Extensible Portal Frame Bridge Synthetic Dataset for Structural Semantic Segmentation

Tatiana Fountoukidou, Iuliia Tkachenko, Benjamin Poli, Serge Miguet

Extensible Portal Frame Bridge Synthetic Dataset for Structural Semantic Segmentation

Tatiana Fountoukidou[1*], Iuliia Tkachenko[1*], Benjamin Poli[2] and Serge Miguet[1*]

[1]LIRIS Laboratory, Université Lumière Lyon 2, 5 av. Pierre Mendès-France, Lyon, 69676, France.

[2]Diadès, Setec, 7 Chem. des Gorges de Cabriès, Vitrolles, 13127, France.

*Corresponding author(s). E-mail(s): tatiana.fountoukidou@gmail.com; iuliia.tkachenko@univ-lyon2.fr; serge.miguet@univ-lyon2.fr;
Contributing authors: benjamin.poli@setec.com;

Abstract

A number of bridges have collapsed around the world over the past years, with detrimental consequences on safety and traffic. To a large extend, such failures can be prevented by regular bridge inspections and maintenance, tasks that fall in the general category of structural health monitoring (SHM). Those procedures are time and labor consuming, which partly accounts for their neglect. Computer vision and artificial intelligence (AI) methods have the potential to ease this burden, by fully or partially automating bridge monitoring. A critical step in this automation is the identification of a bridge's structural components. In this work, we propose an extensible synthetic dataset for structural component semantic segmentation of portal frame bridges (**PFBridge**). We first create a 3 dimensional (3D) generic mesh representing the bridge geometry, while respecting a set of rules. The definition of new, or the extension of the existing rules can adjust the dataset to specific needs. We then add textures and other realistic elements to the model, and create an automatically annotated synthetic dataset. The synthetic dataset is used in order to train a deep semantic segmentation model to identify bridge components on bridge images. The amount of available real images is not sufficient to entirely train such a model, but is used to refined the model trained on the synthetic data. We evaluate the contribution of the dataset to semantic segmentation by training several segmentation models on almost 2000 synthetic images and then finetuning with 88 real images. The results show an increase of **28%** on the F1-score when the synthetic dataset is used. To demonstrate a potential use case, the model is integrated in a 3D point cloud capturing system, producing an annotated point cloud where each point is associated with a semantic category (structural component). Such point cloud can then be used in order to facilitate the generation of a bridge's digital twin.

Keywords: Structural Health Monitoring, Bridge Monitoring, Portal Frame Bridge, Deep Learning, Synthetic Dataset, Semantic Segmentation

1 Introduction

Recent advances in computer vision (CV) and image analysis have shown remarkable performances for numerous applications in different fields. With the emergence of large neural networks, or deep learning (DL), tasks that were long considered time-consuming and tedious are now automated. This is also the case for structural health monitoring (SHM), a multidisciplinary approach that aims, through observation and analysis over time, to evaluate the integrity of infrastructures such as bridges and buildings [1–3]. The automation of SHM tasks

has recently gained attention [4], and the past years several attempts have been made to leverage the benefits of CV in order to lessen the inconveniences of fully manual infrastructure inspections. Such inconveniences are not to be taken lightly, since often these evaluations demand long journeys, they are subject to the weather, and may come with significant safety risks. Automatic identification and localization of structural components or damage has therefore the potential to mitigate the aforementioned risks and difficulties. However, the need of expert annotations remains a limiting factor, as the former constitute a prerequisite for powerful machine learning (ML) models to perform well.

In this work we focus on the task of bridge inspection. Structures such as bridges are heavily subject to wear, as they are exposed to environmental elements, and receive a lot of stress and tension while used. It goes without saying that insufficiently maintained bridges can be a cause for serious and possibly fatal road accidents. For this reason, regular inspections are necessary in order to assess and follow up on their condition, and decide if maintenance is needed, and how urgently. Typically, a bridge inspection should be carried out annually, and a more detailed technical inspection takes place every three years. Traditionally, bridge inspections are performed by qualified engineers, who visually assess deficiencies located on the different bridge structural elements. The identification, localization, and classification of such deficiencies play a role in the final evaluation of the bridge's condition. This is a challenging task [5], that depends on the bridge type. In this work we target portal frame bridges. Those are concrete monolithic structures, in the sense that their decks and abutments form one block, a property that makes them very solid and robust [6]. They constitute a large number of medium to small bridges found in urban and rural settings. A schematic of a portal frame bridge, along with its main structural components can be seen in Fig. 1.

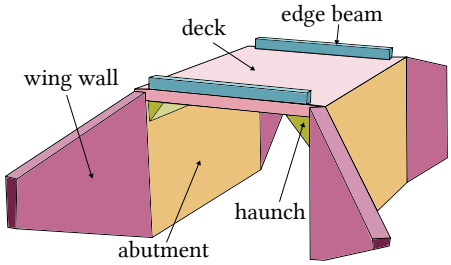

Fig. 1: Portal frame bridge schematic with structural components.

1.1 Semantic segmentation as part of automated bridge inspection

The identification of a bridge's structural elements can serve various SHM tasks. The components can be presented to the user during inspection as a visual aid, or they can be used to label a 3D point cloud, that would then be used to create a numerical geometrical representation of the bridge structure, called digital twin. A digital twin can then be used for building information modelling (BIM). In more details, a digital twin serves to digitally conserve the information of subsequent bridge inspections, and can also be used as additional visual aid during inspection, if coupled with augmented reality (AR) [7, 8]. The masks produced from semantic segmentation of structural components can also serve as a prior in a hierarchical damage identification method, as the identification of damage needs to be combined with the structural component the damage appears on, in order to effectively rate the bridge's condition according to most rating methods [9, 10]. Certain damage types only appear in specific components (e.g. a water trace is not considered damage if seen on the wing walls, since it can be a result of rain, but is noted as damage if it appears on the inside of the bridge). The severity of some pathologies is a combination of the pathology characterization and its localization, e.g. a longitudinal or transversal crack is severe if observed on the deck, while a horizontal or vertical crack is severe if observed on the abutment. Finally, a detected pathology can appear on a non-bridge part (e.g. on the road), and thus be irrelevant to the bridge inspection.

Semantic segmentation can either be done on bridge images, or on bridge point clouds. In this work, we focus on semantic segmentation performed on images, for three main reasons.

1. Point clouds are high dimensional data, whose annotation is very cumbersome. Even when using synthetic data, a large amount of samples is needed to train powerful artificial intelligence (AI) algorithms.
2. Inference time for a point cloud is significantly larger than that of an image, due to the higher dimensionality of the input data and the model complexity. Part of our goal being to produce an annotated point cloud

"on the go" during bridge inspection, using only a mobile device, inferring on the point cloud was both time and resource prohibiting. Inferring on images (or video frames) and using this model to automatically annotate the point cloud overcomes this obstacle.
3. As mentioned above, a model that segments images can be more versatile to be integrated in other parts of SHM, not involving point clouds.

The motivation for the generation of a synthetic dataset stems from the fact that annotating a sufficient amount of real images is cumbersome and time consuming for domain experts. The annotation needs to be repeated for a significant amount of new real images if another bridge type or other structural categories are to be integrated in an automated system, while with a parametrizable automatically annotated synthetic dataset, a limited number of changes in parameters can produce a whole new dataset. Additionally the 3D model that is used to generate those synthetic images can also find use in other steps of SHM, such as the generation of the digital twin of a bridge.

1.2 Related Works

Significant attempts have been made in order to use CV to assist bridge inspections, with various levels of specialized equipment needed ([11, 12]). A number of studies focus on the semantic segmentation of bridge components applied directly on point clouds ([13–16]). These approaches rely on the very tedious task of point cloud labeling, or on synthetic data, and perform the point cloud segmentation after the point cloud has been captured. In [13], for example the authors tackle semantic segmentation of bridge components directly on a 3D point cloud. With the use of specialized equipment, they capture point clouds of a number of bridges, which they manually annotate before training a graph neural network (GNN) for semantic segmentation. In [16] the authors also study the use of synthetic data for bridge monitoring, but from a different perspective than our work. They manually construct 3D models of 27 generic bridges, that they then use to capture annotated synthetic point clouds. They use a combination of synthetic and real bridge point clouds of highway bridges, captured through a terrestrial mobile LiDAR[1]. They train a model to annotate point clouds after they are captured. Such annotated point clouds are then used to create a parametric model of the bridge. Our work differs from this study, as it explores a different type of bridges, and a different modality of synthetic data (images instead of point clouds). While, as we show, an image segmentation model can be used to annotate point clouds, this is not its only possible use, nor it is the main contribution of our work. Finally, we automatically and randomly generate a vast number of 3D models, as opposed to the 27 manually crafted models of [16].

In this work we are mostly interested in inferring the semantic information of bridge components from images. To that end, [17] provides a review of studies for monitoring civil engineering structures using visual information provided from unmanned aerial vehicle (UAV)s. This kind of information is interesting since it can apply to structures that are hard to approach, however it is relatively costly, and it provides a very different point of view of the structures, compared to the 'inspector' views we tackle in this work. The authors of [18] explore the use of transfer learning for structural component segmentation. They work on broader civil infrastructure categories, not specifically bridges, and the pretraining is done on a general image dataset and not a task specific one. [19] focuses on the hyperparameter selection for object detection networks in disaster inspection scenarios. As far as bridges are concerned, they only detect bridge columns. In [20] the authors use CV and DL to classify the bridge type and segment bridge components, using images taken from a UAV. For training, they use photos sampled from the internet.

Since the most powerful CV methods strongly depend on large and detailed datasets, research interest is directed in their construction ([21–24]). An extensive study of such datasets is beyond the scope of this work, so in this section we will present some of those datasets, that are either synthetic or related to bridges, or both. SYNTHIA ([24]) is a remarkable synthetic dataset consisting of more than 200,000 synthetic urban scene images of a virtual city, and was created to serve the development of autonomous driving applications. Another synthetic dataset, this time of images of construction sites, is presented in [25]. The authors use a popular video game from which they extract interesting snapshots. COCO-bridge ([22]) is a thorough attempt to

[1] https://www.ibm.com/topics/lidar

create an annotated dataset of several structural details of bridges (e.g. cover plate termination, bearing), that can be useful during inspections. Bounding boxes are provided for each detail of interest. Often bridge inspections might include videos of the bridge, which motivates the authors of [26] to create a synthetic video simulation dataset of bridges, and train a recurrent neural network (RNN) that leverages information of previous frames in order to segment the elements of a given frame. In [27] a dataset of bridge images selected from google street view and imagenet ([28]) are manually annotated by the authors. A hierarchical scheme with 2 steps, where a broader 'structure segmentation network' precedes a bridge component segmentation network is then trained to identify broad bridge component categories. Closely related to our work, Narazaki et al. [23] introduce the Tokaido dataset, a synthetic dataset of Japanese high-speed railway viaducts, which they use to train a semantic segmentation convolutional neural network (CNN). While they define a small number of parameters (less than 10) that are relevant for viaducts, we define a scheme with around a hundred parameters that are sampled in order to create a great variability of portal frame bridges, by individually creating building blocks of each semantic category. Such approach, though more complicated, can prove more flexible and extensible, and allow for greater variability in the produced structures.

1.3 Contributions

We propose a pipeline that allows the generation of a synthetic dataset for portal frame bridge component semantic segmentation. We first define a number of dimensions and constraints that define a parametric model, allowing us to generate an annotated geometric 3D model of a portal frame bridge. Such parametrization can serve a double purpose. On the one hand, by randomly sampling in the acceptable range of parameters, we can produce infinite instances of valid 3D meshes, which we then enhance with texture and environmental information to create synthetic images with their corresponding ground truth masks. An overview of this procedure is depicted in Fig. 2. On the other hand, this parametrization can offer itself as a base for the construction of a digital twin to be used in BIM related tasks. In this work we mostly tackle the first part, and show some promising preliminary results on the second one, that should be the focus of a future work.

The contributions of this work are the following:

1. Proposal of a pipeline that, respecting a set of constraints, generates specific or random instances of 3D bridge models. The dimensions, constraints, and component interaction can be adapted to fit specific needs and other bridge types, making our proposed pipeline not only reproducible but also extensible.
2. Use of aforementioned 3D meshes to generate an automatically annotated synthetic dataset for semantic segmentation of the bridge structural components. This dataset is specific to not only the bridge type but also to the fact that it contains images simulating an inspection carried out by a human. This implies that the synthetic images are mostly close-ups of the structure, and not overall views, and therefore harder to segment. To the best of our knowledge, such a dataset does not currently exist for portal frame bridges.
3. A benchmark on the performance gain achieved by using this large synthetic dataset and a very limited number of real annotated images.
4. Some first preliminary experimentation on integrating an image-based semantic segmentation model in a mobile application in order to obtained a labeled 3D point cloud.

The synthetic dataset as well as the code to reproduce or adapt it, are publicly available.

The remaining of this paper is organized as follows: The assumptions and procedure for the 3D model generation and the annotated image dataset are described in Sec. 2 and Sec. 3 respectively. We detail our experimental setup in Sec. 4 and present results in Sec. 5. We then present some preliminary experiments and results on 3D point clouds in Sec. 6. We give some final remarks and future directions in Sec. 7.

2 3D model generation

From a set of rules and conditions, we automatically produce 3D annotated meshes in two widely used and versatile formats, namely Wavefront .obj [29] and Blender [30]. We assume that a bridge consists of hexahedral building blocks, with varied dimensions, translations and rotations. The relative positioning of these

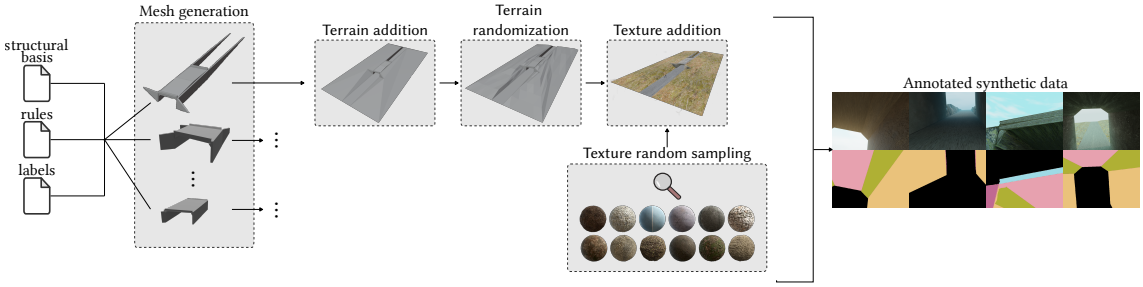

Fig. 2: Overview of synthetic dataset generation.

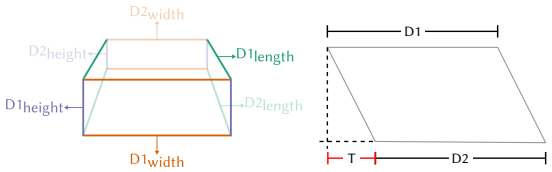

Fig. 3: Parameters defining a building block. **3a**: $D1$ and $D2$ values for each dimension. **3b**: Tilt representation.

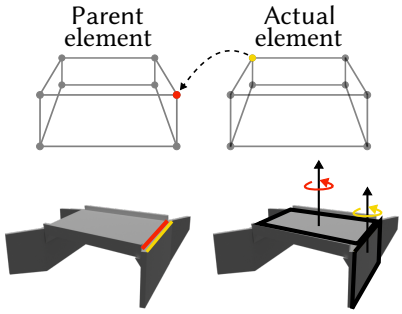

Fig. 4: Constraint examples. **Top:** Positional constraints. **Bottom:** Dimensions and angle constraints. Highlighted red and yellow values should be equal.

blocks with respect to one another instantiates a portal frame bridge. In this work we do not deal with curved elements, we assume all blocks have straight edges. While this is not always the case, it holds for more than 90% of portal frame bridges, as indicated by domain experts.

2.1 Building block description

Each building block belongs to a single semantic category, and its geometry is defined by 15 parameters. We define the vector

$$\begin{aligned}\mathbf{d}_i =&\{D1_i, D2_i, T_i\},\\ &\forall i \in \{\text{width, length, height}\},\end{aligned} \quad (1)$$

where $D1$ and $D2$ correspond to opposing dimensions of a surface, and T denotes the tilt of $D2$ with respect to $D1$. A visual representation of these parameters is shown in Fig. 3. We also define the vector $\mathbf{r} = \{r_x, r_y, r_z\}$, as the block's rotation with respect to each axis, and the vector $\mathbf{t} = \{t_x, t_y, t_z\}$ as the translation of the block's center with respect to the bridge's center. We therefore have the entire building block description,

$$\mathbf{b} = \{\mathbf{d}_{\text{width}}, \mathbf{d}_{\text{length}}, \mathbf{d}_{\text{height}}, \mathbf{r}, \mathbf{t}\}. \quad (2)$$

2.2 Building block constraints

In order to combine various building blocks to form a bridge, several constraints are relevant. Some dimensions or rotations depend on one another, for example the length of the bridge deck should be the same as the length of the abutment. Same goes for their rotation. We also deal with positional constraints, meaning indications on the relative placement of the building blocks, that are crucial for the components to fit together. With the assistance of civil engineer experts we defined a number of constraints as first order multivariate equations of the form:

$$\begin{aligned} f_0(\mathbf{b}_0, ..., \mathbf{b}_{N-1}) &= 0\\ &\vdots\\ f_{M-1}(\mathbf{b}_0, ..., \mathbf{b}_{N-1}) &= 0, \end{aligned} \quad (3)$$

where N is the number of building blocks composing the bridge, and M is the number of constraints. Fig. 4 provides a visual representation of such constraints.

The dimensions of a bridge are bound by a number of factors, that depend on the materials' functionality, the laws of statics etc.([6]). We incorporate those bounds as a number of first order inequality constraints of the form:

$$\begin{aligned} g_0(\mathbf{b}_0, ..., \mathbf{b}_{N-1}) &> 0\\ &\vdots\\ g_{K-1}(\mathbf{b}_0, ..., \mathbf{b}_{N-1}) &> 0, \end{aligned} \quad (4)$$

where N is the number of building blocks composing the bridge, and K is the number of constraints. Such constraints can be either absolute (e.g. a wall must be wider than 30 cm) or relative (e.g. a wing wall's height must be smaller or equal to the height of the abutment it is connected to). The constraint equations are inferred from [6] or indicated by domain experts. We give below the equality and inequality constraints for the deck, and refer the reader to the available code and its documentation for the rest of the building blocks.

$$
\begin{gathered}
2 \leq D1_{\text{width}} \leq 10 \\
2 \leq D2_{\text{width}} \leq 10 \\
2 \leq D1_{\text{length}} \leq 20 \\
2 \leq D2_{\text{length}} \leq 20 \\
-0.5 \cdot D1_{\text{width}} \leq T_{\text{length}} \leq 0.5 \cdot D1_{\text{width}} \\
0.3 \leq D1_{\text{height}} \leq 0.5 \\
0.3 \leq D2_{\text{height}} \leq 0.5 \\
D1_{\text{height}} = 0.045 \cdot D1_{\text{width}} \\
D2_{\text{height}} = 0.045 \cdot D2_{\text{width}} \\
-2.5 \leq r_x \leq 2.5 \\
-5 \leq r_y \leq 5
\end{gathered}
$$

2.3 Randomized geometric model generation

3D models of bridges are randomly generated, while respecting the imposed constraints. All 15 parameters of vector $\mathbf{b}$ (Eq. (2)) for each building block are either sampled from acceptable ranges or directly calculated using the equality constraints (Eq. (3)). The constraints are implemented as parameter files in the model generation pipeline. Table 1 summarizes the overall acceptable dimensions for our setup (portal frame bridges with a single deck), and points out the corresponding building block dimensions that those reflect.

By randomly sampling dimensions from the acceptable ranges and applying the defined constraints, we obtain a randomized 3D instance of a portal frame bridge. The number of parameters for a bridge with 4 wing walls is 165, of which 96 are sampled from a continuous range and not directly calculated. The number of acceptable bridge instances is therefore significant, and a large variability can be assured.

3 Dataset

The 3D model contains all the geometric information of the bridge structure, but in order to produce realistic and varied synthetic images a few more elements are needed.

3.1 Synthetic dataset generation

We use Blender [30] to automatically create and complete the 3D model and produce automatically annotated images. We do so via the following steps (see Fig. 2):

Surroundings completion:

We create a plane hooked on the abutments' corners to represent the road, and we add an uneven terrain as a collection of planar faces around the bridge. We define those planar triangular faces by using the vertex coordinates of the building blocks, so that the terrain is created naturally around the bridge.

Textures:

Following, we randomly sample from a number of appropriate textures, recovered from Poly Haven,[2] and assign a texture to each 3D mesh. The complete set of textures is manually chosen to represent real world conditions. Since portal frame bridges are made of concrete, textures of concrete of different roughness and colorations are chosen. As for the surrounding terrain, earthy textures (e.g. grass, gravel) are selected.

Environmental conditions:

We use the *Dynamic Sky* addon of Blender to randomly set lighting conditions, such as sky color, cloud density and sunlight color. We also use the Blender built-in functionality to add fog.

Camera placement:

We have thus created a realistic instantiation of a bridge from a geometric 3D model. Next, to produce the synthetic images, we place a camera object in the scene. We sample a number of camera positions that simulate a human inspector in terms of height and orientation. The range of camera positions is chosen in accordance to the inspection procedure, meaning there are several views from underneath the bridge, and not only overall views. The inspector is assumed to always stand on the road. Those constraints are assured by defining acceptable coordinate

[2] https://polyhaven.com/

Table 1: Portal frame bridge dimension ranges. In italics, the building block dimensions that the overall bridge dimensions correspond to. Columns **min** and **max** correspond to available ranges. Column **fixed** corresponds to a specific value that is assigned to a dimension.The value of zero to the rotation around the z axis (r_z) is a convention for mathematical simplicity, since any rotation around the z axis would just rotate the entire bridge without changing its geometric representation.

	min	**max**	**fixed**
width [m] (*deck width*)	2	10	-
length [m] (*deck length*, *abutment length*)	2	20	-
height [m] (*abutment height*)	$0.5 \times \text{width}_{\text{deck}}$	$0.7 \times \text{width}_{\text{deck}}$	-
r_x [degrees]	-2.5	2.5	-
r_y [degrees]	-5	5	-
r_z [degrees]	-	-	0

ranges, either absolute (e.g. between 1.4m and 2m for the height) or relative (e.g. between the leftmost and rightmost road coordinates for the width). The constraints are implemented automatically for each bridge, and adapt to the bridge structure and dimensions. The camera positions are then randomly sampled for each separate bridge.

Post processing:

Despite the care taken to generate reasonable images, the camera coordinates can give irrelevant images. To counter this, we deploy an automatic post processing step, where all images that contain less than a user defined percentage of bridge elements are discarded. Fig. 5 shows some examples of such images.

Annotations:

Each structural component is represented by a different object (mesh) in the 3D model, and each object is assigned a semantic class id. The image annotations are automatically generated in the form of binary masks for each category with Blender [30] during rendering. The interested reader is referred to the available code for the implementation details of the binary mask generation.

Examples of retained images and their ground truths are shown in Fig. 10.

3.2 Dataset details and statistics

We automatically generate 500 3D models of portal frame bridges, and sample 100 locations per bridge. We choose an image resolution of 640×480 pixels, and set the image coverage threshold (percentage of image that should be covered by bridge elements) to 10%. We also dispose a limited set of real images, annotated by experts. The final number of images per set, and their corresponding label distribution are summarized in Table 2.

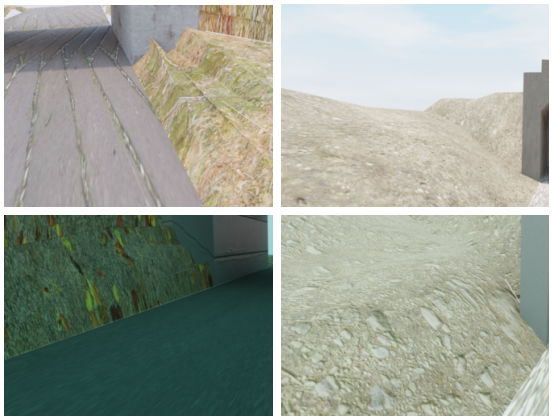

Fig. 5: Examples of discarded images after post processing (the threshold of the image surface that should be covered by the bridge is set to 10%).

4 Experiments

4.1 Semantic segmentation with finetuning

We first trained a semantic segmentation model with the synthetic images. After obtaining the best possible performance on the synthetic dataset, we switched to the limited available real images and further refined the network's

Table 2: Images and label distribution per dataset.

	Synthetic *(PFBridge)*	Real
# of images	3364	146
Background(%)	37.7	56.9
Deck (%)	19.9	15.5
Abutment (%)	20.4	12.4
Wing wall (%)	10.4	7.9
Haunch (%)	9.8	5
Edge beam (%)	1.7	2.3

parameters. The overall training and finetuning pipeline is presented in Fig. 6.

To test the role of the synthetic dataset in effectively training a semantic segmentation model, we trained such models with and without synthetic data. To examine the results' consistency, we performed experiments with different network architectures and initializations.

4.2 Network and training details

We use a Unet [31] architecture for our experiments. Unet is a widely used fully convolutional neural network, that relies on an encoder branch that extracts useful features of an image, followed by a decoder branch that utilizes features of different encoder levels to generate segmentation masks (see Fig. 7). We used a MobileNetV2 [32] backbone as the encoder. MobileNetV2 uses depthwise separable convolutions, factorizing a full convolutional operation into a pair of a depthwise and a pointwise convolution, to reduce computational complexity. The power of this architecture lies in the use of inverted residual modules with linear bottlenecks. Such a module is a block that takes a low dimensional representation, expands it, and reprojects it to a low dimensional space after having performed a depthwise filtering operation. The linear bottleneck layers prevent information loss caused by non-linearities.

We trained for 100 epochs, that were empirically enough to guarantee convergence, with a categorical cross entropy loss, and we used the Adam optimizer [33]. The encoders were initialized with weights trained to perform classification on the Imagenet dataset [28]. The datasets were split to a 60%-20%-20% training, validation, and test set respectively. For the synthetic dataset, the split was made on a bridge level (all images of the same bridge belong to the same set) to avoid bias in the results. The final number of images per set and dataset are presented in Table 3. The model weights that obtain the lowest loss in the validation set are retained and evaluated on the test set.

Table 3: Number of images per set.

	Synthetic *(PFBridge)*	Real
Training set	1911	88
Validation set	699	29
Test set	662	30

The focus of this work is not to design or optimize the CNN, but rather to design and prove the importance of the synthetic dataset. We did not delve into the formulation of an architecture tuned for our task, nor in the search for the optimal network. We chose architectures that are widely recognized as powerful and robust, in order to emphasize on the role of the dataset during training. We were also limited by the fact that the developed model should be fast enough to be integrated in a mobile application. There was thus a trade-off between the performance and the inference speed to be taken into account into the model architecture selection.

4.3 Evaluation setup

To demonstrate the synthetic dataset relevance:

- We train and test a model exclusively on synthetic data.
- We use the model trained exclusively on synthetic data to infer on real data. This way we get an idea of the domain gap between real and synthetic datasets.
- We finetune the model trained on the synthetic data with real data, and we then test on real data.
- We train and test a model exclusively on real data. This way we see whether the synthetic dataset improves the network's performance.

5 Results

We now report results on the dataset generation and comment on its usefulness for training an effective semantic segmentation model.

5.1 Quantitative results

Table 4 presents the F1-score of the CNN model, per training and testing configuration (see Sec. 4.3 for configurations). We report the F1-score per structural category, and for the entirety of the dataset. The F1-score is

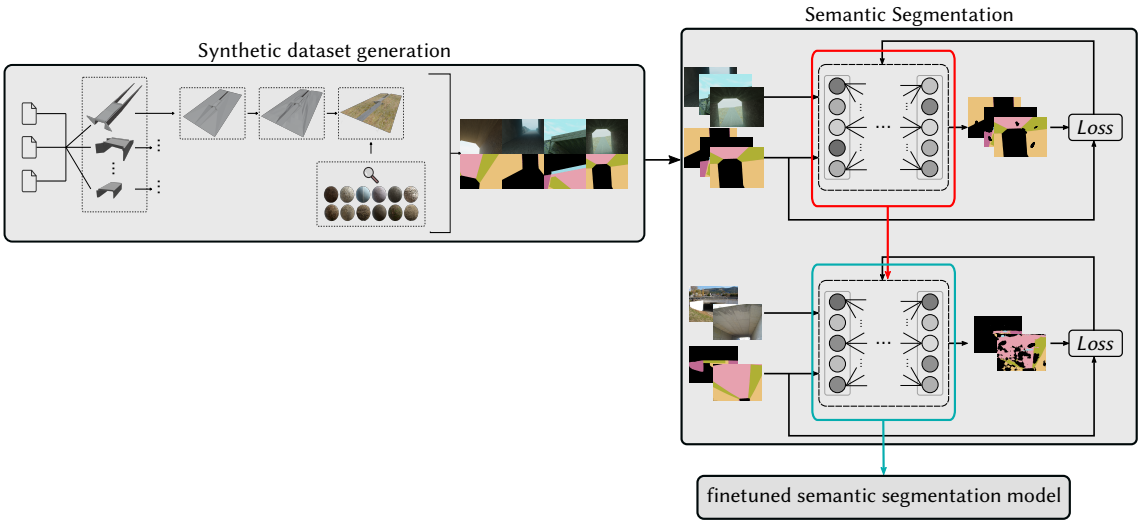

Fig. 6: Overview of semantic segmentation model training. A model is first trained on a vast number of synthetic images (weights in red box). Once optimized on them, the model is further refined by being trained on a limited number of more complex, manually annotated, real images (weights in blue box).

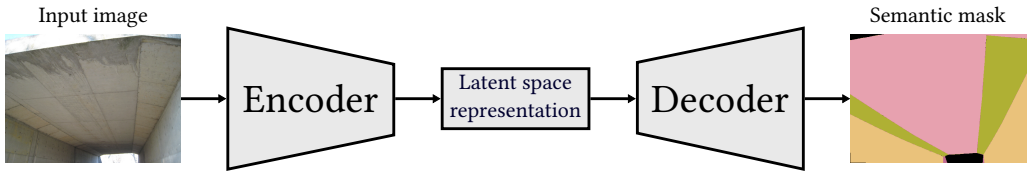

Fig. 7: Overview of encoder-decoder architecture. An image is first passed through an encoder branch, where a latent space representation is produced. The representation is then feeded to the decoder branch, which produces segmentation masks of size same as the original image.

chosen for being a metric appropriate for unbalanced datasets where the distribution of labels is not close to uniform, as is our case. Table 5 presents the mean intersection over union (IoU) for the different evaluation setups. Formally, the F1-score is defined as the harmonic mean of precision and recall, and is given by the formula

$$\mathrm{F1} = \frac{1}{N}\sum_{i=1}^{N}\frac{2\cdot \mathrm{TP}_i}{2\cdot \mathrm{TP}_i + \mathrm{FP}_i + \mathrm{FN}_i}, \qquad (5)$$

where TP, FP, FN are the true positive, false positive, and false negative predicted values respectively, and N is the number of classes. The mean IoU is calculated as:

$$\mathrm{IoU} = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{TP}_i}{\mathrm{TP}_i + \mathrm{FP}_i + \mathrm{FN}_i}. \qquad (6)$$

Fig. 8 and Fig. 9 show the confusion matrices for the different experimental setups, normalized over the true labels and the predictions respectively.

The results expose the undeniable added value of the synthetic dataset. The nature of the data causes some classes to be easily confused with their neighboring elements depending on the image point of view. For example the haunch is really fused between the deck and the abutment, and an image containing only two of the three is not straightforward to segment. Same goes for the wing wall and the abutment, an image containing a close up of their joint might, on its own, be inconclusive regarding which one is which. The confusion matrices confirm that several misclassifications are actually reflections of such inherent properties. Also, some classes have very different appearances when observed from a short distance (as in the synthetic dataset) and when observed from a larger distance (as in most examples the real dataset), namely the abutment and mostly the haunch. Even for those, the real dataset alone is extremely inefficient on its own, which is proven by the fact that the best results in those edge cases are achieved by the models pretrained solely on synthetic data. More, and

Table 4: F1-score on test sets, per class and on the entire dataset, for the different evaluation setups described in Sec. 4.3. A MobileNetV2 network pretrained on Imagenet is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Background	Deck	Abutment	Wing wall	Haunch	Edge beam	Total (macro)
TrS-TeS	0.99	0.97	0.97	0.91	0.95	0.93	0.95
TrS-TeR	0.84	0.50	0.30	0.25	0.13	0.4	0.34
TrR-TeR	0.82	0.46	0.30	0.45	0.02	0.2	0.39
TrSR-TeR	0.92	0.85	0.61	0.58	0.49	0.59	0.67

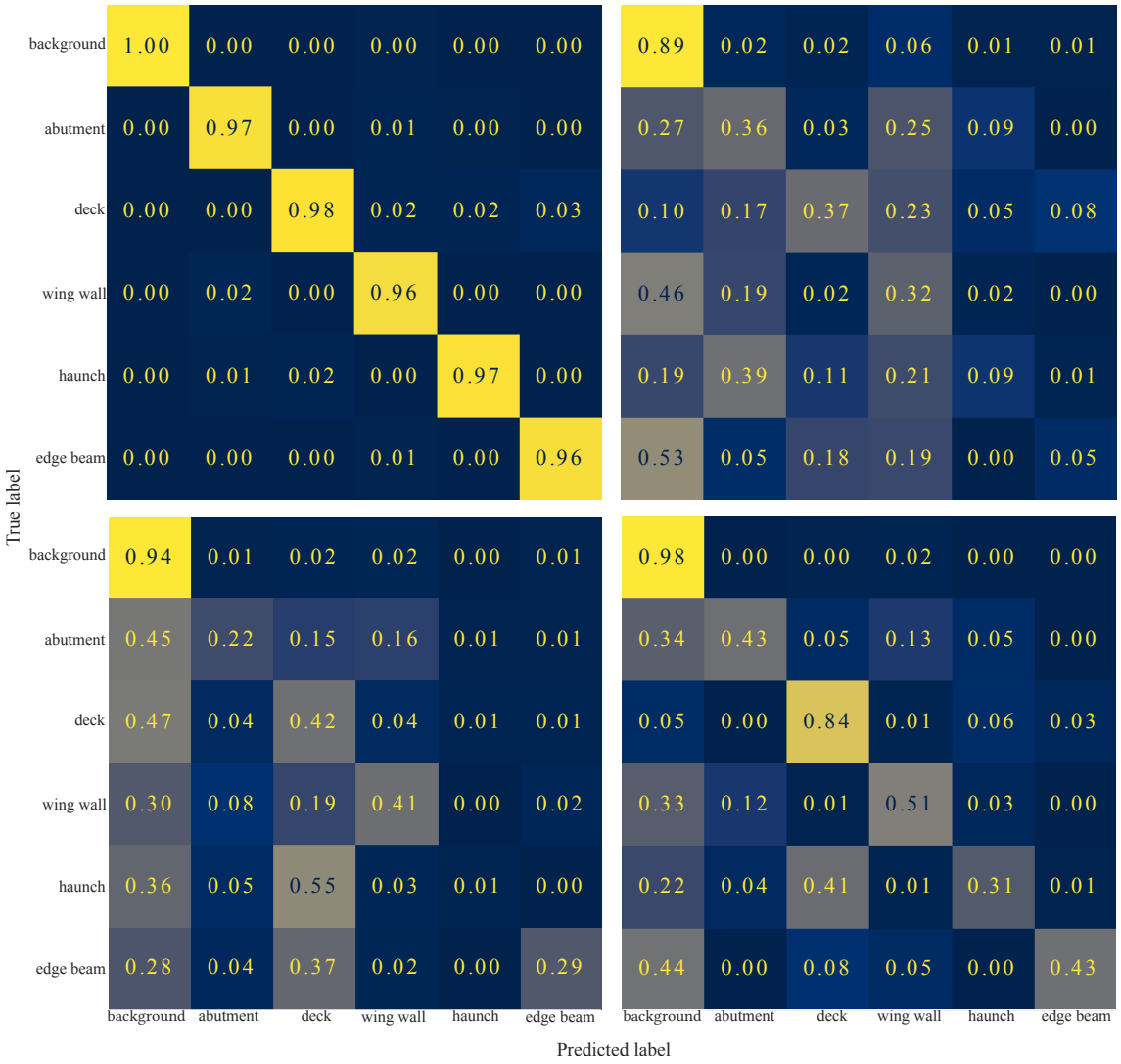

Fig. 8: Confusion matrices for Unet with Imagenet pretrained MobileNetV2 backbone. Normalized over true labels (rows). **Top left:** Training and testing on synthetic data (TrS-TeS). **Top right:** Training on synthetic data, testing on real data (TrS-TeR). **Bottom left:** Training and testing on real data (TrR-TeR).**Bottom right:** Training on synthetic data, finetuning with real data, testing on real data (TrSR-TeR).

more relevant real images, along with richer representations in the synthetic dataset could address some of those limitations. We comment on the results with more detail in Sec. 5.3.

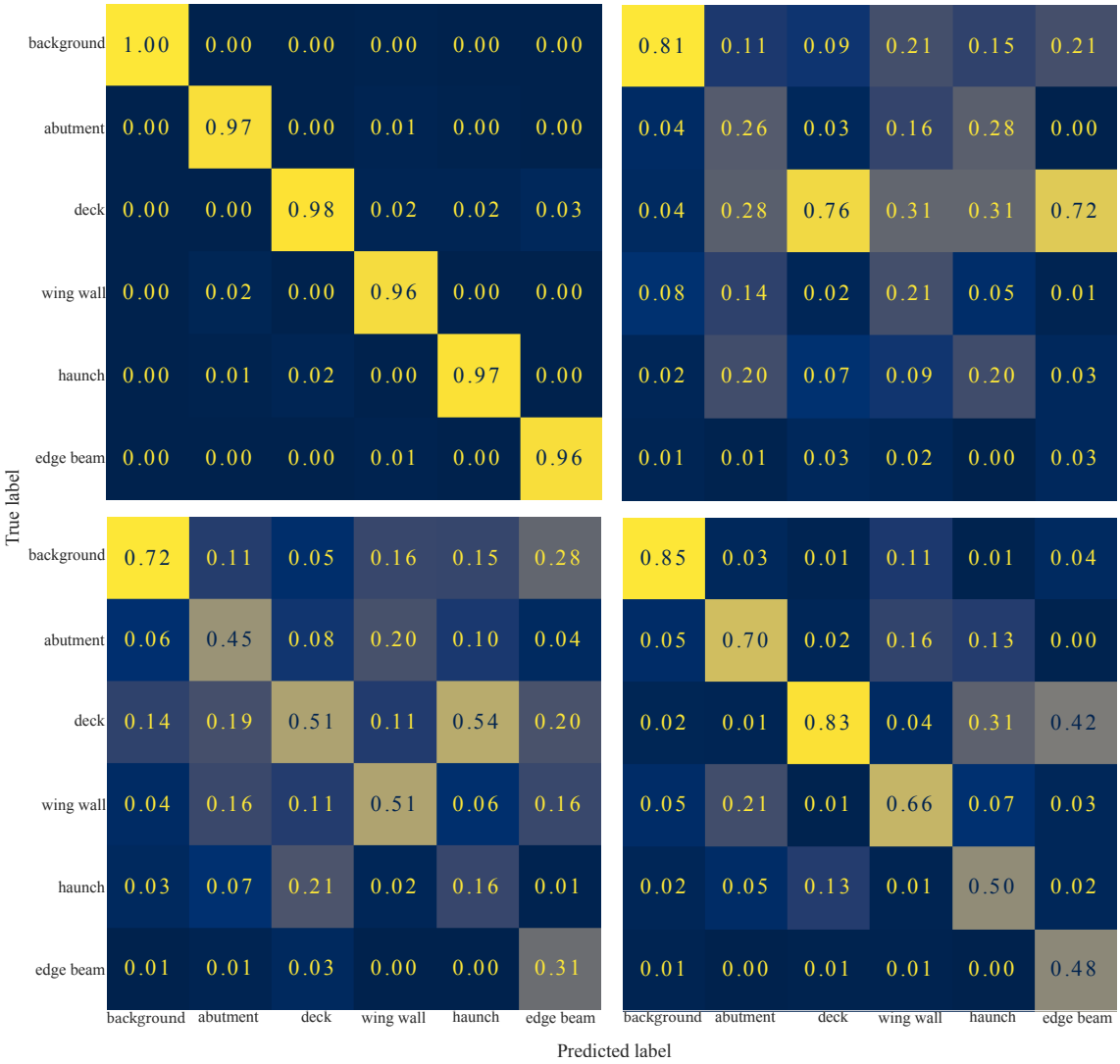

Fig. 9: Confusion matrices for Unet with Imagenet pretrained MobileNetV2 backbone. Normalized over predictions (columns). **Top left:** Training and testing on synthetic data (TrS-TeS). **Top right:** Training on synthetic data, testing on real data (TrS-TeR). **Bottom left:** Training and testing on real data (TrR-TeR).**Bottom right:** Training on synthetic data, finetuning with real data, testing on real data (TrSR-TeR).

We also present quantitative results for other encoder backbones in Appendix A. Similar patterns are observed, confirming that the contribution of the synthetic dataset is robust and not dependent on the specific architecture.

We also trained a model exclusively on our synthetic dataset (with random weight initialization). We then used both models to infer the semantic segmentation masks of the real images. This experiment allows us to compare the predictive power of a model pretrained on our synthetic dataset alone, with a model pretrained on Imagenet. To have a fairer comparison, we trained the last layer of the model trained on Imagenet with the real bridge images, since the classes are not the same between the two sets. The results are presented in Table 6 and Table 7. Despite its limitations, the pretraining on our synthetic dataset is by far superior to the pretraining on real natural images unrelated to bridges, even when the real bridge images are used for the final layer. These results imply that pretraining with our dataset constitutes a much more appropriate starting point for the bridge component semantic segmentation task.

5.2 Qualitative results

We present qualitative results for the Unet with the pretrained MobileNetV2 backbone. Fig. 10 shows examples of synthetic images, along with

Table 5: Mean IoU on test sets on the entire dataset, for the different evaluation setups described in Sec. 4.3. A MobileNetV2 network pretrained on Imagenet is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Intersection over Union (IoU)
TrS-TeS	0.91
TrS-TeR	0.25
TrR-TeR	0.27
TrSR-TeR	0.53

their ground truths and the network's predictions. Fig. 11 shows examples of real images, along with their ground truths and the predictions of the model trained solely with real data, as well as the predictions of the finetuned model. One can clearly see that the finetuned model yields much more reasonable results, and identifies in a better and cleaner way the structural elements of the bridge, even without any post-processing.

5.3 General comments on the results

It should be noted, that despite the relative simplicity of the structure in terms of shape, there is a significant complexity regarding texture. That is, all elements of a bridge are made from the same material, making texture an inappropriate criterion for class separation. The identification of the components needs to rely on context, on the relative position of the elements with respect to one another. That, combined with the fact that the large majority of the produced views are close ups, meant to simulate images taken by an inspector, make the task non trivial.

The results confirm our expectation that training on the synthetic dataset and finetuning on a very small real dataset is highly beneficial. Similar improvement patterns can be observed for the different architectures and initializations, thus strengthening our conclusions. The real dataset size is inadequate to be used on its own. However, such limited real images can play a non trivial role in bridging the domain gap between synthetic and real bridges. This implies, that the proposed synthetic dataset already contains a significant part of the information that are important in the real domain, more than the information contained in real natural images (Imagenet). Even thought the results of the finetuned models are not as good as those that are achieved in the synthetic datasets, which is both common and expected in those setups, there is a remarkable improvement when the synthetic images are used for pretraining. We should also point out here, that while we created the synthetic images with the task of a close inspection in mind, several of the real images that we possess are from a further, overall view of the bridge. Therefore, the observed domain gap is not only due to the limits of the synthetic images in terms of realism, but also due to the slightly inappropriate real dataset.

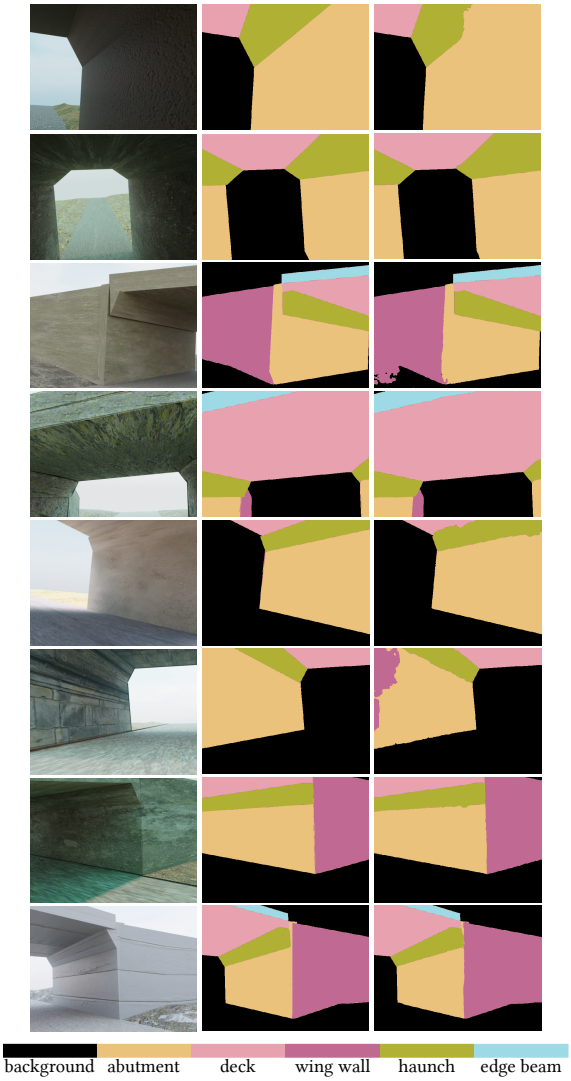

Fig. 10: Qualitative results on synthetic images. The prediction model is a Unet with an Imagenet pretrained MobileNetV2 as backbone, *trained exclusively on synthetic data*. **Left:** Image. **Middle:** Ground truth masks. **Right:** Network predictions.

Table 6: F1-score on the real bridge images, per class and on the entire dataset, for two networks, one trained on Imagenet and one trained on our synthetic dataset (PFBridge). A MobileNetV2 network is used as the encoder backbone.

	Background	Deck	Abutment	Wing wall	Haunch	Edge beam	Total (macro)
Imagenet	0.49	0	0.08	0.01	0	0.03	0.06
PFBridge	0.84	0.46	0.25	0.23	0.29	0.04	0.35

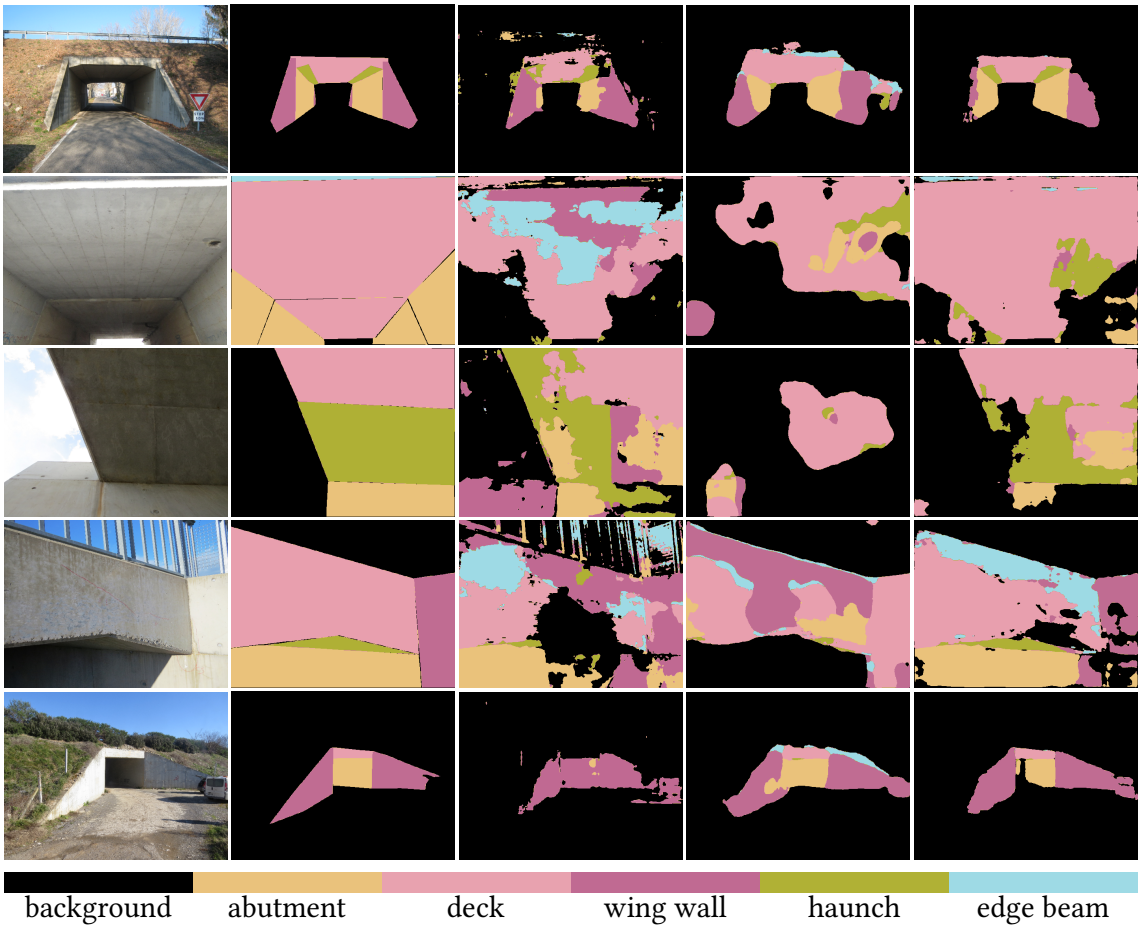

Fig. 11: Qualitative results on real test images. The prediction model is a Unet with an Imagenet pretrained MobileNetV2 as backbone. **Column 1:** Image. **Column 2:** Ground truth masks. **Column 3:** Predictions of network *trained exclusively on synthetic data (TrS-TeR)*. **Column 4**: Predictions of network *trained exclusively on real data (TrR-TeR)*. **Column 5**: Predictions of network *trained on synthetic data and finetuned with real data (TrSR-TeR)*.

Table 7: Mean IoU on the real bridge images, per class and on the entire dataset, for two networks, one trained on Imagenet and one trained on our synthetic dataset (PFBridge). A MobileNetV2 network is used as the encoder backbone.

	Intersection over Union (IoU)
Imagenet	0.06
PFBridge	0.25

We did not take any steps to clean or refine the output masks of the real dataset, since our goal was strictly to show the contribution of the synthetic dataset. The results leave no doubt that the synthetic data provide a valuable way to mitigate the lack of real annotated data for semantically segmenting bridge structural components, and could be even more valuable if coupled with post processing refinement techniques.

6 Preliminary tests on 3D point cloud

This work focuses on the generation of synthetic images from synthetic geometric models, and on the value of such images in the image semantic segmentation task. This is only a step in the overall bridge monitoring process. One of the potential uses of the image segmentation is to use it in order to label 3D point clouds during inspection. In a future work, a labeled point cloud, along with the parametrizable 3D model described in Sec. 2, could be used to generate the bridge's digital twin. More precisely, the set of points belonging to a class, along with the constraints defined in Eq. (3) and Eq. (4) could define a constrained optimization problem, in order to derive the values of vector $\mathbf{b}$(Eq. (2)) of a building block. A set of parametrized building blocks $\mathbf{b}$ define a complete geometric 3D model, in other words, the digital twin of the bridge.

The developed model was integrated in a mobile application that uses mobile LiDAR to capture a 3D point cloud. Thanks to our model, each (x, y, z) coordinate in 3D space is associated with a semantic category. Manually annotating a 3D point cloud can be very time consuming, even prohibiting, since they typically consist of more than a million points. By using a model trained on images to annotate it we overcome this obstacle. In more detail, each video frame during scanning is annotated by the model. Each frame pixel is associated with a 3D coordinate, therefore each 3D coordinate is annotated. In the case of a 3D coordinate getting different annotations from different frames, a majority filter is applied to retain one annotation per point. We tested the application in an actual portal frame bridge, using an iPad Pro 4th generation with a 256GB memory and embedded LiDAR. Some example frames along with the model's predictions that were produced during scanning are shown in Fig. 12.

The 3D model of the above bridge, representing its digital twin, was manually created with the parametrization described in Sec. 2, and can be seen in Fig. 13.

6.1 Point cloud post processing

After the acquisition of the point cloud, we can use some prior knowledge in order to refine the predicted annotations, namely:

Fig. 12: Example frames of demo bridge on which the annotated point cloud acquisition was tested, along with the per frame semantic predictions during scanning.

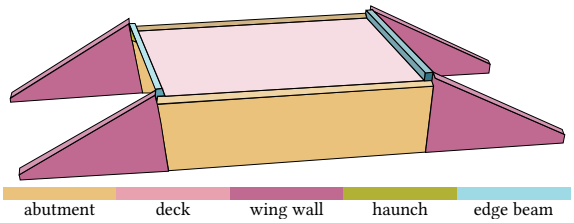

Fig. 13: 3D model of demo bridge on which the annotated point cloud acquisition was tested.

Majority filtering:

We know that the structural elements are consistent blocks, and small blobs of points annotated different than their surroundings constitute noise. We therefore apply a majority filter in order to reassign each point's label to the label of the majority of its neighbors in a 15cm radius.

Element position:

We know the relative position of some elements on the bridge structure. In particular the *deck*, *haunch*, and *edge beams* are always located on the top part of the bridge. Any points in the lower part of the bridge that are assigned to the aforementioned classes are therefore discarded.

Elements as planes:

We know that the structural elements are consisted of planar surfaces. We use the RANSAC [34] algorithm to segment a plane that contains the majority of the points belonging to a class, and discard as outliers all points that are not close enough to this plane. The RANSAC algorithm is run for 1000 iterations with 7 points randomly sampled to estimate a plane. The tolerance distance for a point to be considered inlier is 13cm from the estimated plane.

6.2 Preliminary qualitative results

The annotated point cloud that was automatically produced in the preliminary tests on the bridge shown in Fig. 12 is presented in Fig. 14.

Despite the point cloud being relatively noisy, we can clearly distinguish the different structural elements, and we confirm that all relevant structures are correctly represented by groups of pixels. After the post processing steps we get a point cloud that is even cleaner. Despite the presence of some noise and gaps, the remaining points along with the construction constraints of portal frame bridges can be adequate for the digital twin construction. After all, the goal of this experiment is not to obtain a perfectly labeled point cloud, but one that would be sufficient to define the bridge's dimensions. We are therefore confident that such a point cloud can serve as a good basis in order to infer the geometrical parameters of the bridge and automatically produce a digital twin that closely resembles that of Fig. 13.

7 Conclusions and future work

We devised an automatic pipeline that, respecting some predefined rules can create an infinite number of valid 3D models of portal frame bridges. We used these models to generate a synthetic dataset of 3364 annotated images of 500 bridges, that we named **PFBridge** and made publicly available. We use these images to address bridge component semantic segmentation, a task that is a part of automated or semi-automated structural health monitoring (SHM). Our experiments show that there is a significant improvement to a CNN's performance when it is pretrained with the synthetic dataset and then finetuned with a limited number of real images, as opposed to only training with the real images, or pre-training with irrelevant natural images. Our pipeline is extensible, customizable, and saves a lot of domain experts' time. All the aforementioned allow us to be confident that the proposed method can be used to assist automations in bridge inspections. We also performed some preliminary experiments by integrating the developed model in a mobile application, that uses it to acquire an annotated point cloud of a given bridge. The preliminary results are promising and show potential of using such a point cloud to automatically or semi-automatically infer the exact parameters of a 3D model corresponding to a given bridge, thus producing its digital twin. In such a scenario, an uncertainty criterion can also be considered, where, for dimensions that cannot be inferred from the point cloud with sufficient accuracy, a manual measurement will be requested from the user.

As future work, we plan to expand the geometrical representation to more bridge types, and also allow for more complex surfaces, e.g. curved. Higher order constraints could also be implemented, if they are relevant. In order to narrow the domain gap, we would enrich the synthetic environment with elements such as cars, people, vegetation, painted walls etc. We also plan to enrich the real dataset with a few more images, specifically images corresponding to our use case scenario, meaning images taken from under the bridge or close to its edges. The time performance and optimization of the model for mobile applications are also some important next steps, to achieve a more seamless mobile application integration, and make for an easier use in bridge inspection. No special care was taken in this work to handle the class imbalance, nor pre or post-processing steps were taken to refine the results. The specific nature of the problem (bridge components are solid elements, without holes etc.) could provide insightful priors. Finally, we aim on improving the developed image-based point cloud semantic segmentation, and use the annotated point cloud in order to create a geometric model of

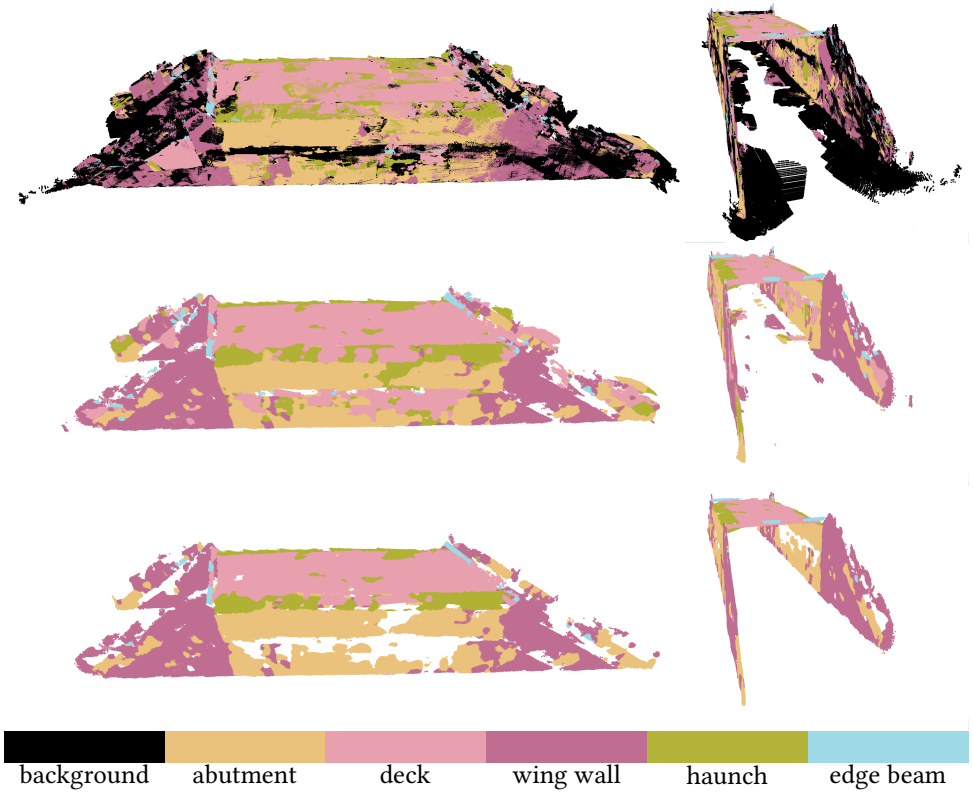

Fig. 14: Annotated 3D point cloud of portal frame bridge. **Top row:** Annotated point cloud returned from the mobile application. **Middle row:** Annotated point cloud after background removal and majority filtering. **Bottom row:** Annotated point cloud after outlier removal (removal of points that are not close enough to a segmented plane per building block)

the visited bridge (digital twin). The same parametric model that was used to generate the synthetic dataset can be used for the digital twin estimation, via constrained optimization methods.

Declarations

Availability of data and material

The code used for the experiments of this work and some sample parametric `.json` files are publicly available in the following repository: https://github.com/tati-/AI_assisted_bridge_inspection

The synthetic annotated images are publicly available in: https://www.kaggle.com/datasets/tatianafou/pfbridge-synthetic-dataset/data, **DOI**: 10.34740/kaggle/ds/5013350

The annotated real images are confidential and limited to the authors.

Competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

Author contributions

All authors read and approved the submitted manuscript.

Funding

The project MIRAUAR received a grant from the call for projects Ponts Connectés funded by the French State and led by Cerema which aims to improve bridge management by using the most recent techniques in terms of monitoring, data transfer and data processing.

Acknowledgments

This work was a collaboration of Diadès Setec, Sogelink, and the LIRIS Laboratory of the Université Lyon 2. The authors wish to thank all involved parties for their help in the project. Special thanks to Kevin Etourneau and Alexis

Delforges from Sogelink for their help in integrating the semantic segmentation model in the bridge inspection application, in order to obtain the annotated point cloud.

The authors would like to thank Cerema for organizing this call for projects and for its scientific involvement in the follow-up of the project.

References

[1] Rashidi, M., Ghodrat, M., Samali, B., Kendall, B., Zhang, C.: Remedial modelling of steel bridges through application of analytical hierarchy process (ahp). Applied Sciences **7**(2), 168 (2017)

[2] Song, G., Wang, C., Wang, B.: Structural health monitoring (SHM) of civil structures. MDPI (2017)

[3] Vardanega, P.J., Webb, G.T., Fidler, P.R.A., Huseynov, F., Kariyawasam, K.K.G.K.D., Middleton, C.R.: 32 - bridge monitoring. In: Innovative Bridge Design Handbook (Second Edition), pp. 893–932. Butterworth-Heinemann, ??? (2022). https://doi.org/10.1016/B978-0-12-823550-8.00023-8

[4] Bao, Y., Li, J., Nagayama, T., Xu, Y., Spencer Jr, B.F., Li, H.: The 1st international project competition for structural health monitoring (ipc-shm, 2020): a summary and benchmark problem. Structural Health Monitoring **20**(4), 2229–2239 (2021)

[5] Huthwohl, P., Lu, R., Brilakis, I.: Challenges of bridge maintenance inspection (2016)

[6] Le Khac, V., Millan, A.L., Faure, E., Gilcart, J.P.: Ponts - cadres et portiques, guide de conception. Technical report, Centre des Techniques des ouvrages d'art (IQOA) (1992)

[7] Dang, N., Shim, C.: Bim-based innovative bridge maintenance system using augmented reality technology. In: CIGOS 2019, Innovation for Sustainable Infrastructure: Proceedings of the 5th International Conference on Geotechnics, Civil Engineering Works and Structures, pp. 1217–1222 (2020). Springer

[8] Kilic, G., Caner, A.: Augmented reality for bridge condition assessment using advanced non-destructive techniques. Structure and Infrastructure Engineering **17**(7), 977–989 (2021)

[9] SETRA: Iqoa - image de la qualité des ouvrages d'art. Technical report (1996)

[10] Administration), F.F.H.: National bridge inspection standards. Federal Register **69**(239), 74419–39 (2004)

[11] Hiasa, S., Catbas, F.N., Matsumoto, M., Mitani, K.: Monitoring concrete bridge decks using infrared thermography with high speed vehicles. Structural Monitoring and Maintenance **3**(3), 277–296 (2016)

[12] Matsumoto, M., Mitani, K., Catbas, F.N.: Bridge assessment methods using image processing and infrared thermography technology. In: Proc., 92nd Annual Meeting, Transportation Research Board, Washington, DC (2013)

[13] Lin, Y.-C., Habib, A.: Semantic segmentation of bridge components and road infrastructure from mobile lidar data. ISPRS Open Journal of Photogrammetry and Remote Sensing **6**, 100023 (2022)

[14] Yang, X., Rey Castillo, E., Zou, Y., Wotherspoon, L., Tan, Y.: Automated semantic segmentation of bridge components from large-scale point clouds using a weighted superpoint graph. Automation in Construction **142**, 104519 (2022)

[15] Lee, J.S., Park, J., Ryu, Y.-M.: Semantic segmentation of bridge components based on hierarchical point cloud model. Automation in Construction **130**, 103847 (2021)

[16] Yang, L., Lin, Y.-C., Cai, H., Habib, A.: From scans to parametric bim: An enhanced framework using synthetic data augmentation and parametric modeling for highway bridges. Journal of Computing in Civil Engineering **38**(3), 04024008 (2024)

[17] Ham, Y., Han, K.K., Lin, J.J., Golparvar-Fard, M.: Visual monitoring of civil infrastructure systems via camera-equipped unmanned aerial vehicles (uavs): a review

of related works. Visualization in Engineering **4**(1), 1–8 (2016)

[18] Gao, Y., Mosalam, K.M.: Deep transfer learning for image-based structural damage recognition. Computer-Aided Civil and Infrastructure Engineering **33**(9), 748–768 (2018)

[19] Liang, X.: Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with bayesian optimization. Computer-Aided Civil and Infrastructure Engineering **34**(5), 415–430 (2019)

[20] Yu, W., Nishio, M.: Multilevel structural components detection and segmentation toward computer vision-based bridge inspection. Sensors **22**(9), 3502 (2022)

[21] Bianchi, E., Hebdon, M.: Visual structural inspection datasets. Automation in Construction **139**, 104299 (2022)

[22] Bianchi, E., Abbott, A.L., Tokekar, P., Hebdon, M.: Coco-bridge: structural detail data set for bridge inspections. Journal of Computing in Civil Engineering **35**(3), 04021003 (2021)

[23] Narazaki, Y., Hoskere, V., Yoshida, K., Spencer, B.F., Fujino, Y.: Synthetic environments for vision-based structural condition assessment of japanese high-speed railway viaducts. Mechanical Systems and Signal Processing **160**, 107850 (2021)

[24] Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3234–3243 (2016)

[25] Quattrocchi, C., Di Mauro, D., Furnari, A., Lopes, A., Moltisanti, M., Farinella, G.M.: Put your ppe on: A tool for synthetic data generation and related benchmark in construction site scenarios. In: VISIGRAPP (4: VISAPP), pp. 656–663 (2023)

[26] Narazaki, Y., Hoskere, V., Hoang, T.A., Spencer Jr, B.F.: Automated bridge component recognition using video data. arXiv preprint arXiv:1806.06820 (2018)

[27] Narazaki, Y., Hoskere, V., Hoang, T.A., Fujino, Y., Sakurai, A., Spencer Jr, B.F.: Vision-based automated bridge component recognition with high-level scene consistency. Computer-Aided Civil and Infrastructure Engineering **35**(5), 465–482 (2020)

[28] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009). Ieee

[29] Wavefront .OBJ. https://en.wikipedia.org/wiki/Wavefront_.obj_file. Accessed: 2023-07-03

[30] Community, B.O.: Blender - a 3D Modelling and Rendering Package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018). Blender Foundation. http://www.blender.org

[31] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, pp. 234–241 (2015). Springer

[32] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)

[33] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[34] Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)

[35] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

[36] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

Appendix A Quantitative results for different encoder backbones.

In order to demonstrate that the synthetic dataset's contribution is beneficial for more than just one networks, we performed the experiments desrcibed in Sec. 4.3 for different encoder backbones. Similar patterns are exposed into all the experiments, confirming the contribution of the synthetic dataset in the semantic segmentation learning.

Table A1 and Table A2 show the F1-score and IoU respectively, for a MobileNetv2 ([32]) backbone with random weight initialization. This experiment shows that the synthetic data alone are capable of providing a reasonable prior. Table A3 and Table A4 show the F1-score and IoU respectively, for a simple MobileNet ([35]) backbone pretrained on Imagenet. Table A5 and Table A6 show the F1-score and IoU respectively, for a ResNet50 ([36]) backbone pretrained on Imagenet. The ResNet architecture uses skip connections to force the layers to represent residual functions with reference to the layer input, instead of learning unreferenced functions.

Table A1: F1-score on test sets, per class and on the entire dataset, for the different evaluation setups described in Sec. 4.3. A MobileNetv2 network with random weight initialization is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Background	Deck	Abutment	Wing wall	Haunch	Edge beam	Total (macro)
TrS-TeS	0.98	0.94	0.94	0.87	0.87	0.90	0.92
TrS-TeR	0.84	0.46	0.25	0.23	0.29	0.04	0.35
TrR-TeR	0.65	0.09	0	0	0	0	0.12
TrSR-TeR	0.88	0.75	0.43	0.52	0.18	0.30	0.51

Table A2: IoU on test sets on the entire dataset, for the different evaluation setups described in Sec. 4.3. A MobileNetv2 network with random weight initialization is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Intersection over Union (IoU)
TrS-TeS	0.84
TrS-TeR	0.25
TrR-TeR	0.09
TrSR-TeR	0.38

Table A3: F1-score on test sets, per class and on the entire dataset, for the different evaluation setups described in Sec. 4.3. A MobileNet network pretrained on Imagenet is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Background	Deck	Abutment	Wing wall	Haunch	Edge beam	Total (macro)
TrS-TeS	0.99	0.97	0.97	0.91	0.95	0.93	0.95
TrS-TeR	0.86	0.52	0.32	0.32	0.06	0.02	0.35
TrR-TeR	0.93	0.77	0.64	0.52	0.40	0.55	0.63
TrSR-TeR	0.92	0.83	0.64	0.57	0.60	0.41	0.66

Table A4: Mean IoU on test sets on the entire dataset, for the different evaluation setups described in Sec. 4.3. A MobileNet network pretrained on Imagenet is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Intersection over Union (IoU)
TrS-TeS	0.91
TrS-TeR	0.26
TrR-TeR	0.49
TrSR-TeR	0.52

Table A5: F1-score on test sets, per class and on the entire dataset, for the different evaluation setups described in Sec. 4.3. A ResNet50 network pretrained on Imagenet is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Background	Deck	Abutment	Wing wall	Haunch	Edge beam	Total (macro)
TrS-TeS	0.98	0.95	0.94	0.86	0.92	0.89	0.93
TrS-TeR	0.87	0.45	0.38	0.19	0.2	0.04	0.36
TrR-TeR	0.69	0	0	0	0	0	0.12
TrSR-TeR	0.92	0.79	0.63	0.53	0.38	0.50	0.62

Table A6: Mean IoU on test sets on the entire dataset, for the different evaluation setups described in Sec. 4.3. A ResNet50 network pretrained on Imagenet is used as the encoder backbone. **TrS-TeS:** Trained and tested on synthetic data. **TrS-TeR:** Trained on synthetic data, tested on real data. **TrR-TeR:** Trained and tested on real data. **TrSR-TeR:** Trained on synthetic data, finetuned with real data, and tested on real data.

	Intersection over Union (IoU)
TrS-TeS	0.86
TrS-TeR	0.25
TrR-TeR	0.09
TrSR-TeR	0.48