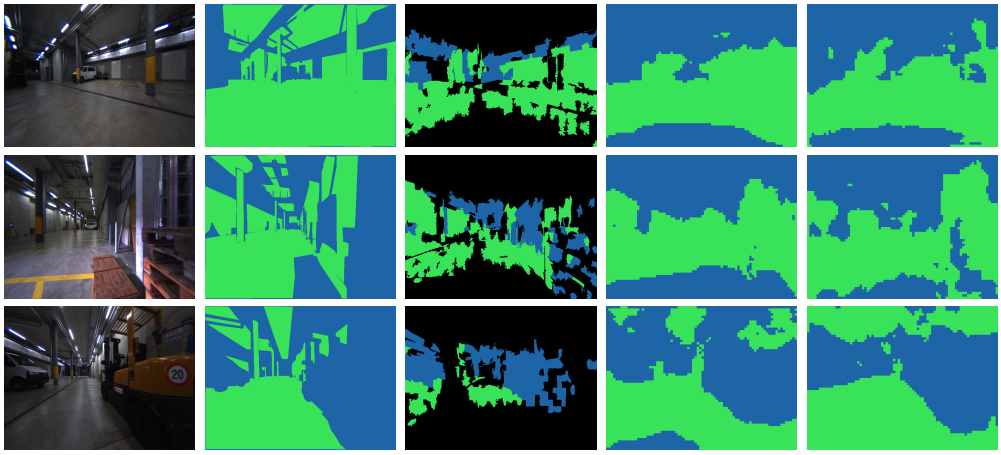
Table 8: While we in general evalaute segmentation quality only in the overlapping field of view of cameras and LiDAR, this table serves as a comparison as to how Table 5 would look when evaluating the whole camera images, including regions where the segmentation never has training signals because pseudolabels cannot be generated. We observe similar trends also in this table, while the results are more noisy.

selection of frames for which we have available ground-truth segmentation and show the predictions obtained both with a model trained with simple finetuning and with one trained with replay from the source and the pre-training datasets.

In the qualitative outputs, we observe that the models learn biases towards regions that are generally unlabeled at training time. In particular, areas in the upper and lower part of the image are commonly classified as foreground, and show a curvature that roughly reflects the regions in the training pseudo-labels where information is missing due to the reprojection of the LiDAR measurements into the camera view. This is in line with our discussion of the FoV mask, as supervision through pseudolabels is missing in those parts of the image; indeed, the learned biases in these unobserved regions often do not match the ground-truth class in these areas (cf., e.g., Fig. 7a, columns `Ground-truth segmentation` and `Prediction with replay`), and the evaluation would reflect this negatively if these areas were considered. We stress that the masked FoV region is most relevant for our application, as it represents the overlap of camera and LiDAR scans that we aim to filter and improve localization with. However, we also provide numbers when evaluating whole camera images instead of FoV masks in Table 8. As expected, the results outside of the LiDAR FoV are more noisy. From the qualitative examples and comparison with the FoV evaluation we know that this is due to wrong biases in image regions where no pseudolabels are available.

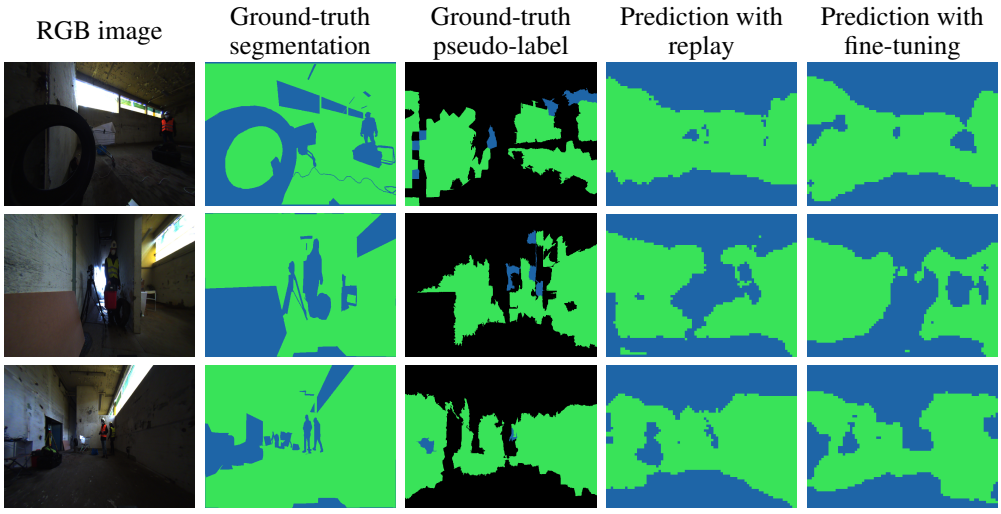


(a) `Garage`→`Construction`

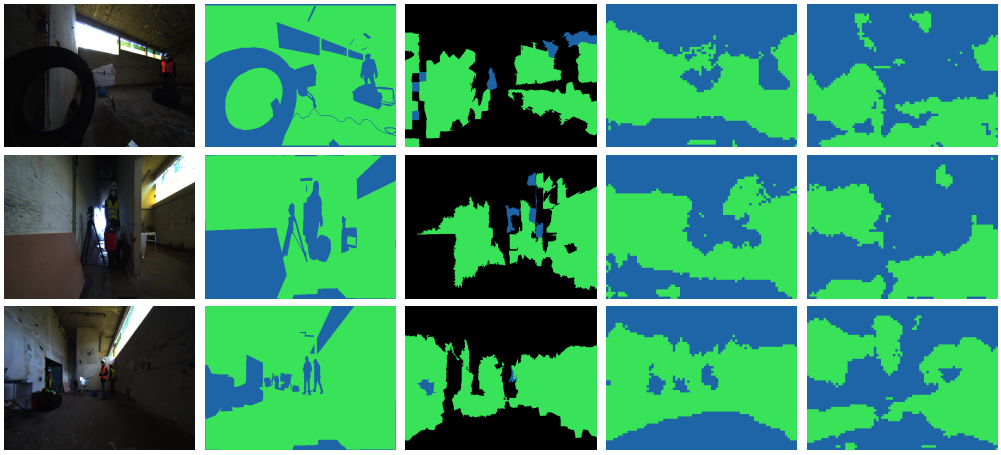


(b) `Garage`→`Office`

Figure 7: Illustrations of (prevention of) forgetting for the parking garage as source environment. Green is *background*, blue is *foreground* and black pseudolabels are ignored in training.

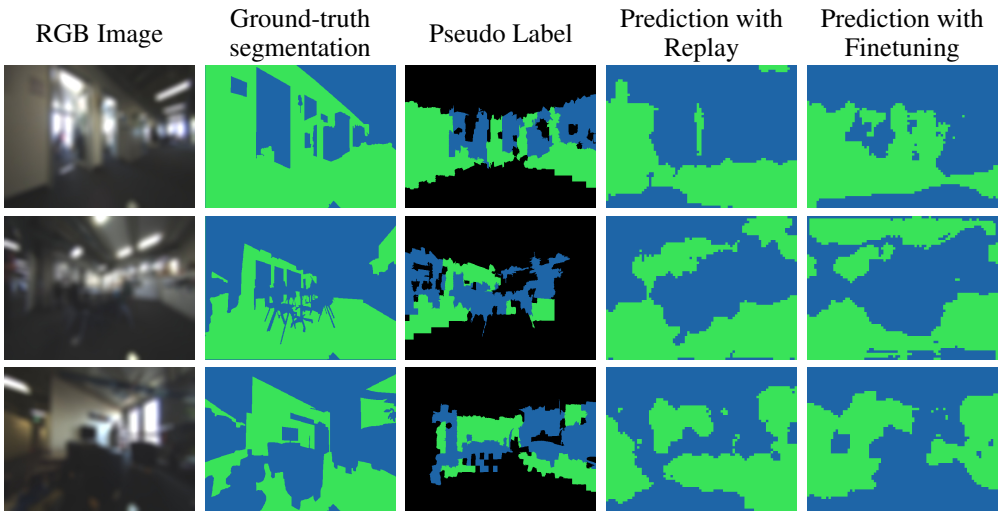


(a) `Construction`→`Garage`

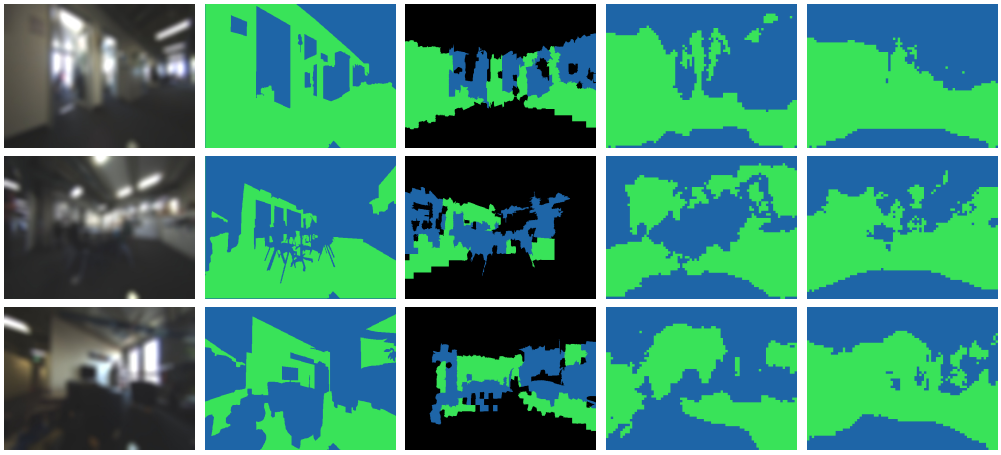


(b) `Construction`→`Office`

Figure 8: Illustrations of (prevention of) forgetting for the construction site as source environment. Green is *background*, blue is *foreground* and black pseudolabels are ignored in training.

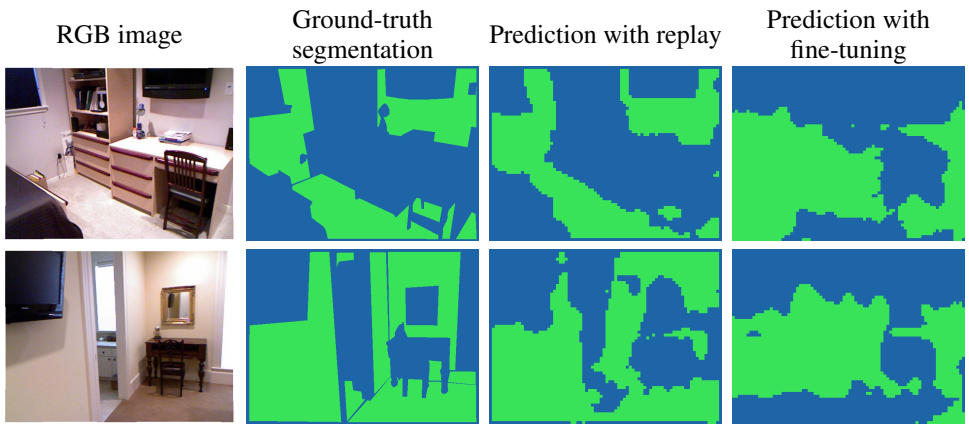


(a) `Office`→`Garage`

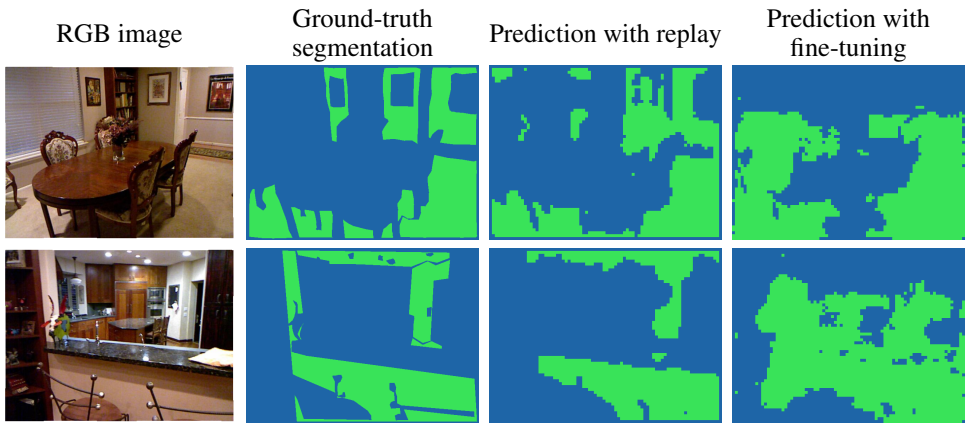


(b) `Office`→`Construction`

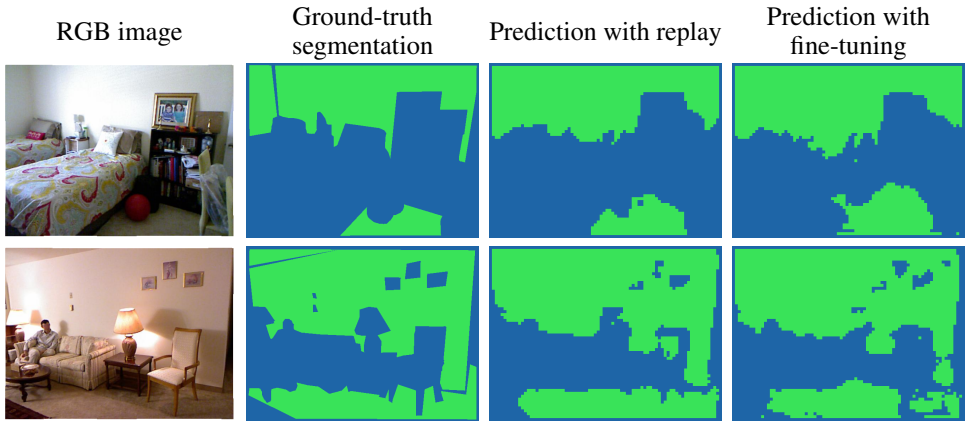
Figure 9: Illustrations of (prevention of) forgetting for the office as source environment. Green is *background*, blue is *foreground* and black pseudolabels are ignored in training. Images are blurred for anonymous submission.



(a) NYU→`Garage`



(b) NYU→`Construction`



(c) NYU→Office

Figure 10: Illustrations of (prevention of) forgetting for the NYU dataset as source environment. Green is *background*, blue is *foreground*.